Evaluation criteria for trust models with specific reference to prejudice filters

By

Marika Wojcik

Submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE (Computer Science)

In the Faculty of Engineering, Built Environment and Information Technology
University of Pretoria
Pretoria

PRETORIA **March 2007**

I declare that

Evaluation criteria for trust models with specific reference to prejudice filters

Is my own work and that all sources that I have used or quoted have been indicated and acknowledged by means of complete references.

Acknowledgements

I would hereby like to express my sincere thanks and gratitude towards:

- Prof JHP Eloff and Dr HS Venter for their leadership and assistance.
- Willie Moller for his programming assistance.
- My family for their encouragement and support.
- Telkom Center of Excellence and the National Research Foundation (NRF) for their financial support. Opinions expressed and conclusions derived are solely those of the author and not necessarily attributed to Telkom and the NRF.
- The Lord for his grace and mercy during this challenging time.

Abstract

Evaluation criteria for trust models with specific reference to prejudice filters

Candidate: Marika Wojcik
Study Leader: Prof JHP Eloff
Co-supervisor: Dr HS Venter
Department: Computer Science
Degree: MSc. (Computer Science)

The rapid growth of the Internet has resulted in the desperate need for alternative ways to keep electronic transactions secure while at the same time allowing entities that do not know each other to interact. This has, in turn, led to a wide area of interest in the issues of trust and trust modeling to be used by machines. A large amount of work has already been undertaken in this area in an attempt to transfer the trust and interaction decision making processes onto the machine. However this work has taken a number of different approaches with little to no correlation between various models and no standard set of criteria was even proposed that can be used to evaluate the value of such models.

The proposed research chooses to use a detailed literature survey to investigate the current models in existence. This investigation focuses on identifying criteria that are required by trust models. These criteria are grouped into four categories that represent four important concepts to be implemented in some manner by trust models: trust representation, initial trust, trust update and trust evaluation. The process of identifying these criteria has led to a second problem.

The trust evaluation process is a detailed undertaking requiring a high processing overhead. This process can either result in a value that allows an

agent to trust another to a certain extent or in a distrust value that results in termination of the interaction. The evaluation process required to obtain the distrust value is just as process intensive as the one resulting in determining a level of trust and the constraints that will be placed on an interaction. This raises the question: How do we simplify the trust evaluation process for agents that have a high probability of resulting in a distrust value?

This research solves this problem by adding a fifth category to the criteria already identified; namely: prejudice filters. These filters have been identified by the literature study and are tested by means of a prototype implementation that uses a specific scenario in order to test two simulation case studies.

Keywords: trust model, architecture, prejudice filters, criteria, prototype, simulation, trust representation, initial trust, trust update, trust evaluation.

Contents

List of figures

List of tables

PART 1

INTRODUCTION AND BACKGROUND

1. Overview

1.1 Introduction

The world of e-commerce is a vast, dynamic domain that often requires 'virtual' businesses to communicate and establish contracts in an environment where changes in business customs can occur almost in an instant (Siyal & Barkat 2002). E-commerce has changed business scale from that of large businesses to that of smaller, faster and more disintermediated businesses, the components (Khare & Rifkin 1997) of which all need to communicate with one another within the e-servicescape (Britner 1992). Disintermediation refers to the removal of middlemen especially as a result of the Internet and e-commerce (Fujiwara 2000). The servicescape, which is the environment in which a service is granted (Britner & Zeithaml 2006), in this context refers to the delivery of services in an electronic environment. If one wishes to operate successfully in this environment, one needs to be able to define whom one can trust and, more specifically, how one trusts, in this environment.

The Internet consists of open systems across a broad range of administrative domains. Entities within these domains wish to communicate with one another in order to accomplish an exchange of transactions. In this context, entities include any devices that communicate with one another within a computerised network environment. This environment leaves such entities open to risk; hence, communication requires some form of trust management. According to Andrew Twigg and Nathan Dimmock (2003), the Internet's open systems have four main characteristics that cause these systems to need computational trust models. The four characteristics are peer-to-peer relationships, interaction among peers that have not previously interacted, multiple administrative domains and a lack of globally trusted recommendation agents.

Defining trust and trust relationships extends the concept of defining security. It is an attempt to define and control the security and nature of information that is shared during a transaction between two agents. An agent, in this context, refers to a computerised agent that has some form of trust mechanism in place. Trust models are required to define and maintain security within a dynamic environment that allows agents that have not previously interacted with one another to mutually participate in transactions. Trust models allow the participants in such interactions to

evaluate one another's trust values. This evaluation is then used to determine the level of trust given to a transaction.

A large amount of research has been conducted with regard to trust models. Various trust models have been defined, analysed and implemented. However, almost every model has its own unique approach, based on what the author of each model believes to be relevant. This study is conducted within the broad field of trust models. Section 1.2 introduces the problems that this research attempts to solve. Section 1.3 discusses the methods this research adapts in order to solve the problems identified. Important terminology used in this dissertation is defined in Section 1.4. Section 1.5 concludes Chapter 1 and provides an outline of this dissertation.

1.2 Problem statement

A number of issues have influenced this study. These include the growth of e-commerce, the rising interest in trust model formulation, the lack of criterion standardisation in trust models and the lack of filtering mechanisms possessed by trust models.

1.2.1 Growth of e-commerce

Traditionally, trust relationships were based on physical identities. However, since the advent of the Internet, this scenario has changed, as more and more businesses create 'virtual' faces for themselves by publishing Internet sites that can be used for what has now become known as e-commerce (Yang, Brown & Lewis 2001).

This is a direct result of the fact that the media have told businesses that the Internet and a Web-based presence would change the world of business. However, the initial rush to move business into the virtual realm was not as successful as many enterprises had hoped. In fact, the failure rate of businesses involved in e-commerce continues to rise (Panther, Erwin, & Remenyi 2003). In their investigation into this phenomenon Yang et al. (2001) point out a number of factors that have caused some e-commerce businesses to fail, such as the lack of effective business products, processes and services.

These problems can be investigated further as issues of trust. Trust is required between the business and the consumer, as well as between the business and its partners. Due to the virtual nature of the business, these trust relationships can no longer be formed in the traditional manner. New trust management mechanisms and new ways of establishing trust relationships are required (Yi, Corbitt & Thanasankit 2002). This need to redefine traditional trust relationships has led to the second issue influencing this study.

1.2.2 Rising interest in trust model formulation

The world of computing, especially the Internet, is a massively networked world that supports a huge population of diverse agents that lack central control. This has led to increased interest in managing this diverse environment, the composition of which is both dynamic and unpredictable. Due to the fact that agents are required to deal with dynamic interactions, including many that are unforeseen, a management system controlled by humans is not feasible (English et al. 2002). Hence, researchers have investigated the implementation of trust models that allow machines to take over trust decisions that would normally require human intervention.

The formation and implementation of such trust models seems like an ideal solution to the dilemma of allowing agents in a dynamic environment to interact with one another. However, trust has been found to be an extremely dynamic concept, the definition of which is often open to interpretation. Since trust is a dynamic concept with no fixed definition, it has resulted in a multitude of varying interpretations thereof. These varying interpretations have further resulted in varying trust model definitions, each supported by different interpretations of the concept of trust. This leads us to the first of two main problems that this research addresses.

1.2.3 The lack of criterion standardisation in trust models

So far, no absolute definition of the concept of trust exists, due to the very abstract nature of the concept. How it is defined is often influenced by a person's perspective, values and situation. Several authors have proposed varying trust models. These include the trust models developed by Manchala (2000), Xiong and Liu (2003), Esfandiari and Chandrasekharan (2001) and Guha et al. (2004), to name but a few. These and other models are investigated more fully in Chapter 2.

The examples referred to are valid trust models relying on various concepts of trust. Each author or set of authors took the subjective concept of trust and trust management, and quantified it in various ways to implement a particular trust model. This has resulted in various trust models, each relying on different quantified forms of trust, which brings us to the main problem: How does one evaluate these different trust models for implementation within a particular environment?

1.2.4 The lack of filtering mechanisms in trust models

Various trust models have been proposed in order to minimise the risk of sharing information and to maximise the chances of successful transactions (Abdul-Rahman & Hailes 1997; Ramchurn et al. 2003). Trust models rely on the abstract principle of trust in order to control what information is shared and with whom. Trust models evaluate the participants of a transaction and assign a numerical value, known as a trust value, to the interaction. This numerical value is used to determine the restrictions placed on the transaction.

Since this process of analysis occurs together with all the interactions an agent executing a trust model encounters, the process can be quite a lengthy one – depending on the trust model implementation in place. It may cause an overwhelming communication load if a trust model is required to make a full analysis of every interaction it encounters and will result in wasted processing every time that the outcome is distrust. This brings us to the second problem addressed by this research: What can be done to lessen the communication and processing load placed on trust models when it is possible to predict in advance that the result of a particular evaluation is likely to be distrust?

1.3 Methodology

The study's main approach towards obtaining solutions involved a literature survey. The survey explored existing trust models in detail in an attempt to converge the various ideas – each with their own merits – into a single cohesive whole. Various trust models were examined in detail in order to identify commonalities between and highlight key concepts from each.

Once the literature survey was concluded, the study shifted towards a more practical approach. The solutions suggested by the literature survey were applied to practical examples of trust models to test their viability. This application was extended to include a prototype simulation (and an analysis thereof) in order to identify the improvements achieved by the suggested solutions.

1.4 Terminology used in this dissertation

A number of terms are used frequently in this dissertation. Because some words can have multiple meanings and some definitions may be considered vague, a number of key terms used in this study are defined below.

1.4.1 Agent

The term ‘agent’ is a broad term that is often applied to various ideas. A well-known application of this term in the field of Computer Science is using the term for artificial intelligence (AI). In such a context, an ‘agent’ uses logical formulae to simulate human intelligence (Nilsson 1998). In the context of this study, the concept of an ‘agent’ refers to a computer or any other device that executes and implements a trust model by relying on the abstract logic contained therein in order to make trust-based decisions.

1.4.2 Trust

The concept of trust has been explored in various contexts, including psychology, philosophy and sociology (Langheinrich 2003). It has attracted attention from several researchers including those trying to establish trust within dynamic e-commerce environments. In the context of this study, the concept of trust concentrates on the formation of trusting relationships between any two agents within a computerised environment. This requires each agent to have access to so-called trust information that allows each agent to establish a trust value.

Witkowski, Artikis and Pitt (2000) paraphrase the definition of trust as follows: ‘Trust is the assessment by which one individual, A, expects that another individual, B, will perform (or not perform) a given action on which its (A’s) welfare depends, but over which it has restricted control.’

1.4.3 Distrust

The concept of trust cannot discussed in it's entirely without considering the concept of distrust. Distrust is defined by Grandison and Sloman (2000) as the direct opposite of trust. In particular they declare that it is the lack of belief in an agent's competence and reliability in a given situation. This definition is, however, not entirely correct. Lack of trust does not necessarily constitute distrust simply due to the fact that a lack of trust does necessarily indicate bad behaviour but can be indicative of a lack of information. A more accurate definition and one that is adopted by this research is the one provided by Ray and Chakraborty (2004:263) which define distrust as 'the firm belief in the incompetence of an entity to act dependably, securely and reliably within a specific context.' This definition requires that an agent possess some kind of knowledge and possibly experience with another agent in order to establish distrust.

1.4.4 Trust models

For the purpose of this research, trust models refer to the coded implementations of trust concepts. These models allow a computer to emulate human trust and make decisions based on these emulations (Khare & Rifkin 1997; Guha et al. 2004; Lamsel 2001; Datta, Hauswirth & Aberer 2003; Patton & Jøsang 2004; Papadopoulou et al. 2001; Langheinrich 2003). In this study, trust models refer to the coded entities that emulate human trust.

1.4.5 Prejudice

Prejudice can be defined as a negative attitude towards an agent, based on a stereotype. This attitude places all entities of a certain stereotyped group in the same category. Such stereotypical belief is defined and coloured by the culture or environment from which it stems (Bagley et al. 1979). In this study, prejudice refers to a negative attitude that one agent has towards another during the trust analysis process.

1.4.6 Context

Context refers to the circumstances or environment in which an event occurs and, hence, the circumstances or environment that surrounds and gives meaning to the event (Schmidt, Beigl & Gellersen 1999). For the purposes of this study, the event that occurs is an interaction between two agents. The context is an important consideration, as it influences the trust that a trust model

assigns to an interaction. For instance, a hacker's presence in a system requires the system to assign lower trust values to machines that are suspected to have been compromised.

1.5 Outline of this dissertation

As can be seen by the index given at the beginning of this dissertation, the document consists of six parts, divided further into a total of thirteen chapters.
Part 1 contains the introduction and background to this research. The current chapter, *Chapter 1*, provides an introduction to the research problem under investigation. The rationale for the study and issues under investigation are introduced and the problem area is defined. *Chapter 2* provides an overview of the current research conducted in this area of interest. It also introduces a basic trust model architecture designed by the author in order to assist in defining the commonality between various trust models.

Trust models have been identified as wide and varied, with no current clear means of analysing their worth. *Part 2* of this dissertation addresses this issue by introducing and discussing trust model criteria based on current trust model implementations. *Chapter 3* identifies four main categories into which these trust model criteria fall and discusses the first, trust representation, in detail. *Chapter 4* expands on the work done in *Chapter 3* and looks at two categories that share trust model criteria: initial trust and trust update. *Chapter 5* concludes the discussion on the four categories of trust model criteria identified in *Chapter 3* by discussing the last of the four categories, namely trust evaluation.

An extension of the set of trust model criteria categories identified in *Part 2* is explored in *Part 3* of this dissertation. *Chapter 6* introduces and discusses an additional category to the trust model criteria discussed in *Chapters 3*, *4* and *5*, namely prejudice filters. Prejudice filters are proposed in order to filter out agents that have a high probability of resulting in a distrust value, before the detailed trust analysis process occurs. *Chapter 7* investigates the relationships that exist between the various prejudice filters. These relationships allow various prejudice filters to be implemented together.

The trust model criteria categories discussed in *Part 2* and extended in *Part 3* are put to the test in *Part 4*, where current trust models that have already been defined are analysed using the identified trust model criteria. *Chapter 8* contains a discussion of a detailed analysis of a single trust model using all five categories defined. Partial analysis in order to test the flexibility of the trust model criteria on a few more trust models is reported in *Chapter 9*.

The concept of implementing prejudice in trust model architecture is explored in more detail in *Part 5* of this dissertation. *Chapter 10* provides a structural and implementation description of a prototype used to test the impact of prejudice filters on trust. The scenario used by the prototype in order to execute the test simulations is given in *Chapter 11*. The results of the simulations are graphically depicted and discussed in *Chapter 12*.

Part 6 is the final part of this dissertation and consists of *Chapter* 13, the references and appendices. The chapter provides a summary of the key concepts of the research that was undertaken. The dissertation concludes with an evaluation of the extent to which the problem has been successfully solved. Areas that need further investigation are reflected upon. Finally, the references are provided, followed by the appendices.

2. Trust and trust models

2.1 Introduction

The advent of the Internet and accompanying technologies has instigated a change in business perspective, driving businesses to create a virtual presence in order to expand their target markets. This has simultaneously influenced the basic principles required to run such a business (Hultkrantz & Lumsden 2001). Businesses have become increasingly information-driven, and they seek to use information as an asset to achieve a competitive edge and assist in decision-making processes. It is therefore important that information used by the business is reliable and accurate so that it can be trusted (Patton & Jøsang 2004). Sharing information carries risk, because it is possible for other parties to tamper with or misuse information. This means that it is also important for a business to determine whether or not the recipient of any information is trustworthy.

Several different trust models have been proposed in order to minimise the risk of sharing and successfully analysing information (Jonker & Treur 1999; Perlman 1999; Linn 2000; Jøsang 1997; Lamsal 2001; Stoneburner 2001; Xiong & Liu 2003). Trust models rely on the abstract principle of trust in order to control what information is shared and with whom. Thus, in order to gain a comprehensive understanding of how trust models work, one first needs to gain insight into the concept of trust itself. The concept of trust is discussed in Section 2.2. Section 2.3 subsequently looks at some of the trust models proposed and how they implement the concept of trust. The various trust models proposed are next consolidated to create a basic trust model architecture that generalises the way in which trust models are implemented. This basic trust model architecture is dealt with in Section 2.4, while the discussions in Chapter 2 are concluded in Section 2.5.

2.2 Trust

Trust is a subjective concept, and the perception of trust is unique to each individual. In any single situation, different individuals have different perceptions and thus make different trust

assumptions and reach different conclusions. Consequently, several definitions of the concept exist.

Even though there are several definitions of trust, there is general consensus about the need for trust. Trust is ultimately required in order to reduce complexity. This complexity refers to the complexity created by large complex environments, the complex intentions of (other) individuals and complexity created through exposure to unfamiliar situations (Deutsch 1962; Luhmann 1979). Trust allows for certain assumptions to be made and hence simplifies the information that an individual is required to analyse.

In order to obtain a definition of trust relevant to this study, ideas and concepts from various experts have been investigated and merged. In particular, this section explores the formal definitions given by Barber (1983), Nooteboom (2002) and Gambetta (1990a), after which they are merged in order to obtain a working definition for this study.

2.2.1 Barber's definition of trust

Barber (1983) defines trust as an expectation of the degree to which another entity's behaviour results in favourable results for the trusting entity. The expectations are based on the social systems and relational expectations of the environment in which the trust exists. Thus, trust is influenced by environment, state and situation.

According to Barber (1983), there are two basic types of trust: trust determined by logical analysis and trust determined by moral assumptions. The first, determined by logical analysis, is basically trust in the technical competence of a particular individual. This type of trust is often determined by the role which an individual plays in society. For instance, members of the police are trusted to act honourably and in the best interests of the citizens that they serve.

The second type of trust is more emotional and refers to trust given in the expectation that the trust will be upheld and that certain responsibilities will be fulfilled. This trust is more intuitive than the first, and it requires a larger risk to be taken in the hope that the trust will not be broken, simply because the trust itself exists. For instance, you may trust your friend to look after your

home while you are on holiday, simply because the person is your friend and you have faith in the person's morals. It is trust given based on certain moral assumptions that are made.

2.2.2 Nooteboom's definition of trust

Nooteboom (2002:8) defines trust as a four-place predicate, stating that '**Someone** has trust in **something**, in **some respect** [competence, intentions, benevolence (Nooteboom 2002:51)] and under **some conditions**' (Nooteboom's emphases). Each of the four key concepts highlighted by Nooteboom is included in most trust model architectures. *Someone* and *something* define two agents participating in an interaction. The former refers to the instigator of the interaction, while the latter refers to the agent accepting the request. The *respect* is defined by the reason for instigating an interaction. Finally, the *conditions* refer to the situational factors that influence the success of an interaction.

2.2.3 Gambetta's definition of trust

Gambetta (1990b) approaches trust in a more technical manner than either Barber (1983) or Nooteboom (2002), formalising trust as a range of values. Although this approach is rather ambiguous in that the same value can be interpreted differently by different individuals, it provides scientists with a means by which to mathematically formalise what is inherently an abstract intangible concept.

Gambetta (1990a:217) formally defines trust as follows: 'trust (or symmetrically distrust) is a particular level of the subjective probability with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such action (or independently of his capacity ever to be able to monitor it) and in a context in which it affects his own action.'

2.2.4 Definition of trust with regard to this study

Each of the concepts defined by Barber (1983), Nooteboom (2002) and Gambetta (1990a) has an impact on the definition of trust used by this study. These various definitions were taken and merged in order to obtain the following definition for trust: 'Trust can be defined as the

probability that another agent's action, when that agent is performing a requested task, under given conditions, will benefit the agent that requested that the particular task be performed on its behalf.' This definition can also be extended to include the trust an agent performing a given task has in the benevolence of the agent it is performing a task for. Trust is bidirectional, unique to each agent and has an effect on all parties involved in the trust relationship. It is important to note that this definition includes the parties involved in the trusting relationship, the context in which the trusting relationship exists and some form of risk.

2.2.5 Risk

Risk is an inherent part of trust. If some degree of uncertainty does not exist, there is no need to trust, as the transaction then becomes a certainty. Thus, trust involves risk (Bohnet & Zechhauser 2004). This view is further confirmed by experts such as Spekman and Davis (2004) and Stahl and Sitkin (2005) who claim that when a transaction carries no risk, trust itself need not exist.

Trust entails acceptance of the fact that one is not in control of the behaviour or intentions of others and thus need to rely on something or someone for successful interaction to occur. The fact that one is not in control of others' behaviour implies that others may choose to behave in a manner that is in contradiction to the trust that has been given. Therefore, by trusting another, one is exposed to risk.

2.2.5.1 Definition of risk

Formally, risk can be seen as the possibility of loss that participating in a particular interaction could incur. The larger the possible loss an agent may encounter, the bigger the risk inherent in the interaction. Interactions that carry high risks consequently require high trust values.

2.3 Trust models

Due to a large interest in trust development, a number of trust models have been formulated and proposed. Trust models are coded implementations that rely on the concept of trust in order to assign a trust value for an interaction. This trust value is then used to control and restrict the interaction. Different trust models have been proposed by various experts. The proposed trust

models approach the issue of establishing trust in various ways. These include, but are not limited to, recommendation, reputation, observation, institution and experience (Jøsang 1997; Xiong & Liu 2003; Esfandiari & Chandresekharan 2001; Papadopoulou et al. 2001; Abdul-Rahman & Hailes 1997). In order to change trust over time, trust models also update trust. The means by which trust is updated usually relies on the same concepts used to initialise it. The trust values a trust model possesses are evaluated before an agent participates in an interaction. If the trust value is favourable, the agent participates in an interaction. If not, the interaction is cut off and denied. A few trust models and the concepts on which they rely are discussed in the subsections that follow.

2.3.1 A business-orientated trust model

Daniel Manchala (2000) takes a business-oriented approach to trust, reasoning that such an approach is required by e-commerce. He suggests a set of trust parameters for risk management that centres on business concepts. Manchala groups these parameters into four broad categories according to their characteristics, which include transaction cost, transaction history, indemnity (intermediary reference), and other variables.

Transaction cost variables are features that make a single transaction costly. These features can involve either a single expensive purchase or a large number of small, cheaper purchases that are costly when added together. Transaction history variables illustrate the success of previous transactions an agent has participated in. Indemnity allows for a guarantee against loss and it is mainly required for those participants in an interaction that do not yet possess a transaction history. The other variables identified by Manchala (2000) include spending patterns and system usage.

These factors greatly influence the level of trust given to an agent. Transaction cost introduces the concept of risk. The higher the transaction cost, the higher the risk involved. A transaction history allows a business to form and analyse transactional patterns. Agents with transaction histories of several successful transactions are more likely to develop trust relationships than those with transactions that have failed. Indemnity allows a trusted third party to furnish a reference for those agents that do not have a transaction history for the trust analysis purposes.

Finally, spending patterns and system usage are closely linked to transaction history, and changes in these patterns should be regarded with suspicion. These trust model criteria can be fine-tuned by making use of time and location. The two values address the way(s) in which time and context (location) influence trust. The number of transactions conducted during a certain period reflects transaction frequency and can also be used to determine change of state. Location investigates the environment in which agents reside. However, Manchala's (2000) parameters evaluate only the influence of time and location on trust and neglect the impact that the situation has on trust.

2.3.2 A reputation-based model

Indirect methods of creating and evaluating trust lower this risk by supplying at least some information that an agent can use in order to make better informed decisions. Xiong and Liu (2003) focus on indirect methods of defining trust and have defined a reputation-based model known as PeerTrust. In order to establish trust with unknown agents, this model evaluates a peer's trust by evaluating a peer's level of reputation. This is accomplished by evaluating five key parameters, namely satisfaction feedback, the scope of the feedback (which includes transaction numbers), feedback credibility, transaction context and community context.

Satisfaction feedback is given in terms of the amount of satisfaction other peers have had from transactions with a particular peer. An agent analyses the trust information it receives from peers and uses this information to create a trust value for the unknown agent. The transaction number can be used by a trust model to evaluate the percentage of failures occurring when interacting with a particular agent. This has the added advantage of preventing agents from covering up failures by simply increasing transaction activity.

To prevent false reputation statements from hindering the trust-building process, a credibility feedback mechanism is included. Feedback from those with better credibility is weighted more heavily in the trust-building process. Due to the situational nature of trust, context becomes an important factor to consider. Two types of contexts are defined by Xiong and Liu (2003), namely transactional and community context. The transactional context refers to the context in which a transaction takes place and includes the size of a transaction, as well as the roles required by the

participants in a transaction. The community context takes into consideration the community to which agents belong, as well as the community in which a transaction takes place.

In essence, Xiong and Liu's model attempts to use social control to build trust. It is a community-based model that removes the involvement of third parties. This model is very strongly based on a community feedback principle, whereby positive or negative feedback is propagated to peers within the network during the outcome evaluation phase, and the feedback is used both to initialise and to update trust.

2.3.3 A recommender trust model

Trust models allowing agents to share trust values with other agents have been proposed by some experts (Guha et al. 2004; Datta et al. 2003; Abdul-Rahman & Hailes 1997; Manchala 2000). These models rely on the assumption that trust can be propagated or shared, or is transitive in some way. Assuming that trust is transitive allows an agent to trust an unknown agent simply because the first agent receives a recommendation from another agent. For, instance, if A trusts B, and B trusts C, the principle of transitivity assumes that A should therefore also trust C. However, trust is not always transitive. It is possible that C has something against A that would cause the transaction to fail (Jøsang 1997). Therefore, in order for transitivity to hold, certain conditions need to be met.

The term 'conditional transitivity' was coined by Abdul-Rahman and Hailes (1997) for this type of trust. The conditions that they identified are the following: the recommender agent must explicitly identify a recommendation as such; recommender trust must exist between two agents (the agent receiving the recommendation needs to trust the agent giving the recommendation as a recommender); an agent must be permitted to make its own judgements about recommendations and check the recommendations against its own rules and judgements and, finally, trust may not be absolute.

In keeping with the identified ways of propagating trust and the issues associated with them, Abdul-Rahman and Hailes (1997) propose a decentralised approach to generalise the concept of trust and to define a protocol to exchange trust-related information. They define explicit trust

statements in order to reduce ambiguity and propose a recommendation protocol to propagate trust in such an environment.

Abdul-Rahman and Hailes (1997) define the following three properties of trust: trust is always between two entities; trust is unidirectional, and trust is conditionally transitive. Mutual trust is indicated by two separate relationships, allowing independent manipulation of each. Each agent has its own defined trust relationships with other agents that may differ from other agents' trust relationships and, therefore, each agent is responsible for its own trust definitions. Furthermore, two distinct categories of trust are defined: direct and recommender trust. This allows an agent to distinguish between the trust it accords a particular agent due to direct interactions, evaluations and experiences, and the trust an agent accords this same agent as a recommender of other trust relationships. This model explicitly separates the definitions of trust gained by direct means from those of trust gained by indirect means, and it emphasises the fact that the perceptions of agents may differ.

2.3.4 A trust model relying on acquisition and negotiation

Esfandiari and Chandrasekharan (2001) propose a trust model that emphasises the process of trust acquisition and negotiation. They define trust acquisition as 'the process or mechanism that allows the calculation and update of T', where T is a trust function between two agents, and also identify three ways in which agents can define trust: observation, interaction and institution.

Trust acquisition via observation is accomplished by observing other agents' past experiences, acquiring a trust value based on these observations and comparing it to the trust values obtained through one's own observations. In this way, an agent is able to ascertain and act upon patterns of consistent behaviour. Allowing agents to interact and collaborate thus allows them to build trust through interaction. Esfandiari and Chandrasekharan (2001) suggest two protocols that assist in defining trust through interaction. The first is the exploratory protocol that is used to look for and find other trustworthy agents in the network. This protocol requires an agent to ask questions to which the agent already knows the answer(s) and then increases the trust ratings of agents that supply the expected answer. The second protocol – the query protocol – is used once trusted

agents have already been defined. It is used by an agent to ask for advice from the defined trusted agents.

Institutional trust is based on the concept of reputation. Reputation-based trust is acquired through the process of aggregating trust situations and sharing the trust values resulting from these aggregations. Once a community shares these values, they become a limiting standard that defines a reputation, which is then used to define trust. Institutional trust allows the complexity of trust to be split into that of trust associated with the agent and trust associated with the environment, allowing trust-building processes to be simplified. Assumptions about the environment can be made based on institutional trust, leaving only a few other trust parameters to be analysed.

Once a trust relationship has been established, this relationship can be propagated to other agents to facilitate global trust knowledge. This process allows for the formation of trust based on reputation. As it is assumed that trust is weakly transitive, the model proposed by Esfandiari and Chandrasekharan incorporates a decrease in the trust value, the further it is propagated. Since this method of trust propagation can result in conflicting trust values from different paths, propagated trust values are used as trust 'intervals' containing a maximum and minimum value. In pessimistic situations, the minimum value is used, while in optimistic situations the agent takes the maximum value. In order to prevent long paths from distorting a trust value, this model limits the path length of trust propagation.

2.3.5 Including distrust into a trust model

According to Guha et al. (2004) there is a problem with only having values for trust within a trust model. This leads to the interesting problem of differentiating between low trust levels as a result of inadequate information and low trust levels as a result of misbehaviour. In order to address this issue, they recommend the inclusion of some way of representing distrust as a separate entity. This is accomplished by including a matrix for both trust and distrust. These matrices are propagated repeatedly through a process known as a 'matrix-powering operation'.

Guha et al. include a belief matrix into their trust/distrust model to store the beliefs an agent holds about the world in which the agent resides. This belief matrix is calculated by using combinations of the trust and distrust matrices, depending on the factoring model chosen. Guha et al. also look at three possible factoring models, namely the trust-only factoring model, the one-step distrust factoring model and the propagated distrust factoring model. In the trust-only factoring model, only the trust matrices are propagated. Accordingly, the belief matrix becomes a result of only the trust matrices received. The one-step distrust factoring model assumes that when a user distrusts somebody, the user discounts all judgements made by that person. Based on this assumption, distrust is propagated in a single step, while trust is propagated repeatedly. The propagation of both trust and distrust together is known as propagated distrust. The trust and distrust can thus be treated as two ends of a continuum and considered together to reach a trust value.

Transitivity of trust does not carry over to distrust. It is possible for instance for A to distrust B, for B to distrust C, and for A to trust C. This is possible since A may distrust B's judgement and therefore also the fact that B distrusts C. This conclusion is based on the principle that your enemy's enemy is your friend. Including distrust as a separate parameter allows for more accurate trust analyses to be made, but it complicates the analysis process. The approach adopted by Guha et al. is flexible in respect of the way in which distrust is handled. Distrust can either be ignored, or propagated only to neighbours, or propagated along with trust to all agents.

2.3.6 A binary trust model

Public Key Infrastructure (PKI) makes use of a process of certification and authentication in order to implement a form of binary trust (Jøsang et al. 2000). Trust is established by evaluating credentials that are provided. This evaluation process yields one of two results, either complete trust or complete distrust (Aberer & Despotovic 2001).

PKI relies on pairs of public and private keys in order to evaluate the trust (Jøsang et al. 2000). When an agent receives a message the message is signed with the sender's private key. The agent then looks up the sending agent's public key and uses the public key in order to decrypt the communication. If the decryption is successful it means that the key is trusted and consequently so is the content of the communication. This approach to trust introduces the complexity of

whether a given public key can be trusted. In order to identify whom a public key belongs to Certification Authorities (CA's) have been put into place. CA's provide a means of identifying who a public key belongs to by supplying signed public keys that are grouped with the owner's identity. CA's allow agents to identify which public keys are considered to be trustworthy by allowing an agent to trust all public keys from a particular trusted CA.

2.3.7 Summary

From the discussion of the different trust models above, it is clear that there are varying ideas on how trust should be handled. Both indirect and direct approaches towards trust acquisition (trust establishment) are adopted. Indirect means of establishing trust that have been proposed include propagation, recommendation, observation, reputation, negotiation and institution. Direct means include interaction and risk management. Just as the experts quoted above consider a variety of approaches towards trust establishment, they also consider various parameters to be vital to the trust-building process. These parameters include business concepts such as transaction cost, transaction history, indemnity, time, location, satisfaction feedback, feedback credibility, transaction context, community context, recommenders, patterns of behaviour and even distrust parameters.

All of these ideas and parameters are well-validated and thought-out concepts related to trust. However, each idea or parameter affects the trust evaluation process in its own manner and will result in different trust evaluations when taken into consideration. When choosing a trust model to implement, the manner in which it establishes trust and the parameters that it applies become important concepts to consider. Different environments each have their own unique needs, which include the manner in which trust is expected to behave in a particular environment. Consequently, when choosing a trust model to be implemented in a particular environment, one needs to consider the impact of the varying ideas and parameters on the final trust evaluation result.

Different environments require uniquely different behaviour from the trust model they wish to implement. The manner in which a trust model has been defined has an impact on its behaviour. In order to determine this impact, certain criteria are required against which trust models can be

evaluated. This research proposes such a set of criteria. In order to successfully propose these criteria, the various trust models discussed in this chapter are considered and evaluated, as well as some additional ones, their varying ideas and parameters and the impact of each.

2.4 Basic trust model architecture

The various trust models discussed have unique features identified by their various authors. However, when one takes a closer look, a common structure related to the interactions in which trust models participate can be identified. According to Ramchurn, Jennings, Sierra and Godo (2004), basic interactions go through three main phases, namely negotiation, execution and outcome evaluation. Since trust plays an essential part in all of these phases, they can be used to illustrate a basic structure found among most trust models.

Figure 2.1 provides a basic overview of the three phases that interactions go through, and the role of trust in each. Each phase is illustrated with a block around the trust processes that belong to the specific phase. The first two blocks indicating the negotiation and execution phases encompass both agents, as they require information to be shared between the two agents. The third phase is illustrated with two smaller blocks that surround only a single agent, because this phase occurs individually on each agent's side.

Two agents attempting to communicate with one another are first required to establish a communication link, usually initiated by one agent and accepted by another. This process initiates the negotiation phase whereby two agents negotiate various parameters, such as the security level of information that is to be shared or the services for which permission will be granted, and these parameters subsequently define the boundaries of the interaction between the agents concerned.

During the negotiation phase, a trust value for the interaction is defined through comprehensive analysis of logical rules. The simplest way of storing and implementing these rules is to have them present in a list that the agents access and process. In Figure 2.1, the storage of these rules is depicted as occurring in the trust definition list.

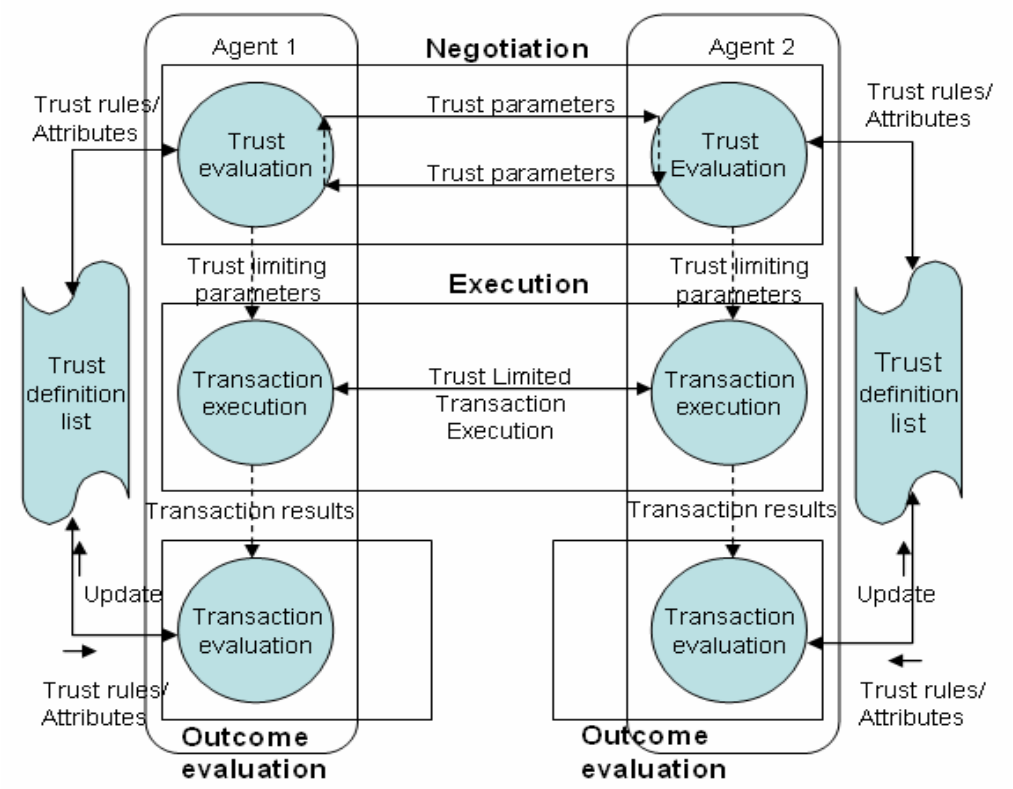

Figure 2.1 Operation of an agent using a trust model

The successful negotiation and establishment of a trust value triggers an analysis of the trust value. Provided the trust value is above a certain acceptable threshold, the execution phase commences. This phase contains the transaction execution process. Trust models place limitations on interactions during the execution phase so as to control the information and services supplied. These limitations are often enforced by the context of an interaction. Contexts may vary from the environmental factors of an agent to the nature of the transaction itself. For instance, travel agents may instigate transactions in the contexts of bookings, purchases and gathering information. These contexts are transaction-related. On the other hand, a trust model for the same travel agent may choose to classify the context according to the airlines that agents belong to, shifting the contextual factor from transaction-related to environment-related issues.

Once the execution phase has terminated, the results of the interaction are sent to the outcome evaluation phase. This phase contains the transaction evaluation process that evaluates the results and updates the trust definition list in either a positive or negative manner. Negative updating of

the logical rules occurs due to business transaction failure (or a bad experience), while business transaction success (or a good experience) will trigger a positive update.

2.4.1 Analysis of the trust models discussed using the basic trust model architecture

The trust models discussed in this chapter implement the three phases identified in Figure 2.1 in various ways. The negotiation phase is basically the phase in which trust models either initialise a trust value or evaluate trust for another agent. The four categories of business-oriented trust parameters identified by Manchala (2000) are analysed in this phase. Xiong and Liu (2003) rely on an indirect method to negotiate trust and use an agent's reputation to initialise trust. This reputation is a result of feedback and context and defines the trust level for another agent. Abdul-Rahman and Hailes (1997) also use an indirect method to initialise trust and rely on recommendation and the propagation of recommendation. Esfandiari and Chandrasekharan (2001) rely on a trust function, observation, collaboration, institutional reputation and an exploratory protocol in order to negotiate and initialise trust for an interaction. Guha et al. (2004) use factoring models that analyse trust and distrust parameters in order to obtain a trust value. PKI (Jøsang et al. 2000) evaluate credentials that are used to establish either complete trust or complete distrust. These matrices are used for both trust initialisation and trust evaluation.

The execution phase uses the results of the negotiation phase to limit the interaction an agent participates in. In the case of the trust models discussed, execution is limited by the trust value that the trust model obtains from the negotiation phase. Manchala's (2000) model furthermore limits an interaction according to the indemnity it has and the location of the agents participating in the interaction. Xiong and Liu's (2003) trust model has additional limits for an interaction and allows the transaction context to limit the interaction.

Once the execution phase has been concluded, trust must be updated for future reference. Most of the trust models discussed here use the same principles that were used in the negotiation phase to update the trust values in the outcome evaluation phase. Three of the business-orientated trust parameters identified by Manchala (2000) are important in this phase: the transaction history, spending patterns and system usage. These parameters keep track of changes in the system. Xiong and Liu (2003) rely on the principle of community feedback to propagate changes in trust

throughout the system. Abdul-Rahman and Hailes (1997) also rely on propagation, but, in addition, they include experience evaluation in the outcome evaluation process. Esfandiari and Chandrasekharan (2001) use the same trust function that they use in the negotiation phase in the outcome evaluation phase, together with a query protocol. Finally, Guha et al. (2004) also use the same methods in the outcome evaluation phase as in the negotiation phase.

One of the trust model's discussed, however, does not update its trust value which is a major identified flaw in the approach. PKI does not change trust level based on experience, once trust or distrust has been established the state remains regardless of changing behaviour. This brings us to a great limitation of a binary approach. A binary approach limits the choices that can be made and has no clear definition in the face of incomplete and inconclusive information (Chakraborty & Ray 2007). The simple yes, no approach to trust also fails to address the situational nature of trust and does not allow agents to change degree of trust in face of a changing dynamic trust environment.

2.5 Conclusion

This chapter has introduced and looked at several concepts vital to the dissertation. The core concept covered is that of trust and trust models. This chapter explored the meaning of trust between humans and linked this to trust model architectures that have been proposed by various researchers in the field. Various important concepts related to the concept of trust have been identified and discussed. These include risk, the situational nature of trust and trust perception.

An overview was given of trust models, after which a few chosen trust models were discussed, with reference to the concepts that make them unique. In Figure 2.1 a basic trust model architecture was illustrated, based on the three phases that basic interactions go through as identified by Ramchurn, Jennings, Sierra and Godo (2004). These three basic phases are negotiation, execution and outcome evaluation. The three phases refer to the way in which trust models initialise, update and evaluate trust, as well as the means by which trust models limit an interaction. Each of the models discussed in this chapter was then analysed in order to identify the ways in which it conforms to basic trust model architecture.

The identified trust models proposed so far cover a wide range of concepts and points. No standard set of criteria has therefore been defined so far, which makes it difficult to evaluate the different trust models. Chapters 3, 4 and 5 introduce and explore a set of criteria to solve this problem.

PART 2

TRUST MODEL CRITERIA

3. Trust model criteria: trust representation

3.1 Introduction

Trust models that rely on the human sociological and psychological concept of trust have been proposed by various experts. These models are varied, and each model focuses on different aspects of trust as a concept. Therefore, the manner in which the trust models handle trust varies and can result in different trust evaluations under the same conditions. Different environments will desire trust evaluation results that best apply to their circumstances. Consequently, they will require a trust model that implements concepts important to the particular environment. The question arises: how do they analyse current trust models in order to identify which one best suits the environment?

Purser (2001) proposes a graphical tool that can be used to identify the trust requirements of a particular environment. This tool guides the user through identification of properties such as context, associated confidence level, risk and transitivity of trust. These factors influence the type of trust model that would best be implemented in a given environment. Analysis of the environment, however, is beyond the scope of this research which focuses on evaluation of the trust models. Purser's model, can however be used in conjunction with the criteria proposed in this research in order to identify the trust model that best suits a particular environment where Purser's model is used to analyse the environment and the proposed solution is used to analyse trust models.

This chapter takes the first steps toward addressing the issue of identifying a trust model best suited to a particular environment by introducing a set of trust model criteria that can be used to evaluate a given trust model. The criteria are based on current trust model implementations and identify advantages and disadvantages of particular approaches. Section 3.2 gives a general overview of the trust model criteria and the categories the trust model criteria have been divided into, as defined in the current study. Furthermore, Chapter 3 looks at the first of these categories in detail (see Section 3.3). An overview of the first category and its criteria is given in Section 3.4, while the chapter's conclusions appear in Section 3.5.

3.2 Criteria for the evaluation of trust models

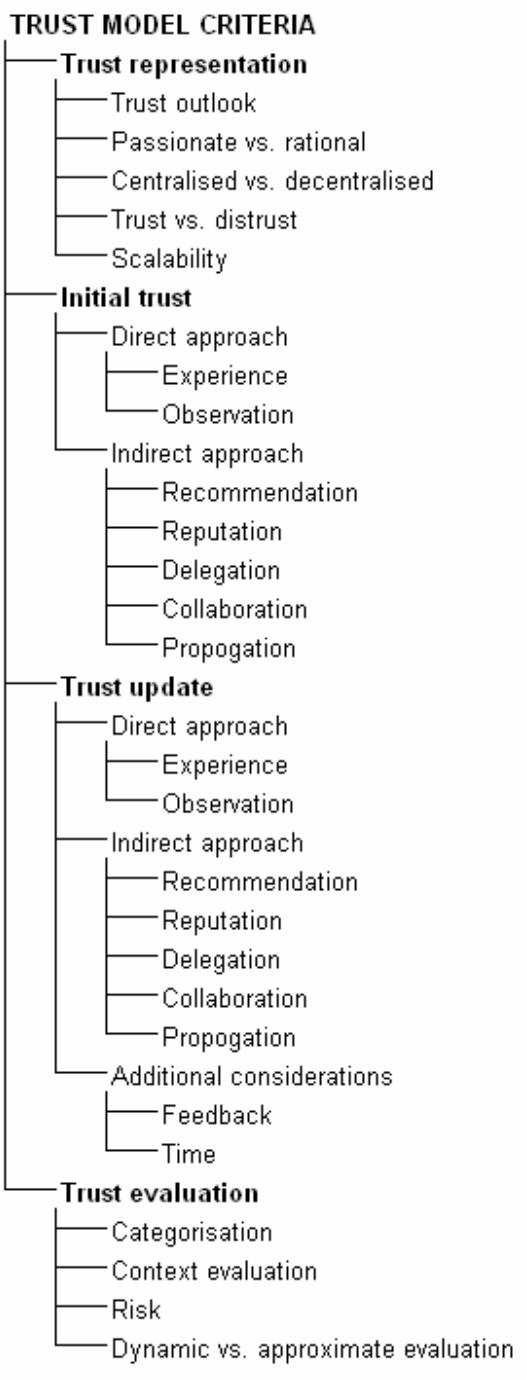

Figure 3.1: Overview of trust model criteria

Several trust models were investigated in order to identify functional consistencies. An analysis of these trust models resulted in the identification of four main concepts that appear in trust model implementations. These four main concepts were then taken and used to create the four categories of criteria identified by this research, namely trust representation, initial trust, updating trust and trust evaluation. Each of these categories has an associated set of criteria that defines various implementation-specific aspects of trust models. An overview of the four categories and their associated trust model criteria is given in Figure 3.1.

The trust model criteria define various properties of trust models and the ways in which these properties ultimately influence the execution of a particular trust model. Figure 3.1 summarises the trust model criteria identified in this study. The four categories are illustrated in bold with their associated criteria beneath them. The first category is discussed in the section that follows.

3.3 Trust representation

Trust representation refers to the way in which trust is represented by a specific trust model. This is an important concept, as it subsequently influences the way in which trust is processed and manipulated. The trust representation criteria are defined from a holistic viewpoint focusing more on general concepts than implementation-specific ones. The specific criteria identified with regard to trust representation are trust outlook, passionate versus rational trust, centralised versus decentralised trust, trust versus distrust and trust scalability. These criteria summarise the various ways in which trust can be represented. Trust outlook defines a trust model's approach to trust. Passionate and rational agents each approach trust representation in different ways. Centralised and decentralised trust defines the environment in which the trust can exist. Trust versus distrust looks at reasons either to represent only trust or to include explicit definitions for distrust.

Finally, scalability is used to determine whether a trust model is scalable, in other words whether it can be implemented in both large and small environments. Each criterion is discussed in detail, including the viewpoints of various experts, after which the author gives a collective definition for each criterion with regard to this study, as well as a short summary of the advantages and disadvantages of the specific criterion.

3.3.1 Trust outlook

When an agent tries to represent trust, it must have a trust outlook or a disposition to trust. An agent's disposition to trust influences the constructs used to represent such trust. Li, Valacich and Hess (2004) define trust disposition as the extent to which one agent is willing to depend on another in a given context. They also defined faith and stance as two constructs that encourage a positive trust disposition. Faith assumes that the other party means well and is dependable. Stance assumes that if one deals fairly with the other party, that party will deal fairly in return. However, this definition only looks at trust from an optimistic point of view. It allows the trusting party to take a trusting approach, which is not always feasible, as the two constructs do not always hold true.

Marsh (1994) extends the concept of an agent's trust disposition to include a pessimistic and a pragmatic approach, as well as the optimistic approach proposed by Li, Valacich and Hess (2004). An optimistic agent expects good outcomes, while a pessimistic agent expects the worst. A pragmatic agent operates somewhere in a spectrum between the pessimistic agent and the optimistic agent, balancing positive and negative experiences. Marsh's definition describes the spectrum into which agent dispositions to trust may fall. Although it is a good definition, Marsh fails to define the various ways in which the three defined agent dispositions can actually be implemented.

Jonker and Treur (1999) define the following six approaches towards trust outlook:

- *Blindly positive:* A blindly positive agent starts from a low trust value and looks for positive experiences with other agents. Once it reaches unconditional trust, it remains there indefinitely. Only positive experiences are evaluated and used to increase the trust value. Negative experiences are ignored.
- *Blindly negative:* A blindly negative agent is the mirror case of the blindly positive one, starting from a high trust value for other agents and dropping the trust value as a result of negative experiences. Once it reaches unconditional distrust it remains there indefinitely. Only negative experiences are evaluated and used to decrease the trust value. Positive experiences are ignored.

- *Slow positive, fast negative:* An agent uses several positive experiences to build trust, but can lose trust very easily, based on only a few negative trust experiences.
- *Slow negative, fast positive:* Trust is built based on only a few positive experiences, but several negative experiences are required in order for trust to be lost.
- *Balanced slow:* This approach allows for a slow update of trust in both the positive and negative domains. Trust is increased or decreased in small increments after thorough evaluation.
- *Balanced fast:* This approach allows for a fast update of trust in both the positive and negative domains. Any change in trust drastically influences the trust level in both a negative and a positive direction.

These six approaches can be grouped under the three main trust outlook approaches identified by Marsh (1994). The blindly positive and the slow negative, fast positive approaches are optimistic, while the blindly negative and slow positive, fast negative are pessimistic approaches. The balanced approaches are pragmatic. Since Jonker and Treur's (1999) approaches cover not only the spectrum of possible agent dispositions, but also the way in which they may be implemented and influenced (for the purposes of the current study), these six approaches are used to define an agent's trust outlook.

3.3.1.1 Definition of trust outlook

In summary, trust outlook refers to the approach a trust model takes towards updating and evaluating trust. This influences the means that an agent chooses to represent trust. The approach can be optimistic, pessimistic or even pragmatic, ranging from slow to fast updates of trust in either direction. An agent's trust outlook ultimately influences the behaviour of a trust model, the constructs used by a trust model and the sensitivity towards changes in trust.

3.3.1.2 Advantages and disadvantages of various approaches towards trust outlook

When the various trust outlooks that are identified are considered, it becomes apparent that each has an effect on the system in which it is implemented. These approaches possess both advantages and disadvantages that influence the manner in which trust is handled. The blind

approaches identified are exactly that – blind. They do not track experiences contrary to those in which they are interested and consequently have no means by which to adjust trust in the opposite direction. Even so, each approach also has its own advantages. The blindly positive approach allows an agent to always be open to new opportunities, while the blindly negative approach greatly decreases risk.

Optimistic agents present a system with more opportunities for interaction, but they also expose a system to greater risks. Pessimistic agents may lower risk exposure, but they risk closing the system off from opportunities when the reason for failure is only temporary. Pragmatic or balanced agents lie somewhere in-between, balancing the advantages and disadvantages of both the optimistic and the pessimistic approaches to varying degrees. The identified advantages and disadvantages are summarised in Table 3.1.

3.3.2 Passionate versus rational approaches

Several trust models define the way in which agents represent and analyse trust (Stoneburner 2001; Pirzada & McDonald 2004; Sung & Yuan 2001). The ways in which agents represent trust include trust labels (Abdul-Rahman & Hailes 2000), a single trust value (Marsh 1994), matrices (Guha et al. 2004) and certifications that must be adhered to (Perlman 1999). These different ways of representing trust have been effectively summarised by Jøsang (1997) as being either passionate or rational.

3.3.2.1 Passionate agents

Passionate agents are considered to have free will and thus act very much like humans. A passionate agent is expected to be either benevolent (behaving in an honest manner and following the expected rules) or malicious (being deliberately dishonest and setting out to cause harm, yet hiding its intention to do so). Passionate agents develop trust according to intangible principles, or rather principles that have no specific binding mathematical rules attached to them. They rely on a loose mathematical formulation of various principles such as collaboration, experience, reputation and recommendation; and they are represented by intuitive constructs and labels.

These principles develop in a dynamic way and change over time as agents are exposed to new and differing experiences.

Jonker and Treur (1999) discuss two concepts that are similar to the passionate and rational approaches identified by Jøsang (1997): quantitative and qualitative trust. These two categories influence the way in which data is handled and established, and the two concepts can be linked to passionate and rational trust. Qualitative trust is a more intuitive approach and is used by passionate agents. This type of trust is most commonly represented by the use of labels and terms, as opposed to numbers. It allows for agents to include their own perceptions in the trust evaluation process. This approach is more useful for dynamic environments where there is no central uniform trust.

3.3.2.2 Rational agents

Rational agents are system-like agents, governed by a specific set of logical rules that tend to remain static. Rational agents take a rational approach towards trust. Trust is therefore represented by a series of constraints that filter out untrustworthy and untrusted agents. A rational agent will resist attempts at malicious manipulation from other agents. A rational agent is often instructed to trust on behalf of passionate entities (Jøsang 1996). The most common rational agent is a firewall, which grants and denies access according to fixed mathematical rules that have been predefined.

Quantitative trust is used by rational agents. It is numerical and is usually represented by a single numerical value that exists within a specific range. This type of data is easily implemented in a computerised environment and the result of the trust evaluation can be easily predicted for any agent in the environment. This approach works well in centralised environments where the environment remains controlled.

3.3.2.3 Definition of passionate vs. rational

The passionate vs. rational criterion is used to identify whether a particular model adopts a rational or passionate approach towards trust. The passionate approach is a more intuitive

approach that allows an agent to make use of its own perceptions to establish trust. The approach also makes use of terms and labels to determine a trust value. The rational approach is more strictly defined than the passionate one and relies on logic and strict mathematical rules for establishing a trust value.

Determinations whether an agent is passionate or rational incorporate the concepts of quantitative and qualitative trust discussed by Jonker and Treur (1999). Consequently, the best criterion that can be used to classify a trust model in this regard is to decide whether a trust model is passionate or rational.

3.3.2.4 Advantages and disadvantages of passionate vs. rational approaches

Passionate agents are more adaptable than rational ones, allowing perceptions to influence the trust value obtained. These agents are more likely to take risks in order to obtain greater rewards. However, the passionate approach results in exposure to greater risk. Due to the fact that these agents are adaptive they require higher processing overheads to maintain the trust environment.

Rational agents are more structured than their passionate counterparts. This makes them more predictable and allows for simple forms of trust to be used. The rational approach loses the intuitive approach towards trust that passionate agents possess, providing a very limited form of trust.

3.3.3 Centralised versus decentralised trust

Trust models can be either centralised or decentralised (Wang & Vassileva 2004), depending on the environment for which they are intended. This has an effect on trust representation. The location where the trust information is stored has the largest impact on trust representation. Centralised structures allow a single centralised node to gather information from all other the agents involved in interactions. This node manages interactions and ensures that all the parties concerned abide by the same trust definitions. A centralised approach works well when the central point is reliable and trustworthy, and when only simple processing is required. It is not

very effective in environments where a large number of agents need to co-exist and are allowed to leave and enter the system dynamically.

Decentralised systems are a bit more involved in the trust establishment process than centralised nodes. Trust is subjective; and it is established individually by each and every agent for other agents (Liang & Shi 2005). Thus, agents' levels of trust in others vary from agent to agent, depending on each individual subjective context. Reputation is not global, and the process of acquiring a reputation requires an agent to query other agents and combine the results received so as to obtain a global estimate. This form of establishing trust implies high communication overheads.

3.3.3.1 Definition of centralised vs. decentralised trust

Using the centralised vs. decentralised criterion in order to define whether a trust model makes use of centralised or decentralised trust determines the location of the trust- related information and the manner in which this trust-related information is handled. A centralised model stores all its trust-related information at a central point in the system. This centralised point stores all the trust information for agents in the trust environment and results in all agents having the same trust values for one another. A decentralised model requires that each agent handles its own trust processing and stores its own trust information for other agents in the environment.

3.3.3.2 Advantages and disadvantages of centralised vs. decentralised trust

A centralised approach allows for a single global trust value to be kept throughout the system, thus simplifying the trust evaluation process. Such an approach also lessens the space the system requires for storing trust-related information. However, centralised trust makes it difficult to include perception in the process of trust evaluation. This forces all agents to have the same opinion and makes it difficult to track dynamic agents that enter and leave the system.

Decentralised trust is a more dynamic type of trust that allows agents to have their own perceptions about other agents in the environment and to use these perceptions to establish trust. The downside of decentralised trust is that each agent needs to define its own trust-building

processes as well as trust-related information, thereby increasing both space and processing required by each agent.

3.3.4 Trust vs. distrust

Many trust models use a single value over a specific range in order to represent trust (Marsh 1994; Marx & Treur 2001; Boon & Holmes, 1991). This value is known as a trust value. For instance, values in the range between -1 and 1 are used to represent trust (1 represents trust in another agent and -1 represents distrust). This representation is simple and can be quite effective. However, it fails to differentiate between a low trust value that is the result of a lack of knowledge and a low trust value that is the result of negative experiences.

Guha et al. (2004) addressed this problem by introducing the concept of distrust as a separate parameter in trust models. They proposed that both trust-related and distrust-related information be stored, and calculated the final trust value for another agent as a combination of the two. However, when working with distrust values, it is important to bear in mind that negative behaviour generally tends to have more of an impact on trust levels than positive behaviour. An interesting quote by Gambetta (1990a: 233) summarises this phenomenon rather well: ‘While it is never that difficult to find evidence of untrustworthy behaviour, it is virtually impossible to prove its mirror image.’ Once distrust has been established, it is often difficult to regain trust (Marsh 1994). Consequently, this issue needs to be addressed in order to successfully include distrust as a separate parameter.

Even though trust is more difficult to prove than distrust, distrust remains an important concept. Distrust has a large effect on trust representation, as well as on the way in which trust is handled by a trust model. However, not all trust models incorporate the concept of distrust, making it important to determine whether this concept is implemented in a given model or not.

3.3.4.1 Definitions of trust vs. distrust

The trust vs. distrust criterion is used to determine whether a trust model has a means by which it can differentiate between low trust values as a result of bad experiences or as a result of too little

information. Most trust models define distrust by using the same variables that they use to define a specific trust level. They consequently merge both distrustful and trustful actions into a single trust value.

Explicitly grouping together the results of actions that cause distrust allows a trust model to differentiate between actions that result in trust and those that result in distrust, while keeping track of how much knowledge it possesses about the trust relationship.

3.3.4.2 Advantages and disadvantages of trust vs. distrust

Measuring only trust is simpler than including distrust and requiring the trust model to merge both the trust and distrust values in order to obtain a single value for the trust relationship it wishes to participate in. However, as already stated, this makes it difficult to determine why a particular trust value is low. The inclusion of distrust as a separate concept into trust models solves this problem but also brings negative consequences, as distrust is often easier to prove than trust.

3.3.5 Trust scalability

Luo et al. (2002) identify an important concept that remains unaddressed in most trust models, namely that of scalability. This concept determines whether a trust model can be implemented in environments that are expected to grow over time, while still continuing to function correctly. The way in which trust is represented influences its scalability. Most authors do not address the scalability of their trust models (Azzedin & Maheswaran 2003; Huynh, Jennings, & Shadbolt 2004; Liang & Shi 2005; Abdul-Rahman & Hailes 1999). Although the concept is not addressed by most trust models, it still influences the trust models, making trust scalability an important criterion in this study.

Due to the fact that scalability remains unaddressed by most trust models, this criterion requires deductive reasoning during the analysis process. A few basic factors should be taken into consideration when discussing the scalability of a trust model, namely processing requirements, space requirements and communication load.

Constant update of experience and historical information allows trust models to calculate and store a dynamic trust value. However, this requires space in which to store this information. The need for storage space can be limited if a trust model is allowed to keep historical and experience-based information only for a short and limited time before overwriting it with new information. Another consideration is the networking capabilities of a particular environment. Some trust models require more messages to be exchanged than others, thus incurring a higher network load. The more complex a trust representation is, the more likely it is that a higher message load will be required to successfully establish a trust relationship.

3.3.5.1 Definition of scalability

The scalability of a trust model refers to the limitations of such a model that may prevent it from being implemented in environments ranging from small to large. The limitations identified include space constraints, processing constraints and communication constraints, all of which may make the trust model inefficient in large network environments.

3.3.5.2 Advantages and disadvantages of scalability

A scalable model can be easily implemented in an environment that is dynamic and expected to grow. The model will adapt along with the environment. However, such a model often requires high maintenance to ensure that new agents in the network conform to the trust model in place. A model that is not scalable does not require such maintenance, but can also not be implemented in environments that are expected to grow.

3.4 Overview of trust representation

Table 3.1 below gives an overview of trust representation and the criteria discussed in the sections above. This table is to be used as a checklist when analysing a trust model for trust representation. The first column of Table 3.1 represents the category and the criteria associated with it. The category is shaded in dark grey while the main criteria therein are shaded in light grey. Each main criterion in this category consists of subcriteria. These subcriteria represent the possible ways in which the particular criterion can be implemented. For instance, when looking at

the passionate vs. rational criteria, a trust model's trust representation can be either passionate or rational. The second column of the table allows this table to be used as a template in order to evaluate a trust model. This column is to be used during the actual evaluation process where insertion of an 'X' beside a particular criteria means that the trust model being evaluated conforms to the criteria beside which the 'X' appears. The use of this column is illustrated later in chapters 8 and 9 which provide sample analysis of trust models using the defined criteria. The last column of the table summarises the advantages and disadvantages of the particular criterion as identified by this study. An 'X' is not put in the grey shaded areas, as the grey shaded areas indicate that a criterion contains a series of subcriteria. The 'X' should consequently be placed next to the relevant subcriterion.

Table 3.1 Overview of trust representation

CRITERIA	X	ADVANTAGES AND DISADVANTAGES
TRUST REPRESENTATION		
Trust outlook		
Blindly positive		ADVANTAGE: Agent is always open to new opportunities, even ones involving other agents that misbehaved initially. DISADVANTAGE: Agent is exposed to greater risk and it is impossible to decrease the trust level.
Blindly negative		ADVANTAGE: Agent drastically lowers risk. DISADVANTAGE: Good behaviour is not tracked, so it is impossible to improve an agent's trust level.
Slow negative, fast positive		ADVANTAGE: Agent is exposed to more interactions and greater opportunities. DISADVANTAGE: Agent is exposed to greater risk, and adjustments based on negative feedback are slow to be made.
Slow positive, fast negative		ADVANTAGE: Agent lowers risk and can increase trust value slowly. DISADVANTAGE: Agent is exposed to fewer opportunities as negative experiences increase.
Balanced slow		ADVANTAGE: The process balances both positive and negative experiences. Due to the slow changes in values, agent is exposed to more interactions and hence has more opportunities before reaching a state of distrust. DISADVANTAGE: Agent is exposed to greater risk when the values decrease slowly.
Balanced fast		ADVANTAGE: The approach balances both positive and negative experiences. Due to rapid changes in trust, agent is exposed to less risk when trust values are decreasing.

		DISADVANTAGE: Agent is exposed to fewer opportunities when values decrease quickly.
Passionate vs. rational		
Passionate		ADVANTAGE: A dynamic, more adaptive approach. DISADVANTAGE: Exposes agents to higher risk and has higher processing overheads.
Rational		ADVANTAGE: Rational agents are less likely to deviate in behaviour. This is a simple form of trust that is easy to represent and calculate. DISADVANTAGE: Rational agents are static agents and do not incorporate more intuitive forms of trust.
Centralised vs. decentralised		
Centralised		ADVANTAGE: A single global trust level can be kept throughout the system. DISADVANTAGE: Impossible to include differing perceptions of trust among agents and difficult to track dynamic agents.
Decentralised		ADVANTAGE: A dynamic approach that includes differing perceptions. DISADVANTAGE: Every agent needs to keep its own trust-based information and requires more processing and storage capabilities.
Trust vs. distrust		
Trust only		ADVANTAGE: A simple representation that requires fewer calculations than when including distrust as a separate concept. DISADVANTAGE: Difficult to differentiate between low trust as a result of bad experience and low trust as a result of lack of information.
Distrust		ADVANTAGE: Allows for differentiation between low trust values as a result of lack of experience or as a result of negative experiences. DISADVANTAGE: Distrust is easier to prove than trust and can lead to false distrust values.
Scalability		
Scalable		ADVANTAGE: A scalable system can be expanded dynamically. DISADVANTAGE: A scalable system often requires higher maintenance.
Not scalable		ADVANTAGE: A system that is not scalable requires low maintenance. DISADVANTAGE: A system that is not scalable cannot be adapted for environments that change.

3.5 Conclusion

Chapter 3 introduced the concept of a set of criteria that can be used to evaluate current trust models and guide future trust model implementations. These criteria have been divided into four main categories: trust representation, initial trust establishment, trust updates and trust evaluation.

Only trust establishment was discussed in detail in this chapter. Each criterion belonging to trust establishment was discussed in detail and the advantages and disadvantages of each were identified before being listed in Table 3.1. In later chapters this table is used as a checklist during the analysis process. Since only trust establishment is detailed in this chapter, subsequent chapters will look at the other three categories and the criteria associated with them.

4. Trust model criteria: initial trust and trust update

4.1 Introduction

Chapter 3 introduced the concept of trust evaluation criteria and divided the criteria into the categories of trust representation, initial trust, trust updates and trust evaluation. These four categories define the four important concepts that need to be addressed by trust model implementation. Since only trust representation has been discussed in detail thus far, the remaining three defined categories still need to be discussed.

Chapter 4 builds on the concepts introduced in Chapter 3 and aims to address the issues still to be discussed. Two further categories of trust model criteria are dealt with in this chapter in detail. Section 4.2 discusses criteria shared by initial trust and trust updates while an overview of these criteria is given in Section 4.3. Section 4.4 discusses the additional concerns with regard to trust update. An overview of these additional concerns is given in Section 4.4. Since trust is often updated in the same way as it is initialised, the same concepts appear in both categories. Section 4.5 concludes this chapter.

4.2 Criteria shared by initial trust and trust update

Initial trust and trust updates are closely linked. Initial trust refers to the strategy a trust model adopts in order to obtain an initial trust value for another agent in the environment, while trust updates allow an agent to incorporate changes in the environment and changes in trust. Since the same concepts are carried over from initial trust through to trust updates, these concepts are discussed in detail in this section with reference to both initial trust and trust updates. Each criterion in this chapter is discussed in the same manner as those in the previous chapter, giving the viewpoints of various experts. Collective definitions as defined by the author are then given. The manner in which each criterion influences both initial trust and trust update is indicated, followed by a discussion of the advantages and disadvantages of each criterion.

The criteria influencing initial trust and trust update as identified in the current study are experience, observation, recommendation, reputation, delegation, collaboration and propagation.

The subcriteria identified for recommendation define whether recommendations are received from intermediaries or agents that are considered to be friends of the agent receiving the recommendation. Subcriteria of reputation include the sources of the reputation information (witness, certified and institutional) and the reputation belief (local and global).

The initial trust and trust update criteria have been divided into two groups – those that take direct approaches and those that take indirect approaches towards initial trust and trust update. Direct approaches, as defined by this dissertation, refer to approaches where an agent forms its own opinion about another agent's trustworthiness without depending on a third agent's perception. Indirect approaches refer to approaches that use information gathered from other agents, as well as the perceptions of other agents, to establish trust.

Two of the identified criteria are defined as direct approaches: experience and observation. All the other criteria are indirect, as they are formed as a result of indirect means and incorporate another agent's perception.

4.2.1 Direct approaches

Acquiring and updating trust in a direct manner requires an agent to be able to formulate trust based on its own perception. An agent uses its own defined trust rules and constraints. As indicated above, two criteria can be classified as direct approaches, namely experience and observation.

4.2.1.1 Experience

According to Sung and Yuan (2001), trust is expected to correct itself with exposure to direct interactions. Consequently, regardless of the assumptions made or even the assurances received, trust is ultimately gained through experience. Most trust models allow experience to have an impact on the trust value, even if the trust model is not entirely reliant on experiences. For instance, McKnight et al. (2002) implement a recommendation-based trust model; yet, at the same time, they allow experience to affect trust. Furthermore, reputation-based trust models rely on experiences to establish and update reputation information (Teacy et al. 2005).

Definition of experience: Experience is a result of a direct interaction between two agents that provides each agent with information that is influenced by the agent's own perception. Experience can be used to evaluate trust in the agent with whom the experience is gained, both to initialise and update trust.

Initialising trust using experience: Trust models that use experience to initialise trust interact with unknown agents directly before establishing a trust level. The result of the interaction is then evaluated and transformed into a trust value. This approach requires several direct interactions to take place in order to establish a stable trust value.

Updating trust using experience: Even if a trust model has not used experience to initialise its trust value for another agent, it uses the experiences it gains from direct interactions with other agents to update its trust value. This allows an agent to adjust the trust value to resemble more closely its own perceptions instead of that of other agents.

Advantages and disadvantages of experience: When using experience to initialise or update trust, an agent is allowed to base its trust values upon its own unique perceptions. However, if experience is used to initialise trust, this implies that an agent does not possess any value for the agent it is interacting with. In order to obtain the trust value it exposes itself to unknown risk by engaging in a direct interaction with the agent it possesses no trust value for. This risk can be minimised by limiting such interactions until an acceptable trust value has been established.

4.2.1.2 Observation

Observation allows an agent to record other agents' behaviour in a networked environment without directly interacting with the other agents. This recorded behaviour is then analysed to establish a trust value for other agents in the environment. Agents include their own perceptions and use their own trust rules and definitions to establish the trust value, which makes this a direct approach.

Pirzada and McDonald (2004) created a trust model that allows trust to be established through observation. Their model takes the results of observation and analyses these results to compute a trust level. Mui, Mohtashemi and Halberstadt's (2002) model goes one step further and allows

observed behaviour to be propagated so as to encourage co-operation. Marsh (1994) considers observation-based models such as those of Pirzada and McDonald (2004) and of Mui, Mohtashemi and Halberstadt (2002) and identifies a flaw in their proposal to establish trust through observation. In a trust relationship based on observation, one agent knows only a little about another agent, and a lot more about the agent's behaviour.

Definition of observation: During observation, information about other agents is gathered passively, simply by allowing an agent to follow what goes on in the network. Observation thus allows trust to be established and updated without exposing an agent to the risk of a direct experience.

Initialising trust using observation: In order to initialise trust through observation an agent is required to record the behaviour of other agents in the environment. This recorded behaviour is then analysed and correlated to establish an initial trust value for an agent in the environment.

Updating trust using observation: Just as observation can be used to initialise trust, it can also be used to update trust. Once an agent possesses an initial trust value for another agent, it is allowed to continue to observe the agent's behaviour. Changes in another agent's behaviour are noted and used to update the initial trust value gained.

Advantages and disadvantages of observation: Observation allows an agent to establish a trust value for another without directly exposing itself to the risk of an interaction, while at the same time still including its own perception into the trust calculation. However, an agent is limited by what it can observe and the manner in which it interprets these observations. An agent that establishes trust through observation only knows how the other agents behave in the particular contexts in which it has observed their behaviour.

4.2.2 Indirect approaches

Indirect approaches to initialising and updating trust allow agents to form a trust evaluation for unknown agents by using information gathered from other agents. Criteria identified in this study that are considered to be indirect approaches include recommendation, reputation, delegation, collaboration and propagation. These criteria and their impact on trust evaluation are discussed in this section.

4.2.2.1 Recommendation

Recommendation-based trust is trust formed by 'word of mouth' and is closely related to reputation (Mui, Mohtashemi & Halberstadt 2002). The key difference is that an agent asks other agents that it trusts to give recommendations, while a reputation is built as a result of evaluations from trusted, distrusted and unknown agents (Whitby et al 2005). Recommendation is an indirect approach, since the information an agent receives has been influenced by the other agents' perceptions. In a recommendation-based relationship, the agents giving the recommendation are known as recommender agents. Agents can act as recommenders and as requestors of a recommendation (Abdul-Rahman & Hailes 1997). An agent analyses the recommendations it receives so as to establish a trust value for other agents in the environment.

In order to be able to trust a recommendation it has received, an agent needs to trust the recommender that gave the recommendation (Montaner, Lopez & Lluis de la Rosa 2002). However, simply trusting another agent does not mean that its recommendations can be trusted as well. An agent may be capable of performing a task asked of it, but due to its own inner perceptions of trust it may not necessarily be trusted to give recommendations. Thus, separate values are required for direct trust in a recommender and trust in a recommender's recommendation (Beth et al 1994).

Depending on the environment in which an agent resides, recommendations can be obtained from two possible sources – central or distributed sources. These sources determine which agents are trusted to be recommender agents. Central environments rely on a central source for trust-related information. Consequently, intermediaries act as recommender agents (McKnight et al. 2002), as they store trust-related information about other agents in the environment and act as mediators for two agents during interactions. Distributed environments rely on distributed trust information due to the nature of this environment. Agents are therefore required to have a list of 'friend agents' that they trust to act as recommender agents.

Definition of recommendation: A recommendation refers to trust-related information that an agent receives about other agents in the environment. This information is influenced by the perception of the agent from which it came and is used to obtain trust values for other agents.

Initialising trust using recommendation: Recommendations can be used to initialise trust. If an agent does not possess any trust information about another agent in the environment, it requests recommendations from agents it trusts. It then analyses the recommendations it receives and establishes an initial trust value for the unknown agent.

Updating trust using recommendation: Using recommendation to update trust requires a community environment where recommendations are continually exchanged. Agents would in such a case receive recommendations about agents it possesses a value for, as well as agents it does not possess a value for. If an agent already possesses a value for an agent it has received a recommendation for, it analyses the recommendation and uses it to update the trust value it possesses.

Advantages and disadvantages of recommendation: The advantages and disadvantages of recommendation differ according to the environment in which the recommendations are found. Agents in a central environment receive their recommendations from an intermediary. Defining the agent that is trusted as a recommender in a centralised environment is rather simple as this refers to the intermediary already in place. The disadvantage of this approach is that agents are limited by the perception of the intermediary. Agents in a decentralised environment are allowed to use their own perceptions to influence their choice of recommender agents to be trusted. This allows them to receive recommendations that mirror their own perceptions. However, the fact that they have to choose the ‘friend agents’ that they trust, complicates their trust evaluation processes.

4.2.2.2 Reputation

Azzedin and Maheswaran (2003:2) formally define reputation as follows: ‘The reputation of an entity is an expectation of its behaviour based on other entities’ observations or the collective information about the entity’s past behaviour within a specific context at a given time.’ From this definition it can be seen that reputation relies on historical information that is connected to an

agent's behaviour (Stakhanova et al. 2004). This historical information is influenced by the various agents that contribute to the establishment of the reputation and is, hence, an indirect approach (Liang & Shi 2005).

When using reputation to establish a trust value an agent sends a query to the environment asking for reputation information. The agent then takes all the information it receives from other agents in the environment and integrates it to obtain its own evaluation on the trustworthiness of the specific agent in question. All agents in an environment are permitted to contribute towards an agent's reputation. The reputation information that an agent receives therefore includes the perceptions of agents it trusts, distrusts, as well as the perceptions of those it does not know.

Reputation requires other agents to freely advertise their opinions about other agents in the environment. These opinions are ideally based on direct experiences that the advertising agent has had with another agent reflecting both behaviour and trustworthiness (Buchegger & Le Boudec 2004). This leads to the problem identified by Xiong and Liu (2003), namely that of negative feedback. Negative feedback often stands out more and has a larger impact on trust values than does positive feedback. Another problem with negative feedback involves the malicious behaviour of some agents who deliberately supply negative feedback so as to lower the reputation of a specific agent. The problem of negative feedback can be combated by requiring that a several different agents provide a reputation value for an agent in the environment (Kamvar 2003). The agents providing the reputation information may be further evaluated for their own trustworthiness in order to evaluate the likelihood of the reputation information being accurate (Whitby et al 2005). This approach provides a more accurate evaluation of reputation but requires that additional processing be done. Agents that deliberately provide negative feedback can also be identified by taking a more statistical approach that analyses the reputation values provided by other agents. This approach assumes that unfair ratings can be identified by the statistical properties. More information regarding this approach can be found in the word by Whitby et al. (2005).

Negative feedback is not the only problem with relying solely on reputation. If an agent decides to suddenly change its behaviour the reputation it possesses at the time of the change works

against it (Huberman & Wu 2004). There is a delay between the time of behaviour change and the reflection of that change in behaviour in the reputation in agent possesses. Aside from these issues, reputation has two main criteria that influence its formation: sources of reputation information and reputation beliefs.

Sources of reputation information: Huynh et al. (2004) identify two sources of reputation information: witness-based reputation and certified reputation. In witness-based reputation, a target's reputation is gleaned by gathering information about its behaviour from other agents in the same environment. This requires an agent to find other agents in the environment that have had direct interactions with the agent(s) in question. Certified reputation information is gathered directly from the agent that an agent wishes to interact with and comes in the form of certifications about the other agent's performance. These certifications have been given to the other agent by agents it has interacted with in the past. Allowing an agent to supply reputation information about itself has the flaw that the agent is allowed to decide which certifications it will supply. It is fairly obvious that the agent will choose the best ones and not the worst ones. This results in over-estimations.

The two sources of reputation identified by Huynh et al. (2004) are greatly influenced by the agents from whence the reputation originates and they can differ greatly, depending on the agent that provides the information. They neglect to take into consideration reputation information of a more central nature. Hence, the two criteria identified by Huynh et al. (2004) can be extended to include one more source of reputation information – that of institutional trust information. The formulation of institutional trust assumes that agents in a certain context adhere to certain specific limitations defined by the context in question (McKnight et al. 2002).

Reputation beliefs: Belief is a result of an agent's local state and the knowledge that it possesses about the global state of the environment (Rangan 1988). Since the global state is made up of the individual states of the agents that reside in the particular environment, knowledge about the global state includes knowledge about other agents in the environment.

The information used to establish belief may be obtained either locally or globally. Consequently Yu and Singh (2002) subdivide reputation into two main beliefs: local and total (which can be seen as global). Local beliefs are formed on the basis of direct interactions and can be propagated

on demand. Global beliefs are the culmination of local beliefs and reports received from other agents about the reputation of an agent in question. These global beliefs are used to determine the trustworthiness of an agent (Yu & Singh 2002).

Definition of reputation: When considering reputation, one needs to also consider the sources of reputation information and the reputation beliefs defined. Both of these influence the final outcome of a reputation evaluation. Reputation can consequently be seen as the culmination of information received from various sources. This information can be a result of witnesses, certifications and institutional assumptions. Furthermore, reputation is influenced by the beliefs of the environment in which the reputation is formed. These beliefs can either be local or global.

Initialising trust using reputation: Reputation can be used to initialise trust. Should an agent not have a trust value for another agent in the environment, it can simply query the agent's reputation in order to receive a trust value.

Updating trust using reputation: An agent's reputation may be updated by other agents in the environment, on a continual basis, as a proactive result of interactions. If an agent wishes to update its own trust value for a specific agent's reputation, it can simply make a query requesting the agent's updated reputation information and use this information to update the trust value it possesses.

Advantages and disadvantages of reputation: Reputation makes use of community feedback in order to establish trust. This allows communities to influence the manner in which trust is established. The greatest disadvantage of this approach is negative feedback. Negative feedback stands out more than positive feedback and malicious agents in the environment may deliberately give this negative feedback to harm an agent's reputation. A second disadvantage of reputation is the time it takes to reflect a behaviour change.

The various sources of reputation information all have their own advantages and disadvantages. Witness reputation allows an agent to influence the reputation it establishes, using its own perception. However, the agent is required to identify other agents that have interacted with the agent it is seeking reputation information for. Certified reputation is a simple query process where the agent seeking the reputation information for a target agent asks the target agent for its

certifications. The disadvantage of this is that the target agent can choose which certifications it shares and will consequently share only the good ones. Institutional reputation simplifies the trust evaluation process by allowing certain assumptions to be made based on the institution an agent belongs to. This has the disadvantage of possible over- or underestimations for individual agents in the institutional environment.

The beliefs that define a reputation also carry their own advantages and disadvantages. Local beliefs allow an agent to have its own unique perceptions of trust but require each agent to have higher processing and storage capabilities. Global beliefs centralise the reputations of agents but limit the unique perceptions agents are allowed to have.

4.2.2.3 Delegation

Delegation allows for the decentralisation of certain tasks. This is considered to be a separate criterion because of the way in which trust is passed on. Delegation allows for the passing on of privileges more than of the actual trust levels themselves (Blaze et al. 1999). If an agent possesses a particular privilege, it is trusted within the limited constraints of that privilege. This is an indirect approach, because the privileges that a delegating agent has, influence the privileges that the agent is able to pass on.

Agents are allowed to give specific privileges to other agents, assuming they have the right to delegate those privileges (Wen & Mizoguchi s.a.). This process allows agents to grant privileges to other agents whom they trust. In the same way, these agents are also allowed to revoke any privileges they may have granted, should their trust prove to have been misplaced and result in abuse of the privileges that have been granted.

Definition of delegation: Delegation allows an agent that possesses specific privileges to pass those privileges on to another agent. The privileges that are delegated grant access to some resources and deny access to others.

Initialising trust using delegation: In order to initialise trust using delegation, an agent needs to request privileges from another agent that possesses those privileges. The other agent then makes the decision whether to delegate those privileges or not.

Updating trust using delegation: An agent that has delegated privileges to another agent needs to observe the agent it has delegated the privileges to. If it finds that the privileges it has delegated are being misused, it updates the trust given by revoking all delegated privileges.

Advantages and disadvantages of delegation: Delegating privileges simplifies the trust establishment process. If an agent possesses a certain privilege, it is granted access to a given resource – if not, then access is denied. Although this simplifies the trust establishment process, it places responsibility on the agent that has delegated the rights (delegator). The delegator needs to ensure that the agent it delegated the rights to does not misuse them.

4.2.2.4 Collaboration

Collaboration requires a group of agents to work together in order to obtain a general trust evaluation that the entire group adheres to. This is a community-based approach whereby agents form groups of collaborating agents. Jones (1990:3) defines collaboration as the process during which 'one works jointly with other[s] on a task'. Marsh (1994) extends the definition of collaboration to include co-ordination. In order for collaboration to be successful, co-ordination of the various parts participating in the collaborative relationship is required. From the very nature of co-operating to complete a task, it can be deduced that collaboration relies on a social context of trust.

When an agent receives information, recommendations or requests from an unknown agent in the same environment, it seeks to establish trust (Agarwal et al. 2003). The agent seeks out its friends or trusted agents with whom it collaborates and together with these agents it analyses the information to obtain a trust value for the agent the information is about. Information is gathered

from all the collaborators and brought together. The group of agents then collaborates and makes a joint decision on the trustworthiness of the agent in question. This allows agents to work together to make joint decisions regarding initial trust values.

Definition of collaboration: Collaboration refers to the process by which several agents co-ordinate their efforts to analyse trust-related information, recommendations and requests. This analysis process results in a trust value that will be used to limit interactions with other agents in the environment.

Initialising trust using collaboration: Collaboration may be used to initialise trust. If an agent receives requests, information or recommendations from or about an unknown agent, it seeks out the agents with whom it collaborates. The agents then share all information they have regarding the unknown agent and reach a joint decision regarding the trust value that is to be assigned to the unknown agent. This value is used as the initial trust value.

Updating trust using collaboration: In the same manner that collaboration can be used to initialise trust, it can also be used to update trust. If an agent wishes to update trust using collaboration, it need to share all new information and results of interactions with the agents it collaborates with and analyse this new information along with the old in order to obtain a new trust value.

Advantages and disadvantages of collaboration: Collaboration allows several agents to share the trust evaluation load. The disadvantage of this approach is that an agent normally wishes to collaborate with agents whose opinions are similar to its own. In order to successfully do so, it is required to find such agents.

4.2.2.5 Propagation

Propagation is a way of distributing trust throughout a distributed environment. Trust by propagation requires agents to forward various kinds of information to their neighbours, who then forward this information to their neighbours; and so it continues, until the information has been spread throughout the trust environment. Information that is forwarded can vary from information on the results of interactions, to complaints and praise (Aberer & Despotovic 2001). Agents subsequently use this information to form initial trust opinions or to update existing trust

structures. When analysing information received through propagation, agents need to take into consideration their trust level in the agent that propagated the trust result. More weight should be given to trust information propagated by reliable trusted agents in the environment than to information propagated by those agents that are considered unreliable.

Definition of propagation: Propagation is the process by which interaction results, complaints and praise are forwarded from the place that they originate to other agents in the environment, who then forward them further. Agents in the environment use this forwarded information to initialise and update trust values for other agents in the environment.

Initialising trust using propagation: An agent with no prior knowledge of trust in a particular environment receives propagations that contain interaction results, complaints and praise, and uses these to form initial trust structures of its own. Direct experiences later influence these results and the results of such experiences are propagated back throughout the environment.

Updating trust using propagation: Agents that receive propagated information may also use this information to update their trust values. If an agent already possesses trust information for agents it receives propagated information about, the agent analyses the new propagated information along with the information it already possesses and updates its trust values.

Advantages and disadvantages of propagation: Propagation allows for both negative and positive feedback to be shared with other agents in the environment. This negative and positive feedback gives other agents in the environment vital information regarding the trustworthiness of agents that this feedback is about. However, the very process of sharing all feedback received from interactions throughout the environment requires that several messages pass through the network to share just one interaction result. This carries the danger of overloading the network with these messages.

4.3 Overview of criteria shared by initial trust and trust update

Table 4.1 provides an overview of the criteria shared by initial trust and trust update. This table is read and interpreted in the same way as Table 3.1 (in the previous chapter). The first column lists the criteria identified, the second column is to be used as a checklist for the criteria a trust model should conform to and the last column lists the advantages and disadvantages of a particular criterion. The text in italics represents the criteria while the text not in italics represents the subcriteria that belong to the criteria listed above the text involved.

Table 4.1 Overview of initial trust

CRITERIA	X	ADVANTAGES AND DISADVANTAGES
INITIAL TRUST		
Direct approach		
Experience		*ADVANTAGE: An agent reaches a trust value based on its own perceptions and judgements.* *DISADVANTAGE: When an agent has not had an experience with another, it also does not possess any trust information about the other agent and exposes itself to risk in order to get this information.*
Observation		*ADVANTAGE: Information can be gathered about agents without direct exposure to risk and can be filtered to include an agent's own perception.* *DISADVANTAGE: An agent knows only about the behaviour of other agents in a given context.*
Indirect approach		
Recommendation		
Intermediaries (Centralised)		ADVANTAGE: Definition of trusted recommender agents is simple. DISADVANTAGE: The intermediary limits agents' perceptions.
Friend agents (Decentralised)		ADVANTAGE: Agents are allowed to have different perceptions of trust and use these perceptions to choose trusted recommender agents. DISADVANTAGE: Agents need to establish trusted friends (recommenders).
Reputation		*ADVANTAGE: An agent is able to obtain a trust evaluation for another based on the community feedback that influences reputation.* *DISADVANTAGE: Often negative feedback does not occur when it should and does not have the impact it should have on a reputation.*

Sources		
Witness reputation		ADVANTAGE: An agent is able to gather information and make its own evaluation of the information it has gained. DISADVANTAGE: An agent needs to identify agents that have interacted with the agent for which it seeks a reputation.
Certified reputation		ADVANTAGE: Agents are able to store their own reputation information and share it with others on demand. DISADVANTAGE: Agents can decide which certificates to give and which to withhold.
Institutional reputation		ADVANTAGE: Institutional assumptions simplify the trust evaluation process. DISADVANTAGE: Institutional assumptions may lead to over- or underestimations of the trustworthiness of individual agents.
Belief		
Local (decentralised)		ADVANTAGE: Each agent may have its own unique perceptions. DISADVANTAGE: Each agent is required to have a higher storage and processing capability than agents relying on global beliefs.
Global (centralised)		ADVANTAGE: A single global value for trust can be uniformly controlled. DISADVANTAGE: Limited trust perception.
Delegation		*ADVANTAGE: Access to resources can be delegated, thus simplifying the trust evaluation process.* *DISADVANTAGE: The delegator needs to track for misbehaviour and inform the agents it has delegated certain trust values to in cases of misbehaviour.*
Collaboration		*ADVANTAGE: Agents share the trust evaluation load by sharing information.* *DISADVANTAGE: Agents need to find other agents that they can collaborate with.*
Propagation		*ADVANTAGE: Both negative and positive feedback can be shared with other agents, allowing other agents to update their trust values even without direct experience.* *DISADVANTAGE: Many messages are passed through the network. Ways of sharing the information with appropriate agents need to be devised.*

4.4 Trust update's additional concerns

Agents update trust in order to gain more accurate trust values and to protect themselves from other agents that used to be trustworthy but that have become malicious over time. As discussed

in the previous section, trust can be updated in the same ways in which initial trust has been established. Having already discussed this process, this section now only looks at the additional considerations that are unique to trust update. These additional concerns are feedback and time.

4.4.1 Feedback

A famous quote by Ronald Reagan summarises the core philosophy of the principle of feedback: 'Trust but Verify' (Shrobe, Doyle & Szolovitz 1999). It is important to monitor what happens within a trust environment. Successfully established trust does not always mean that an interaction will not fail and that the trust will not be betrayed. The environment in which agents exist is dynamic, and so is the nature of trust. Changes in the environment and the development of agents over time could lead to a situation where previously trusted agents end up acting in a manner contrary to their original trust setup. Thus, a way of updating trust is required.

Updating trust requires some form of evaluation. In order for this evaluation to occur, the agent ultimately requires some form of feedback that it can use as input to the evaluation process. The feedback can be a result of direct interactions, or it can be obtained by one agent about another through delegation, recommendation, propagation or any other indirect means. The feedback needs to be adaptive, ideally allowing a model to seek more updates and feedback during a time of vulnerability or attack than it would during normal operation (Sung & Yuan 2001).

4.4.1.1 Definition of feedback

Feedback refers to information an agent receives after a trust level has already been established for another agent. This feedback can be a result of an agent's own direct interaction with the agent in question or another agent's direct interaction with the agent in question. If the feedback is a result of another agent's direct interaction with the agent in question, the agent receives this feedback in an indirect manner, either through propagation, recommendation or any of the other indirect methods already discussed.

4.4.1.2 Advantages and disadvantages of feedback

Feedback ensures that trust values change over time, allowing trust to be dynamic. This is necessary because of the nature of trust and the environment in which it exists. A disadvantage to feedback is usually the processing power required to analyse this feedback. This disadvantage has a greater impact when large quantities of feedback are handled by the trust-updating process.

4.4.2 Time

When one is storing and engaging in trust evaluations, it is important to consider the impact of time on trust. Environments, needs and even criteria change over time. A trust model should be capable of capturing this change and of changing trust levels appropriately (Yu & Singh 2002; Pirzada & McDonald 2004). The most effective means of capturing and changing trust over time is feedback, as has already been discussed. This illustrates the obvious link between feedback and time. However, this is not the only manner in which time can affect trust.

An agent does not always receive feedback for agents it possesses trust values for. This does not mean that the trust has not changed over time. Interacting with an agent for which an old trust value exists, can expose the agent to great risk especially if the agent in question has become malicious over time. Azzedin and Maheswaran (2003) and Ray and Chakraborty (2004) address this problem by allowing trust to decay over time. This allows time to have an effect on trust without the need for feedback.

4.4.2.1 Definition of time

Time is an important factor that affects trust. Usually feedback occurring over a given period of time can be used to successfully update trust values. If this feedback does not occur, time itself can be used to update trust by forcing trust values to degrade over time.

4.4.2.2 Advantages and disadvantages of time

Allowing time to degrade the trust value when no feedback is received lowers the risk an agent takes when interacting with agents it has not received feedback for over a long period of time. The disadvantage, however, is that the agent loses trust in the agent in question and will have to

re-establish a good trust level even if the agent in question has remained trustworthy over this time.

4.5 Overview of trust update's additional concerns

Table 4.2 is to be read in the same manner as Table 4.1 and provides an overview of the additional considerations related to trust update by listing advantages and disadvantages of each. These considerations expand on the concepts discussed under initial trust with regard to trust update.

Table 4.2 Overview of trust-updating processes

CRITERIA	X	ADVANTAGES AND DISADVANTAGES
TRUST UPDATE		**Usually as result of experience or new information**
Additional considerations		
Feedback		ADVANTAGE: Agents are able to use feedback to change a trust value after an interaction. DISADVANTAGE: Processing power is required to analyse feedback.
Time		ADVANTAGE: An agent is exposed to lower risk if it allows trust to degrade. DISADVANTAGE: Allowing trust to degrade over time can result in underestimations if the agent has not actually become untrustworthy.

4.6 Conclusion

Trust model criteria have been proposed in order to assist in the evaluation of trust models. The first of four identified categories (namely trust representation) was discussed in Chapter 3. This chapter extended on the work done in Chapter 3 by discussing two more of the categories: initial trust and trust update. These two are discussed together because they are closely related. Trust updates often use the same concepts as those identified by initial trust.

Tables 4.1 and 4.2 provide an overview, as well as the advantages and disadvantages of the concepts discussed. A checklist for each criterion associated with the different categories is also provided. This ultimately assists in the identification of a trust model's strengths and weaknesses. Since the first three categories of trust model criteria have now been discussed, only the last

remains to be examined. Chapter 5 looks at the last category (trust representation) and uses the same approach as that taken in Chapters 3 and 4.

5. Trust model criteria: trust evaluation

5.1 Introduction

Three of the categories of trust model criteria identified by the author have already been discussed in detail: trust representation, initial trust and trust update. Trust evaluation therefore remains the last to be discussed in detail. This category looks at the impact trust evaluation has on the manner in which a trust model operates.

Chapter 5 concludes the discussion of the trust model criteria that have been proposed based on current trust model implementation. The last category indicated in Chapter 3, namely trust evaluation, is discussed in detail in Section 5.2. This section looks at the criteria associated with trust evaluation and discusses them in detail. Section 5.3 provides an overview of the trust representation criteria discussed and Section 5.4 concludes this chapter.

5.2 Trust evaluation

Trust evaluation refers to the processes by which trust models evaluate trust in order to obtain a trust value for another agent. Trust evaluation processes are used to establish initial trust and to update trust. The result of a trust evaluation defines the limitations that are placed on interactions. Many trust models rely on categorisation to limit the trust given to an interaction. This trust categorisation is used to evaluate an interaction and occurs in various ways. However, several other criteria influence the success of trust evaluations. These criteria include trust context and risk, as well as the difference between approximate vs. dynamic evaluation. Each criterion is discussed in detail with regard to the viewpoints of various experts. This discussion is followed by a collective definition of the concept as given by the author for the purpose of this study.

5.2.1 Trust categorisation

Categories greatly assist trust evaluation. Grouping agents into specific categories allows for assumptions to be made during trust evaluation. These assumptions are defined by the characteristics of a particular category and simplify the trust evaluation process. Two main

methods of creating various categories have been identified by Li, Valacich and Hess (2004), namely reputation and stereotyping.

Reputation allows an agent to create and analyse categories as a direct result of the reputation information it receives about other agents. The assumptions that are made depend on the environment an agent resides in and the reputation value it possesses. Hence, the assumptions that are made are not based on the specific characteristics of the agent that possesses the reputation in question. Stereotyping allows for agents to make assumptions about other agents, depending on which category an agent falls into (Khare & Rifkin 1997). For instance, agents can be categorised according to the organisation they belong to, the rights they possess, the roles (Shand et al. 2004) they play and even the policies they are required to adhere to. Each category is allocated specific privileges and rights associated with that category (Twigg & Dimmock 2003).

5.2.1.1 Definition of categorisation

Categories allow the trust evaluation process to make certain assumption, thereby simplifying the actual trust evaluation process. The two main types of categories identified influence the manner in which the trust evaluation process interprets the category. Reputation-based categories allow for environmental assumptions, while stereotyped categories allow for assumptions with regard to the agent itself.

5.2.1.2 Advantages and disadvantages of categorisation

Each type of category identified possesses its own advantages and disadvantages. Reputation categories allow reputation level to determine the assumptions made and are influenced by the environment. The problem with these assumptions is that they are based on environment and reputation level and not on the actual actions that resulted in a particular reputation level. Stereotyped reputation allows agents to be grouped according to their characteristics. The problem with this approach is that the characteristics are generalised and limited to predefined categories.

5.2.2 Trust context

Trust is often based on situation. An agent, perhaps wishing to exploit another agents trust, may, under certain circumstances, supply information that it itself does not believe in (Jones & Firozabadi 2000). It is also possible that an agent that has always been known to be trustworthy has been compromised by external factors such as a hacker who may influence the agents known trusting behaviour. This phenomenon is discussed by Jones and Firozabadi (2000) who point out that the performance of a certain act does not always result in the same resulting state due to the environmental factors that influence the reason for the act. Consequently an agent needs to take into consideration situational constraints before it chooses to participate in a trust-based interaction. Such constraints include capability, the need for an interaction and the state of the environment (Wang & Vassileva 2004). Often the environment changes due to extenuating circumstances, such as the crash of a critical machine or hacker activity. Agents need to be able to detect such changes in the system and take alternative action, especially if such unexpected changes have influenced a critical part of the system as a whole (Shrobe et al. 1999). Thus, it is important to consider context when evaluating and updating trust.

A contextual factor that may influence the formation of trust is intention. The reason an agent seeks to establish a trust relationship is a clear indicator of a need that has been expressed. It is also the determining factor as to which information will be shared and which will be withheld. The context in which a transaction occurs is also critical when evaluating the feedback that is received. For instance, the size of a transaction is indicative of the effort that was required to complete the transaction successfully. Small transactions carry a lower risk than large transactions and so a small successful transaction should have a smaller influence on trust than a large successful transaction does.

Contexts include both a transactional context and an environmental context. The transactional context includes the size, category and time of a transaction. Environmental contexts are more concerned with the state of the environment at the time of a transaction. Such contexts include which agents were running and which were not, any suspicious increase in activity, network

overload and the addition of new agents (Li & Liu 2004). In order to evaluate and update trust accurately, both the transactional and the environmental context need to be considered.

5.2.2.1 Definition of context

Context refers to the environmental and transactional factors that influence the success of an interaction. These factors determine the risk that a particular interaction carries and, consequently, influence the trust that is assigned to a given interaction. Contextual factors that provide higher risks should have a greater impact on trust level when interactions defined by these factors are successful.

5.2.2.2 Advantages and disadvantages of context

Having explicit trust evaluation rules that take context into consideration allows for more accurate trust evaluations to occur. However, context evaluation requires additional processing rules as well as time in order to be successful. Ignoring the context is less process intensive but also makes a trust evaluation less accurate.

5.2.3 Risk

Due to the dynamic and changing nature of trust, trust relationships carry some form of risk. In order to address this factor successfully, risk needs to be assimilated into the decision-making process and accepted as inevitable. The very definition of trust implies some form of uncertainty. It is this very uncertainty that introduces risk into the interaction. Trust evaluation needs to consider the risk an interaction carries. This is due to the fact that interactions with greater risk require greater trust.

There are several approaches towards controlling and evaluating the risk an agent is exposed to. The approaches identified in this study are fallback mechanisms, constraints, ignorance and risk evaluation during trust update. Fallback mechanisms assist models in dealing with risk. The fact that an agent has a fallback mechanism allows it to take greater risks, while improving the agent's chance of recovering from failure.

Knowledge of risk allows the agent to make plans that take this risk factor into account, thus allowing agents to limit their interactions more if the risk is greater (Marsh 1994). In order to successfully limit the risk taken, an agent needs to be able to evaluate the type and level of risk a possible interaction poses. One way to control the risk that is encountered in an interaction is to place constraints on both the truster and the trustee, while requiring the interaction to take place within these predefined constraints (Li, Valacich & Hess 2004).

The very presence of risk in trust relationships has the potential to influence trust levels both positively and negatively. Consequently, the risk an interaction carries is important to consider during trust evaluation. The successful completion of an interaction that poses a risk boosts the trust level. Unsuccessful completion can cause a serious drop in the trust value, especially if the risk was large and serious harm has resulted from the interaction.

5.2.3.1 Definition of risk

Risk refers to the potential harm an agent exposes itself to. The greater the risk an agent exposes itself to, the greater the potential harm that can be done to it. Consequently, taking greater risks requires larger trust values. By the same reasoning, successful interactions that carry large risks should have a greater effect on trust than those that carry smaller risks.

5.2.3.2 Advantages and disadvantages of risk

The various approaches towards handling risk involve advantages and disadvantages. Having a fallback mechanism ensures that an agent is able to recover from disastrous failure. This is a costly means of managing risk recovery as it requires additional space and processing to ensure that these fallback mechanisms are in place. Placing constraints on interactions controls the level of risk associated with an interaction but limits the interaction itself. Evaluating the level of risk that an agent took during trust updates allows for a more accurate trust update to occur but requires additional processing.

5.2.4 Dynamic versus approximate evaluation

When one is implementing trust, it is important to decide on the manner in which one wishes one's trust evaluation to occur. Li and Liu (2004) identified an important trade-off between the accuracy of a trust evaluation and the processing power required to do that trust evaluation. In order to get an accurate trust evaluation, a more dynamic approach is taken that continually incorporates changes in the environment and agents' interactions into the trust evaluation. This is clearly a time-costly procedure. In order to save on processing power and time, an agent can choose to do an approximate evaluation of trust. An approximate evaluation of trust evaluates the state of trust at a particular time and uses the result for a particular period. This disregards the number of interactions that actually occur during that period. Shorter time intervals between evaluations allow for trust evaluations to lean towards dynamic evaluations. This alternately carries the risk that a trust evaluation may be inaccurate.

Different environments have different attributes and the properties that environments consider to be vital differ. This also applies to trust evaluation. Systems with high risks may opt for a more dynamic approach, while stable systems that do not change much in a short time may opt for approximate evaluations. Environments thus have an impact on the trade-off that is acceptable. When one is analysing a trust model for implementation, this criterion is clearly an important consideration.

5.2.4.1 Definition of dynamic vs. approximate evaluation

Determining the type of trust evaluation that occurs determines the impact of the evaluation on the trust value obtained. The trust value can, consequently, be dynamically accurate or approximately generalised. This ultimately influences the effect of time on trust. Time affects a dynamic evaluation to a larger extent than an approximate one. However, due to the dynamic nature of trust, even approximate evaluations are eventually affected by the passage of time.

5.2.4.2 Advantages and disadvantages of dynamic vs. approximate evaluation

As already identified, there is a trade-off between dynamic and approximate evaluation. Dynamic evaluation is more accurate but involves a higher processing cost. An approximate evaluation does not require as much processing power as a dynamic one but provides a less currently accurate trust value.

5.3 Overview of trust update

Table 5.1 summarises the various criteria identified for trust evaluation and lists them in the first column. The second column can be used as a checklist to indicate which criteria a particular trust models conforms to. The last column lists the advantages and disadvantages of a particular criterion.

Table 5.1 Overview of trust evaluation

CRITERIA	X	ADVANTAGES AND DISADVANTAGES
TRUST EVALUATION		
Categorisation		
Reputation		ADVANTAGE: Agents can make assumptions according to reputation level. DISADVANTAGE: Assumptions are controlled by the reputation value and not the reason for the given reputation.
Stereotyping		ADVANTAGE: Agents make assumptions according to category. DISADVANTAGE: The categories are limited to those that have been defined or by the rules used to define them.
Context evaluation		
Takes context into consideration		ADVANTAGE: Evaluating context results in a more accurate trust evaluation. DISADVANTAGE: Additional processing is required to analyse context.
Does not take context into consideration		ADVANTAGE: Less processing is required to do a trust evaluation. DISADVANTAGE: The trust evaluation is less accurate.
Risk		
Fallback mechanism		ADVANTAGE: It allows an agent to recover from a disastrous state. DISADVANTAGE: Agents require more processing and backup space.

Constraints		ADVANTAGE: Constraints control the risk associated with an interaction. DISADVANTAGE: Constraints limit the interaction.
Risk evaluation during update		ADVANTAGE: This allows for updating trust information after each interaction and a more accurate trust evaluation. DISADVANTAGE: It requires more processing.
Dynamic vs. approximate evaluation		
Dynamic		ADVANTAGE: It allows for changes to influence trust level. DISADVANTAGE: One needs more processing power.
Approximate		ADVANTAGE: It takes less processing power. DISADVANTAGE: It results in less accurate trust evaluations.

5.4 Conclusion

The four categories of criteria – trust representation, initial trust, trust updates and trust evaluation – have been proposed in order to facilitate the analysis of currently proposed trust models, as well as to guide future implementations. This chapter has concluded discussion of the four identified categories by discussing the last of the categories, namely trust representation, in detail. As with Chapters 3 and 4, a table containing the criteria and the advantages and disadvantages for each was given. Each criterion identified in the table was then discussed in detail.

Since the various categories and criteria identified are based on current work in the field of trust models, future changes in the field of trust models will affect these criteria. In the current study, an extension to these criteria is identified that is not currently implemented by trust models. This extension refers to an additional category, namely prejudice filters, as will be discussed in Chapters 6 and 7 of this dissertation. Chapters 8 and 9 will then take the categories and criteria identified and show how they apply to an existing trust model.

PART 3

PREJUDICE FILTERS

6. Prejudice filters

6.1 Introduction

Trust models have been proposed to minimise the risk of sharing information with other agents (Ramchurn et al. 2003; Li, Valacich & Hess 2004; Li, Lyu & Liu 2004). Furthermore, important criteria by which to evaluate these trust models have been identified and discussed as these criteria influence the trust evaluation process. The process required by trust models to obtain a trust value is a long and intricate process. The intricacy of the evaluation process depends on the trust model definition itself. A more accurate trust evaluation implies higher processing overheads. Valuable resources are tied up if the results of a lengthy evaluation simply result in a distrust value. Distrust refers to the firm belief in an entity's incompetence (Ray & Chakraborty 2004). Allowing interactions under conditions of distrust consequently expose a system to risk. Some systems may deem this risk acceptable due to a potential high reward but such is not the norm. Depending on the nature of the business, keeping system resources tied up in evaluating distrust that will simply result in refusal to interact may be undesirable. In order to control the number of potential interactions that will result in a negative trust evaluation, prejudice filters have been proposed. Chapter 6 defines the concept of prejudice from a psychological perspective (see Section 6.2). The concept is then incorporated into the basic trust model architecture (as defined in Chapter 2) in Section 6.3. Specific filters and their implementations are discussed in Section 6.4, and Section 6.5 concludes this chapter.

6.2 Prejudice

Just as humans rely on trust to limit interaction, they also rely on prejudice to filter out unwanted communications and simplify the environment. Hence, since trust models rely on the human cognitive definitions of trust to limit interaction, filters based on the cognitive definition of prejudice have been proposed to filter computerised trust interactions (Wojcik, Venter, Eloff & Olivier 2005).

Prejudice is an extension of the concept of trust-building processes. It can be defined as a negative attitude towards an entity, based on a stereotype (Bagley et al. 1979). Prejudice is

negative in nature and tends to lean towards negative assumptions about a given entity, concept or situation. Prejudice influences trust by allowing certain negative assumptions to be made about certain groups. This stereotypical belief is defined and coloured by the culture from which it stems (Simpson & Yinger 1985). Entities of a certain stereotyped group are placed in the same category (Fiske 2000), allowing assumptions to be made and simplifying the processing required before trust can be established. It is important to note that prejudice does not necessarily lead to a refusal to interact with an individual belonging to a certain group. In actual fact, in some cases it takes the form of simplifying and limiting interactions with such an individual. This involves allowing interactions to occur in limited contexts, which leads to very limited forms of trust (Glassner 1980).

Prejudice can thus be seen as a negative attitude towards something that allows for stereotypical assumptions and is based on the environment from which this negative attitude stems. The prejudice filters proposed for incorporation in trust models rely on this basic definition of prejudice.

6.3 Prejudice filters

Prejudice filters, for the purposes of this research, are defined as code that relies on the concept of prejudice to filter out agents that have a high likelihood of being untrustworthy. Agents see prejudice filters as simplified trust rules that rely on the concept of prejudice in order to limit the number of interactions an agent needs to analyse in detail. Prejudice filters furthermore rely on broad definitions of attributes that lead to distrusted interactions, denying interactions that can be defined by these attributes. For example, if an agent has interacted with another agent from a particular organisation and the interaction has failed in terms of expectations, future requests from agents belonging to the same organisation will be discriminated against.

These prejudice filters affect two phases of the three-phase interaction cycle originally identified by Ramchurn, Jennings, Sierra and Godo (2004) and applied to the concept of trust models in Chapter 2 of this study (see Figure 2.1), namely negotiation and outcome evaluation. The effect of the prejudice filters is illustrated in Figure 6.1. The three phases are illustrated by labelled boxes that encompass the agents involved. The boxes illustrate which agents participate in a

particular interaction phase. The first two phases occur between two agents that want information to be exchanged. The outcome evaluation process occurs individually on each agent's side and no longer requires an exchange of information. In the negotiation phase, the prejudice filters are consulted first in order to provide a quick, simplistic evaluation of trust and filter out unwanted interaction requests before they are required to go through detailed trust evaluation and definition.

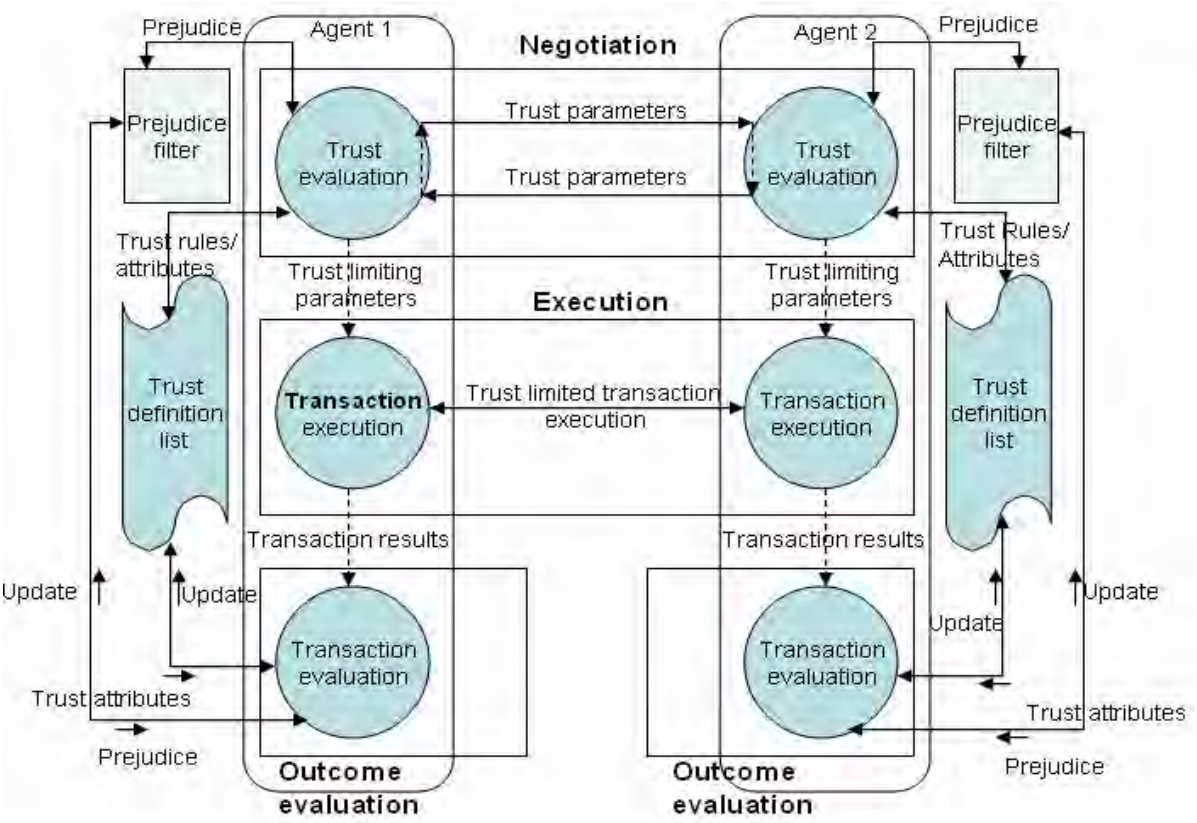

Figure 6.1 Operation of an agent using a trust model with prejudice filters

When the transaction execution phase is concluded, the outcome evaluation phase commences. The outcome evaluation phase evaluates the interaction, bearing in mind prejudice filters so that these filters can be updated with experience. Failed transactions update the prejudice filters in order to filter out other transactions of a similar nature at an earlier stage in future.

6.4 Extending existing trust models to include prejudice filters

Existing trust models rely on various ways of establishing initial trust and updating trust. These include experience, recommendation, observation, reputation, delegation, collaboration and propagation (Hultkrantz & Lumsden 2001; Patton & Jøsang 2004; Papadopoulou et al. 2001; Carbone, Nielsen & Sassone 2003; Marx & Treur 2001; Jonker & Treur 1999; Guha et al. 2004; Xiong & Lui 2003; Ramchurn, Sierra, Jennings & Godo 2004). Keeping these methods in mind, five ways of implementing prejudice filters have been identified in the current study so as to simplify the extension of existing models to include prejudice. The five ways of implementing prejudice filters, along with their collective definitions as defined by the author for this study, are listed and discussed in the text that follows.

Categorisation: An agent creates various categories that are trusted. If an interaction request does not fall into a trusted category, the agent filters out that interaction in a prejudiced manner. This can also be implemented in reverse, where an agent creates categories that are distrusted and filters out communications that fall into those categories.

Recommendation: Agents that are trusted to make recommendations are known as recommender agents. Implementing prejudice by using recommendation allows a particular agent to trust only other agents that are trusted by the particular agent's trusted recommender agents.

Policy: Policies define the operational environment in which an agent exists. Policies define organisational processes and procedures that guide decision making (http://www.pmostep.com/290.1TerminologyDefinitions.htm). Furthermore, policies contain rules that govern interaction between entities (http://www.microsoft.com/windows2000/techinfo/howitworks/activedirectory/glossary.asp). Policy-based prejudice filters out interactions with agents whose policies differ from those of the agent doing the filtering.

Path: Path-related prejudice allows an agent to refuse an interaction, simply because of the fact that the communication path between two agents passes through a distrusted agent.

Learning: When one is using the learning filter, prejudice is not defined explicitly. An agent relies on 'first impressions' to learn prejudice. If an interaction fails, the agent analyses the interaction's attributes and looks for attributes that make a particular interaction unique from other interactions. These attributes include the organisation that an agent belongs to, the role an agent requested the other to play, the transaction size and the sensitivity of the information handled. These unique attributes are used to create a category that can be used as a prejudice filter.

The above five filters can be incorporated into current trust models to extend their capabilities, while at the same time allowing for these filters to merge with a particular trust model's main philosophy. The implementation of these filters, as well as the advantages and disadvantages of each, is discussed in the sections that follow.

6.4.1 Prejudice categorisation

Categorisation is a simple way to implement prejudice and its concepts. Agents can be grouped into categories depending on their various properties. These categories can then be used to determine the assumed properties of agents that are considered to belong to a particular category. Based on these assumed properties, certain categories can be discriminated against and filtered out even before a lengthy trust evaluation process begins. For example, as illustrated in Figure 6.2, agents can be categorised by the core services they provide and participate in.

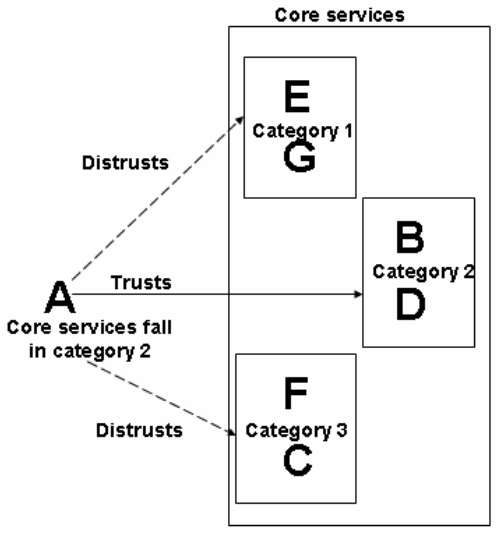

Figure 6.2 Prejudice categorisation

Core services refer to services that define the core of what a business does. For instance, when selling items online, an agent needs to elicit a payment, schedule a shipment and update the status of its inventory (Tenenbaum, Chowdhry & Hughes 1997). These services can be provided by various agents. An agent only needs to trust agents that provide those services that it requires to accomplish its given task. The various services an agent provides and makes use of define the category it belongs to. An agent will then trust only agents that fall into the same category as the agent itself.

Alternate implementation allows agents to keep lists of either trusted or distrusted categories. If trusted categories are recorded, other agents are trusted only if they fall into a trusted category. If distrusted categories are recorded, other agents are trusted only if they do not fall into the distrusted category. Filtering out categories that are distrusted or unknown decreases the possibility that the trust evaluation will result in a distrust value. Three main types of categories that are to be used by prejudice filters have been identified, namely organisation, domain and roles.

6.4.1.1 Organisation

There are various mechanisms for identifying other agents one communicates with. Digital signatures (Yang, Brown & Lewis 2001) vouch for people; IP addresses identify computers; and organisations represent themselves with signed certificates binding together groups of people and IP addresses (Khare & Rifkin 1997). Signed certificates can be used to implement forms of ‘organisation’ prejudice. An agent can implement organisation-based prejudice to filter out organisations that it does not trust. This is accomplished by requesting signed certificates that identify the organisation an agent belongs to. If the organisation is identified as either distrusted or unknown, the interaction is denied. This form of prejudice places agents into organisational categories and the interaction involved is illustrated in Figure 6.3. Agent A trusts Agents E and G, due to the fact that it knows and trusts Organisation 1. Agent A distrusts Agents F and C, since it distrusts Organisation 2. Agent A also distrusts Agents B and D, since it does not know the organisation, if any, with which those two agents are affiliated. Filtering by organisation allows agents to filter out other agents that come from organisations that have not proven themselves to

be trustworthy. This reduces the likelihood that an interaction will fail simply because another agent comes from a distrusted organisation.

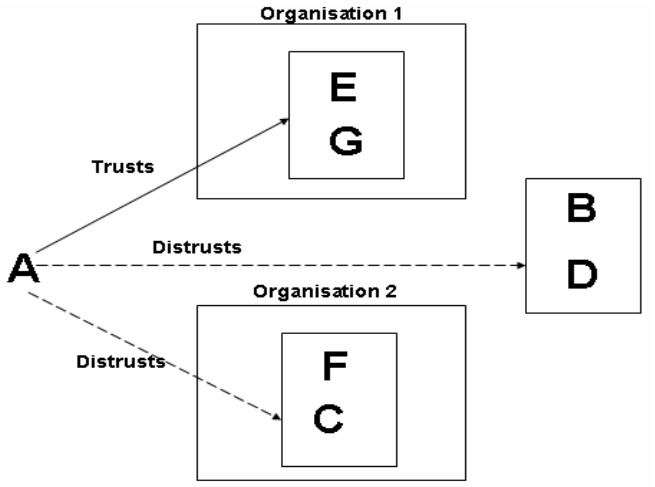

Figure 6.3 Organisational prejudice

6.4.1.2 Roles

Using roles is a form of categorisation that groups agents by roles and assigns to them the privileges associated with specific roles. Should an agent wish to participate in interactions that are outside of the role definition to which it belongs, this interaction is denied. The grouping of agents into roles allows a business agent to define standard actions and limit privileges to a particular role instead of worrying about defining unique access for each individual agent it comes into contact with. Role-based prejudice, as illustrated by Figure 6.4, is implemented by requiring agents to identify the roles that they are permitted to play when they ask to participate in an interaction. When an agent analyses another, it checks the roles that the other agent can play. If the other agent does not possess a role definition that is trusted by the agent doing the analysis, the agent is distrusted and the interaction is denied.

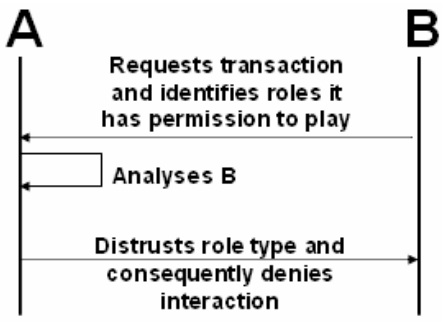

Figure 6.4 Process of establishing role-based prejudice

6.4.1.3 Domain

Prejudice in a domain can be seen as an extension of organisational prejudice and can be implemented in a similar way. However, the key difference here would be that domain prejudice can be implemented within an organisation itself in order to limit the number of interactions an agent needs to deal with. This form of prejudice also allows an agent to filter out domains that may be considered to be unstable or unreliable.

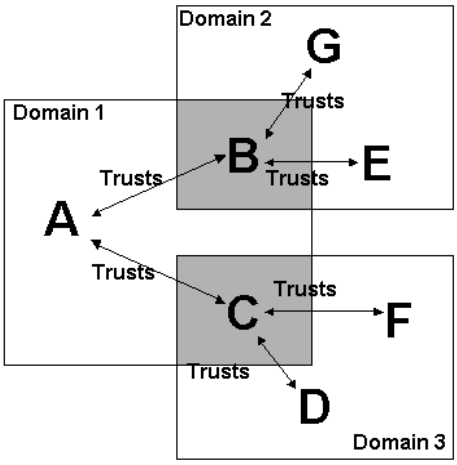

Figure 6.5 Domain-based prejudice

Alternate implementation allows agents to trust only agents within their own defined domain. Some agents are allowed to exist in more than one domain and possess trust definitions for all the domains to which they belong. Should an agent wish to communicate with an agent outside of the domain it belongs to, it uses other agents from the domains it does belong to as intermediaries. The intermediaries define the level of trust given. Limiting the number of agents an agent can interact with also limits the number of agents it needs to evaluate for trust, thus limiting processing load. Such a scenario is illustrated by Figure 6.5. Agent A communicates only with Agents B and C. If Agent G wishes to communicate with Agent A, it does so through Agent B, using Agent B as an intermediary. The principle here is that Agents G and E need to move through Agent B to communicate with Agent A, while Agents F and D need to pass through Agent C. However, if the intermediated interaction fails, communications that pass through the intermediary become distrusted in a prejudiced way. For instance, if Agent A's communication with Agent G passing through Agent B fails, future interactions that are required to pass through

Agent B will also be distrusted, resulting in a situation where Agent E will be discriminated against.

6.4.1.4 Advantages and disadvantages of prejudice categorisation

Categorisation allows for assumptions to be made. It is on the basis of these assumptions that certain categories are prejudiced against. The problem with this approach is that an agent will trust or distrust only categories it knows about.

6.4.2 Prejudice recommendation

Trust models rely heavily on social concepts to control interactions. A way to manage trust often relies on a community of agents. An agent trusts another agent, and it has trust values for the trust it has in the other agent's recommendations (Abdul-Rahman & Hailes 1997). If Agent A trusts Agent B and propagates that trust to Agent C, many trust models allow Agent C also to trust Agent B, provided that Agent C also trusts Agent A, as shown in Figure 6.6.

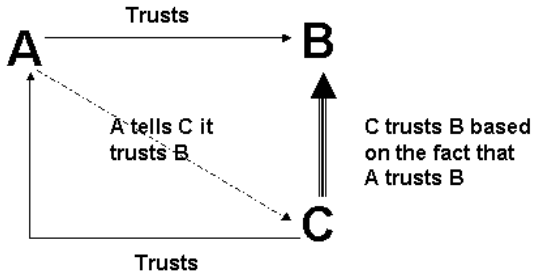

Figure 6.6 Trust and recommendation

Prejudice recommendation requires trusted groups of recommenders. An agent establishes a list of recommenders it trusts. The agent then trusts only other agents that are trusted by its recommenders. This minimises the risk of encountering agents that are untrustworthy. It also reduces the amount of processing that an agent needs to do in order to determine trust values for unknown agents.

Communications between the two agents pass through intermediaries whose purpose it is to control the interaction. During the interaction, intermediaries are considered to be neutral parties that have nothing to gain. Intermediaries can be seen as particular types of recommenders that are

used solely for the collection, grouping, definition and categorisation of trust-related data (Papadopoulou et al. 2001). Intermediaries can incorporate the concept of prejudice by keeping lists of trusted agents. Agents that trust a particular intermediary then have access to this information, which extends the concept of trusted intermediaries such as Certificate Authorities (Siyal & Barkat 2002). Prejudice filters can be implemented by allowing an agent to trust only other agents that are trusted by particular intermediaries. Figure 6.7 illustrates this concept.

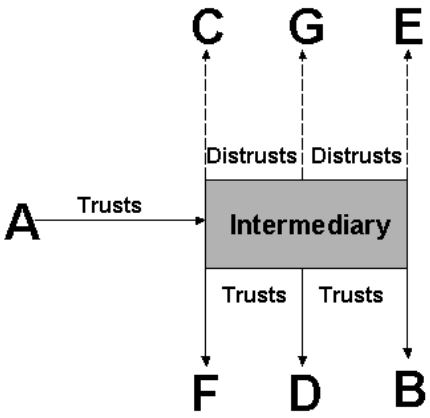

Figure 6.7 Implementing prejudice by means of an intermediary

The intermediary that Agent A trusts, trusts only Agents F, D and B; therefore, Agent A will also trust only Agents F, D and B. The intermediaries possess all the trust-related information and do the trust processing, thus eliminating the need for the agent to process trust. This increases the probability of successful interactions, as an agent communicates with other agents that already have an established level of trust with an intermediary.

6.4.2.1 Advantages and disadvantages of prejudice recommendation

Requiring an agent to trust only agents that are trusted by recommenders that it trusts, ensures that the agent receives information from trusted sources. This form of prejudice, however, limits an agent to trusting only agents that are trusted by its trusted recommenders.

6.4.3 Prejudice policy

Trust models tend to approach the issue of trust from a generic point and often neglect the fact that agents interact with one another according to the policies defined by the environment or

community to which they belong. For instance, a library's policy defines a rule that determines how late book returns should be handled. Books borrowed from institutes that are not libraries do not necessarily abide by this rule. Thus different environments require different policies.

Policies that communities and organisations follow refer to three basic sources of expectation. These include general rules shared by all agents; social rules that the agents belonging to a particular group share; and the organisational rules that are defined and enforced by the organisation within which the negotiating agents interact (Ramchurn et al. 2003). These rules define agents' expectations and include concepts such as privacy policies, the purpose information is used for, encryption policies and the transaction contexts used by agents during interactions.

Since policies define the way in which interactions are handled, agents with policies that differ vastly from one another are more likely to fail in an interaction with one another. Prejudice filters are a good way to avoid policy misunderstandings simply by allowing the agent to disregard other agents with policies that differ vastly from its own. In order to implement this type of prejudice, an agent defines its own policy. When encountering another agent, the agent requests the other agent to define the policy rules that it considers vital and compares them to its own. If the rules that it receives differ vastly from its own, the agent refuses communication with the other agent. This prejudice filter can be used in conjunction with categorisation (whereby an agent creates organisational and domain categories for agents that have policies that differ vastly from its own). Possible implementations include the dividing of agents into world zones. This is possible due to the fact that world zones define varying cultural beliefs, which tend to dominate the business world in which an agent resides. Examples include the differences in cultural values between Asian and Western culture. Asian cultures emphasise the community, while Western cultures tend to put more emphasis on individualism and public reputation (Earley & Gibson 1998).

An agent also needs to know which other agents are trusted by the agents it trusts. This is due to the fact that agents share information with those they trust. For example, as is shown in Figure

6.8, Agent A is in the process of evaluating Agent B to determine whether it should trust Agent B, while also analysing whom Agent B trusts.

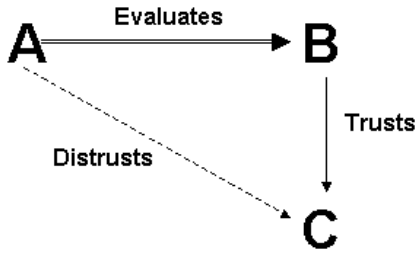

Figure 6.8 Prejudice using policy

Agent A requests Agent B to send a list of the agents trusted by Agent B and subsequently analyses this list, comparing it to its own. Agent A knows that it distrusts Agent C. Agent A sees from the list that Agent B trusts Agent C. Now Agent A will give Agent B less trust, based on this fact. The reason for the decrease in trust is that Agent A may not wish to disclose sensitive information to Agent B, due to the fact that Agent B trusts Agent C and may potentially in the future share this information with Agent C. This interaction results in Agent A's being prejudiced against all agents that trust Agent C, whether the agents that trust Agent C would disclose information gained from Agent A to Agent C or not.

6.4.3.1 Advantages and disadvantages of prejudice policy

Enforcing policy prejudice ensures that agents communicate with other agents that adhere to the same policy constraints as the agent itself. However, this limits the opportunities that an agent is exposed to when interacting with agents from environments that differ vastly from its own.

6.4.4 Prejudice path

Path length is already a form of prejudice filter used by agents in a network. Path length in the context of this research refers to the number of agents that a message has to pass through to reach its designated destination. This is due to the fact that extremely long paths are seen as untrustworthy (Datta et al. 2003). The probability of interaction failure increases with path length. This is not necessarily the case, but this assumption allows an agent to filter out interactions that have a high probability of resulting in a distrust value.

Trust in another agent is strongly influenced by the communication path between two agents. The more secure the path, the safer the transaction and, therefore, the higher the possible level of trust. An agent has the right to refuse communication with another agent, based on the fact that the communication path passes through agents that the agent in question does not trust (Datta et al. 2003). The communication path can be obtained by requiring incoming communication to use the record route option of a datagram such as the one already existing in the IP datagram (Forouzan & Fegan 2003). To ensure that communications always travel along a trusted route, using an option such as the strict source route in IP datagrams can force a communication to take the strictly predefined route. This option discards a communication as soon as it passes through a router that is not defined in the option. Denying interactions that have passed through distrusted nodes decreases the probability that an interaction will fail.

Prejudice and path length are illustrated in Figure 6.9. Under normal network conditions without the inclusion of trust, communication between Agents A and B would pass through Agent C due to the fact that that is the path that contains the fewest nodes. However, due to the fact that either Agent A or Agent B distrusts Agent C, this path cannot be used. Therefore, the path with the next lowest number of nodes between the two desired points of interaction is chosen, provided that both Agent A and Agent B trust all the agents along that path. The path chosen in Figure 6.9 passes through Agents G and D since there is no distrust indicated along that path.

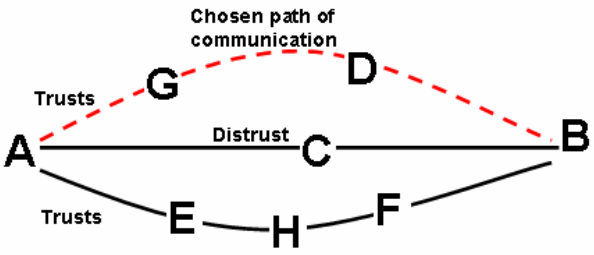

Figure 6.9 Prejudice using path

6.4.4.1 Advantages and disadvantages of prejudice path

Disregarding communications that come along distrusted paths lowers the risk of losing information due to malicious behaviour of distrusted nodes along the path. The disadvantage of

this approach is that agents are unable to communicate with one another if no trusted path between them exists.

6.4.5 Prejudice learning

A more flexible approach to prejudice filters is allowing agents to learn prejudice and define prejudice in their own way. To accomplish this, an agent relies on 'first impressions'. Possible implementation of this concept allows an agent to begin with a basic set of rules that it uses to evaluate the success of an interaction. Initially, the agent interacts with any agent with which it comes into contact, under restricted conditions of trust. Each interaction triggers an analysis process by means of which the agent identifies parameters such as the location of an agent, the organisation an agent belongs to and even factors such as an agent's reputation. These parameters become the characteristics of the particular interaction. If the interaction fails, the parameters are analysed to identify a means of filtering out interactions of a similar nature in the future.

Figure 6.10 demonstrates this by means of a sequence diagram and the use of organisational information. As the figure shows, Agent A refuses any transactions from Organisation 1 simply based on the fact that its first impression of Organisation 1 came from Agent B with whom Agent A's transaction failed. Agent A does not care that interactions with other agents from Organisation 1 might succeed. If the transaction succeeds, a trust relationship is formulated.

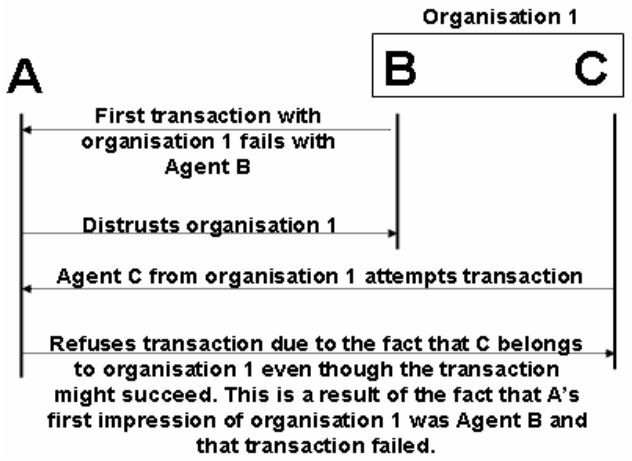

Figure 6.10 Establishing prejudice through first impressions

It is important to note that the first impression is the key one. If a transaction fails at a later stage, it does not terminate the agreement or the relationship between agents, but simply lowers the trust value of the relationship (Sung & Yuan 2001).

6.4.5.1 Advantages and disadvantages of prejudice learning

Learning allows agents to define their own unique forms of prejudice. This is, however, a costly procedure that is process intensive and requires detailed analysis of the circumstances that resulted in an interaction failure.

6.6 Overview of prejudice filters

The concept of prejudice filters expands on the criteria discussed in Chapters 3, 4 and 5 by adding a fifth category to the four existing ones. This fifth category is geared at reducing the network and processing load required to analyse an agent that will likely result in a distrust evaluation. Table 6.1 provides an overview of this category and the criteria associated with it.

Table 6.1 Overview of prejudice filters

CRITERIA	X	ADVANTAGES AND DISADVANTAGES
PREJUDICE		
Categorisation		ADVANTAGE: It is easy to look at the category and category characteristics. DISADVANTAGE: Only trust or distrust constitutes known categories.
Recommendation		ADVANTAGE: Information is received from trusted sources. DISADVANTAGE: This is limited to trusted recommenders.
Policy		ADVANTAGE: Agents with similar policies have similar constraints. DISADVANTAGE: Interactions are limited to similar environments.
Path		ADVANTAGE: There is a lower risk of loss of information due to path problems. DISADVANTAGE: Agents cannot communicate if no trusted path exists.
Learning		ADVANTAGE: Agents can learn their own form of prejudice. DISADVANTAGE: This is process intensive.

6.6 Conclusion

This chapter extended the criteria identified in Chapter 3 and introduced the concept of prejudice. Prejudice allows agents to deny communication with other agents based on similarities to agents with whom an agent has previously had a bad experience. The purpose of prejudice is to decrease the processing load required for analysing agents for whom there is a high likelihood of interactions resulting in a distrust value. Using a prejudice filter will enable an agent to dedicate its processing to interactions that are more likely to result in positive trust establishment, discarding ones that have a high possibility of failure, and to do so before the detailed analysis phase.

Since prejudice relies on negative assumptions, it runs the risk of filtering out communications that would in fact be successful based on these assumptions. This inherent risk is countered by the fact that environments influence agents' behaviour. Even though not all agents from a particular environment may behave in exactly the same way, certain behavioural patterns tend to dominate and can be used to make negative assumptions.

Five main types of prejudice filters have been identified, namely categorisation, recommendation, policy, path and learning. The five prejudice filters are based on current trust model implementations and have advantages, as well as disadvantages. The greatest disadvantage of prejudice filters is the fact that they limit the communications an agent participates in. This disadvantage is countered by the fact that prejudice filters simplify the trust evaluation process and filter out communications that are likely to fail, thus increasing an agent's efficiency in interactions with agents that are in fact trusted. The filters identified have various relationships to one another. These relationships allow some of the prejudice filters to be implemented in conjunction with one another in order to improve their efficiency. Chapter 7 defines and discusses these relationships.

7. Prejudice filter relationships

7.1 Introduction

Prejudice filters have been proposed to filter out agents that have a high likelihood of resulting in a distrust value before the detailed trust evaluation process actually takes place. These prejudice filters also filter out unknown agents that carry a high risk due to the environments they reside in. Each prejudice filter is a unique concept that can be implemented on its own. The prejudice filter chosen for implementation should depend on the way in which a trust model represents and handles trust. Trust models implement varying concepts and even varying combinations of concepts due to the co-dependence of these concepts. Since prejudice filters are based on the concepts that trust models implement, it stands to reason that they also possess co-dependencies. This then raises the question of what co-dependencies exist between prejudice filters that allow more than one of these filters to be implemented by a single trust model.

Chapter 7 addresses this problem by discussing various co-dependencies between prejudice filters in the form of prejudice filter relationships. Section 7.2 illustrates these relationships and each relationship is then discussed in detail. Section 7.3 concludes the chapter.

7.2 Defining interrelationships between filters

The five prejudice filters discussed in Chapter 6 can be organised into a structure that shows the relationships between these filters, as shown in Figure 7.1. The nodes represent the prejudice filters and the arrows represent the relationships between them. The node at the tail of an arrow illustrates the filter that dominates the relationship. The filter at the head of the arrow is incorporated into the working of the dominant filter. Each of the relationships identified is discussed with reference to the dominant filter in the relationship as illustrated by the various labels. The labels refer to the dominant filter of the relationship and the order in which the filters are discussed. L refers to ‘learning’-dominated relationships, P refers to ‘policy’-dominated relationships and R refers to ‘recommendation’-dominated relationships. The relationships identified by the labels are discussed in the sections that follow.

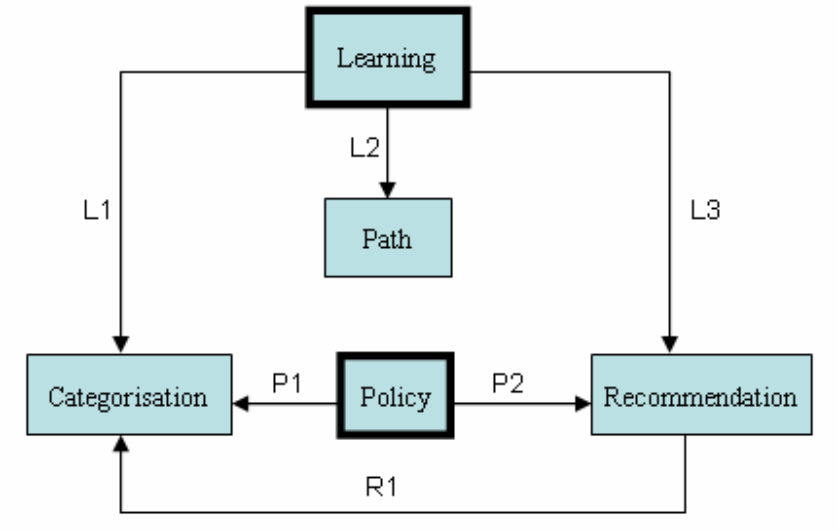

Figure 7.1 Overview of the interrelationships between prejudice filters

7.2.1 Learning-dominated relationships

The nature and success of learning is governed by the nature and variety of information and experience that an agent is exposed to (Bowling & Veloso 2002). Experiences and information are filtered to form templates unique to each agent. Templates are default rules that have been formed by experiences and that are subsequently used to evaluate other, similar experiences. These templates are then used to filter through future experiences of a similar nature to the ones that created the templates. Due to the fact that learning creates various forms of templates (Dasgupta 1997), it is possible to learn various forms of prejudice. These are discussed below.

7.2.1.1 Learning by categorisation (L1)

The process of learning prejudice relies heavily on categorisation. Learning analyses a transaction to determine its unique features. If the transaction fails, the agent uses this analysis process to create a category of failure that can be used in future category-based prejudice decisions. The implementation of this concept relies on allowing an agent to form categories defined by the trust rules that are already in place. For instance, if the trust rules in place require transactions to be analysed in order to determine the environmental factors used by the agents in question, these agents can be categorised by the organisations or domains that define the environments in which the agents reside. Learning may define categories according to the three broad categories already identified, namely roles, organisations and domains. However, categories defined by learning do

not need to belong to one of the generic categories that have already been defined. An agent may be free to create its own categories.

7.2.1.2 Learning path-related prejudice (L2)

Path prejudice relies on determining the path a message has taken and, consequently, it requires an agent to keep track of nodes that are either trusted or distrusted. The learning filter will form a path prejudice category if no other form of prejudice can be determined. This is determined by analysing an interaction. If the characteristics of an interaction identified by the analysis process are known to have resulted in successful interactions before, yet the interaction fails, learning then analyses the path the communications have taken and creates path prejudice filters. In order to create path-related prejudice, the learning process keeps track of distrusted nodes. When checking the path of an interaction that failed unexpectedly, the list of distrusted nodes is consulted and used for reference. If a known distrusted node exists in the path, a category based on the path that passes through the distrusted node is created. If no distrusted node along the path can be identified, the transaction triggers an alert state and the explicit path is stored. Future interactions with the same agent with whom the unexpected failure occurred will trigger a path evaluation process. Gathering path information will allow the agent to compare various paths so as to pinpoint the difference in path between interactions that have failed and interactions that have succeeded. Once a suspect agent along the path has been identified, this agent is added to a distrusted agents list and used to implement path-based prejudice.

7.2.1.3 Learning recommendation prejudice (L3)

Agents in an environment need to be able to learn which agents they can trust as recommenders, as well as to spot the differences in perceptions that exist between their own perceptions of trust and those of the recommenders they trust. Being able to adapt and determine their own trust in other agents is vitally important in dynamic environments. Recommender agents provide vital assistance in these dynamic environments. In the same way that agents need to learn and establish trust in dynamic environments, they also need to establish prejudice. Learning recommendation prejudice requires an agent to test the recommendations it receives from potential recommenders. It requires the agent to test not only for the agents that the recommender trusts, but also for the

agents that the recommender distrusts. This allows an agent to evaluate the similarities between its own perceptions and those of the recommender it trusts. If the semantic differences between its own perceptions and those of its recommender are close, an agent can use the recommender to implement the defined concept of recommendation prejudice and trust only other agents trusted by the recommender it has chosen to trust.

7.2.2 Policy-dominated relationships

Policies define the rules and actions that govern the environment in which agents exist. These rules include procedures for interaction management, failure recovery and access management. Since policies already contain information that defines and restricts a particular environment, they can also contain prejudice-related information. This prejudice-related information can take various forms and can incorporate some of the other prejudice filters defined above, in particular, categorisation, recommendation and path filters.

7.2.2.1 Policy and categorisation (P1)

In order to implement category-based prejudice, an agent needs to define either categories it will trust, ignoring other categories, or categories it will distrust. These categories need to take policy constraints and restrictions into account. It is undesirable to have an agent trust a category that breaches any of the policy constraints defined by a certain environment. When considering policy constraints in certain categories, agents also need to realise that the same category may have different constraints across different environments. For instance, an agent playing the role of a customer in one domain may be given more freedom in that domain than it would have been given playing the same role in a different domain. When agents from different domains communicate, these differences need to be considered.

7.2.2.2 Policy and recommendation (P2)

Just as policy constraints influence categories, they also influence recommendations. When choosing recommender agents, it is important that agents seek out other agents that adhere to similar policy constraints. In this way, the recommender agent's perception will be more closely related to that of the agent seeking a recommendation. Since policies are usually defined by the

environment in which agents reside, this is not a problem when agents and the recommender agents that are trusted reside in the same environment. However, when an agent receives a recommendation outside of its known environment, it needs to consider the policy constraints of the recommender.

Policy constraints can become a problem in dynamic environments such as *ad hoc* networks, where agents come from different environments with differing policies and form a new, dynamic and changing environment. In such environments, policy constraints become particularly important and need to be agreed to by all the agents that want to interact with one another. A new agent coming into such an environment needs to seek recommendations as well as information about the existing policy restrictions of the current environment.

7.2.3 Recommendation-dominated relationships

Recommender agents give recommendations based on their own knowledge about another agent's trustworthiness. Hence, recommendations depend on perceptions. Agents' perceptions are created within given contexts. Consequently, good trust models tend to possess more than one trust value for a single agent, based on context. In situations where there is more than one trust value for an agent, some way to identify and differentiate between these values is required. Trust models therefore rely on categorisation.

7.2.3.1 Recommendation and categorisation (R1)

Because trust models rely on categorisation to differentiate between various contexts of trust, it becomes important for recommendations to reflect this. Recommendations should be given for the context in which the agent seeking a recommendation wishes to interact. In the same way that agents need to know the context of the recommendation they are receiving, they also need to know their own trust in recommenders in given contexts. This is particularly important in dynamic environments, because perceptions may differ across agents. Since the context of an interaction changes the trust given, an agent needs to know that it trusts a recommender to give recommendations for trust that is influenced by specific contexts.

7.3 Conclusion

Prejudice filters are based on concepts used by trust models that can be implemented together. This implies that prejudice filters can be co-dependent, just as the concepts on which they are based can be co-dependent. When they are implemented in this way, the co-dependence among them can be illustrated as various relationships. Chapter 7 has identified and discussed these relationships with reference to the dominant filter in each particular relationship. Three learning-dominated relationships, two policy-dominated relationships and one recommendation-dominated relationship were discussed. The various relationships that have been identified influence the way in which the various prejudice filters function. Prejudice filters and the other four categories of trust model criteria that have been defined up to this point need to be tested against a trust model to demonstrate their efficiency. Chapter 8 provides a detailed evaluation of Abdul-Rahman and Hailes's (2000) trust reputation model, using the trust model criteria defined up to this point.

PART 4

TRUST MODEL ANALYSIS USING THE DEFINED TRUST MODEL CRITERIA

8. Analysis of a trust reputation model

8.1 Introduction

Several trust model criteria have been defined as a guideline for trust model analysis. Analysis of trust models assists in the identification of trust models' specific characteristics. These criteria have further been grouped into four main categories: trust representation, initial trust, trust updates and trust evaluation. The identification of the characteristics of a trust model is helpful in the identification of the best environment for a particular trust model. Ideally, a trust model's characteristics should match those of the environment it is to be implemented in. Part 1 of this dissertation identified and discussed these main categories and the criteria associated with them, while Part 2 extended them to include the concept of prejudice filters for trust models.

Chapter 8 illustrates a sample analysis by using the four main categories of criteria and Abdul-Rahman and Hailes's (2000) trust reputation model. Furthermore, it identifies the simplest way in which to extend the chosen trust model to include prejudice filters, as discussed in Chapters 6 and 7. Section 8.2 provides a detailed analysis of Abdul-Rahman and Hailes's (2000) trust reputation model using the four main categories of criteria identified in Chapter 3 and the additional category, the prejudice filters, as defined in Chapters 6 and 7. Section 8.3 concludes this chapter.

8.2 Abdul-Rahman and Hailes's trust reputation model

A wide and varied range of trust models has been proposed, each implementing a number of different concepts. As was mentioned in Chapter 2, these concepts include recommendation, reputation, observation, institution and experience (Jøsang 1997; Xiong & Liu 2003; Esfandiari & Chandresekharan 2001; Papadopoulou et al. 2001; Abdul-Rahman & Hailes 1997). Thus there are many trust models to choose from. The proposed trust model criteria are intended to be used to analyse trust models for implementation purposes.

Several trust models were considered for this analysis process. These included Aberer and Despotovic's (2001) P-Grid system for managing trust in a peer-2-peer environment; Luo et al.'s (2002) localised trust model for *ad hoc* networks; Mui, Mohtashemi and Halberstadt's (2002)

computational model for trust and reputation; and Abdul-Rahman and Hailes's (2000) trust reputation model. Aberer and Despotovic's (2001) P-grid system is a decentralised approach that can be implemented in environments that do not possess a central source of information. Their approach relies on analysing interactions agents have already participated in and obtaining a reputation value. This value is then used to analyse the probability that an agent will cheat in an interaction. Luo et al.'s (2002) localised trust model for ad hoc networks provides a distributed authentication service for ad hoc networks. Trust is used to determine which authorities are trusted to authenticate others. Mui, Mohtashemi and Halberstadt's (2002) computational model for trust and reputation relies on reciprocation to establish a reputation and influence trust. Abdul-Rahman and Hailes's (2000) trust reputation model allows peers to share recommendations. Agents analyse the recommendations that are received to derive a reputation for other agents.

Reputation-based models dominate current trust model implementations. Consequently, most of the trust models considered for detailed analysis in this chapter are reputation-based. All the models considered are cited frequently and have obviously had an impact on this field of research. Of all the models considered, the model chosen for detailed analysis is Abdul-Rahman and Hailes's (2000) trust reputation model, because it is a reputation-based model and reflects the dominant trend in the types of models currently proposed and implemented. The model is cited by several experts who have made a worthy contribution to research into trust. These experts include Aberer and Despotovic (2001), Mui, Mohtashemi and Halberstadt (2002), Xiong and Liu (2003) and Yu and Singh (2002). Furthermore, Abdul-Rahman and Hailes's (2000) trust reputation model is well structured; it implements the concepts of direct trust, recommendation and reputation.

8.2.1 Overview of Abdul-Rahman and Hailes's trust reputation model

The model of Abdul-Rahman and Hailes incorporates several of the concepts discussed in the four categories of criteria defined in Chapters 3, 4 and 5. A subset of direct experiences and recommendations is summarised to obtain a trust value. Experiences are recorded in two separate sets so as to be able to differentiate between those that are a result of direct trust and those that are linked to recommendation. These trust values are separated by the particular context for which they are formed, resulting in several trust values for a single agent, depending on the

context. In order to obtain a direct trust value for a particular agent in a particular context, this model relies on a basic system of counters. For every agent within a specific context, an agent has four counters. The four counters this model uses are counters for *very good*, *good*, *bad* and *very bad*, and they represent various trust levels, namely very trustworthy, trustworthy, untrustworthy and very untrustworthy. The counters are incremented as a result of the interactions in which an agent participates. The trust model executes a `MAX` function on these four counters that returns the counter that has the highest value for that particular agent in a specific context. The trust counter with the highest value indicates the trust level that is to be assigned.

Trust values that agents receive as recommendations rely on a semantic closeness value for analysis. This semantic closeness value determines how closely a recommender agent's perception of trust with another agent resembles that of the agent seeking a recommendation. The value calculated from semantic closeness is used to adjust the trust values received as recommendations from other agents. An agent obtains this adjustment value by comparing its own trust evaluations to that of other agents, for a particular context, with the trust evaluations of the recommender agent for the same agents and in the same context. The difference in these values is used as the adjustment value. For example, Agent A is looking for an adjustment value for Recommender B. Agent A knows that the trust value it has for Agent C is 't' (trustworthy) in Context X. Agent A receives information from Agent B that Agent B's trust value for Agent C in Context X is 'vt' (very trustworthy). In other words, Agent A's value for Agent C, in the same context as Agent B's value for Agent C, is one level lower than Agent B's. The adjustment value that Agent A obtains is then -1. This value is used to lower the trust value of all recommendations coming from Agent B by 1 so that it will resemble Agent A's own perceptions more closely.

8.2.2 Evaluation of Abdul-Rahman and Hailes's trust reputation model, using the trust model criteria identified in this study

8.2.2.1 Trust representation

Trust representation refers to the way in which a trust model chooses to represent trust-related data. Abdul-Rahman and Hailes's (2000) approach to trust representation can be summarised as illustrated in Table 8.1. The first column of the table lists the criteria that belong to the trust

category under discussion. In this case, it is the category of trust representation. The 'X' values in the second column of the table represent the criteria in the category of trust representation that are implemented by the trust model. The last column of the table lists the advantages and disadvantages of a particular criterion. All the tables that follow are to be interpreted in the same way.

Table 8.1 Analysis of trust representation based on Abdul-Rahman and Hailes's(2000) trust reputation model

CRITERIA	X	ADVANTAGES AND DISADVANTAGES
TRUST REPRESENTATION		
Trust outlook		
Blindly positive		
Blindly negative		
Slow negative, fast positive		
Slow positive, fast negative		
Balanced slow	**X**	ADVANTAGE: The process balances both positive and negative experiences. Due to the slow changes in values, agent is exposed to more interactions and hence has more opportunities before reaching a state of distrust. DISADVANTAGE: Agent is exposed to greater risk when the values decrease slowly.
Balanced fast	**X**	ADVANTAGE: The approach balances both positive and negative experiences. Due to rapid changes in trust, agent is exposed to less risk when trust values are decreasing. DISADVANTAGE: Agent is exposed to fewer opportunities when values decrease quickly.
Passionate vs. rational		
Passionate	**X**	ADVANTAGE: A dynamic, more adaptive approach. DISADVANTAGE: Exposes agents to higher risk and has higher processing overheads.
Rational		
Centralised vs. decentralised		
Centralised		
Decentralised	**X**	ADVANTAGE: A dynamic approach that includes differing perceptions. DISADVANTAGE: Every agent needs to keep its own trust-based information and requires more processing and storage capabilities.
Trust vs. distrust		
Trust only		
Distrust	**X**	ADVANTAGE: Allows for differentiation between low trust

		values as a result of lack of experience or as a result of negative experiences. DISADVANTAGE: Distrust is easier to prove than trust and can lead to false distrust values.
Scalability		
Scalable		
Not scalable	**X**	ADVANTAGE: A system that is not scalable requires low maintenance. DISADVANTAGE: A system that is not scalable cannot be adapted for environments that change.

The attributes chosen by Abdul-Rahman and Hailes (2000) and the ways in which they conform to each identified trust representation criteria category are discussed in the text that follows.

Trust outlook: The agents in this trust model have a rather pragmatic approach. They keep a record of both positive and negative experiences in order to balance them in given contexts. Since the agents have counters for experiences ranging from very good to very bad, both positive and negative experiences are recorded. When the values of the counters are close to one another, then the approach is balanced fast, and a single experience can cause a drastic change in trust. If the different counters vary drastically, the approach becomes more of a balanced slow approach requiring several experiences to change the trust value. The flaw in this approach is that if the *very good* and *very bad* counters have a high number of experiences stored and differ only by a small degree, the trust decisions that are made become flawed. For instance, if the *very good* counter has a value of 100 001 and the *very bad* counter has a value of 100 000, then the trust model allows the agent to trust the other agent in question based simply on the fact that the good experiences outweigh the bad by a minute percentage.

Passionate versus rational: This model takes a passionate approach, which incorporates a mechanism of social control. The social concept this model is built on is that of reputation and recommendation, and it relies on the 'opinions' of other agents to make trust-based decisions. Relying on the opinions of other agents carries the risk of receiving false information. Abdul-Rahman and Hailes's model allows agents to adjust their trust values for recommender agents. However, evaluating the difference between an agent's and a recommender's opinion is process intensive.

Centralised versus decentralised: This model is decentralised. It is proposed for open distributed systems and there is no central agent that gathers all the information. Each agent holds its own evaluations of others and is even allowed to adjust the trust values it has received as recommendations. This adjustment process complicates the trust evaluation process and requires agents to dedicate more time and processing to trust evaluation.

Trust versus distrust: Since the trust model includes a counter for bad experiences, it offers a means of differentiating between low trust values due to a lack of information and low values due to a high number of bad experiences. The model only needs to check the values of the bad counters to see if a distrust value is not a result of lack of knowledge. However, this information is never checked by the model when it is used to make a trust-based decision. The model is only interested in the counter returned by the `MAX` function and it does not check to see how many experiences actually formed the values in question. This results in a very narrow and limited view of trust as a whole.

Scalability: Although it does not require a lot of space to store information about a single agent in a single context, this model is not scalable to systems that contain a large number of agents and possible contexts. The model does not limit the number of agents or contexts that are stored by agents. Thus agents are allowed to store information for an unlimited number of agents as well as an unlimited number of contexts. If the number of agents and contexts encountered by an agent is large, the agent is required to store large amounts of data. Furthermore, old information is not purged from the system, which leads to a waste of space. Agents store additional trust structures for every agent and context pair they encounter and never rid the system of information that has not been used in a long time. This situation is further complicated by the continual addition of adjustment values to the recommendation set when interacting as a result of a recommendation.

8.2.2.2 Initial trust

Initial trust looks at how a trust model obtains initial trust values for an interaction. Abdul-Rahman and Hailes's (2000) approach to establishing initial trust values is summarised in Table 8.2.

Table 8.2 Analysis of initial trust using Abdul-Rahman and Hailes's (2000) trust reputation model

CRITERIA	X	ADVANTAGES AND DISADVANTAGES
INITIAL TRUST		
Direct approach		
Experience	**X**	*ADVANTAGE: An agent reaches a trust value based on its own perceptions and judgements.* *DISADVANTAGE: When an agent has not had an experience with another, it also does not possess any trust information about the other agent and exposes itself to risk in order to get this information.*
Observation		
Indirect approach		
Recommendation		
Intermediaries (Centralised)		
Friend agents (Decentralised)	**X**	ADVANTAGE: Agents are allowed to have different perceptions of trust and use these perceptions to choose trusted recommender agents. DISADVANTAGE: Agents need to establish trusted friends (recommenders).
Reputation		
Sources		
Witness reputation		
Certified reputation		
Institutional reputation		
Belief		
Local (decentralised)		
Global (centralised)		
Delegation		
Collaboration		
Propagation		

As illustrated in Table 8.2, Abdul-Rahman and Hailes (2000) chose a fairly simple approach to initialising trust, namely experience and a decentralised approach towards recommendation. The mechanisms and reasons for the given classification are discussed below.

Initial trust: This model uses both the direct and indirect methods of establishing trust. When it receives a direct communication from another agent, the model checks to see whether or not it

possesses a trust value for that agent. If it does not possess a trust value for the other agent, it assigns an equally good and bad level of trust to the agent. This allows the interaction to proceed under conditions of uncertainty. In other words, initially, when all the counters are zero, the assigned trust level for a direct interaction is equally good and bad. The result of the interaction is evaluated and used to increment the appropriate counter.

The model relies on recommendation to establish an indirect trust value. Each agent possesses a recommender experience set. This experience set is used to analyse recommendations that are received and to obtain an adjustment value for a particular recommendation coming from a particular recommender. The adjustment value is used to adjust a recommendation to represent a particular agent's perception more closely.

8.2.2.3 Trust update

Trust updates look at how trust is updated by a trust model. This is usually done in the same way as trust initialisation. Abdul-Rahman and Hailes's (2000) trust reputation model implements trust updates in the same way as trust initialisation, as is illustrated in Table 8.3.

Table 8.3 Analysis of trust updates using Abdul-Rahman and Hailes's (2000) trust reputation model

CRITERIA	X	ADVANTAGES AND DISADVANTAGES
INITIAL TRUST		
Direct approach		
Experience	**X**	*ADVANTAGE: An agent reaches a trust value based on its own perceptions and judgements.* *DISADVANTAGE: When an agent has not had an experience with another, it also does not possess any trust information about the other agent and exposes itself to risk in order to get this information.*
Observation		
Indirect approach		
Recommendation		
Intermediaries (Centralised)		
Friend agents (Decentralised)	**X**	ADVANTAGE: Agents are allowed to have different perceptions of trust and use these perceptions to choose trusted recommender agents. DISADVANTAGE: Agents need to establish trusted friends

		(recommenders).
Reputation		
Sources		
Witness reputation		
Certified reputation		
Institutional reputation		
Belief		
Local (decentralised)		
Global (centralised)		
Delegation		
Collaboration		
Propagation		
Additional considerations		
Feedback	**X**	ADVANTAGE: Agents are able to use feedback to change a trust value after an interaction. DISADVANTAGE: Processing power is required to analyse feedback.
Time		

Abdul-Rahman and Hailes (2000) use the same mechanisms to update trust as to initialise it. The text below looks at how this is accomplished and the mechanisms that accomplish this.

Trust update: Abdul-Rahman and Hailes's (2000) model stores two main experience sets: direct experiences and experiences that resulted from recommendations. When participating in a direct experience, trust updates are relatively simple and only the direct experience set counts. When participating in an interaction as a result of a recommendation, trust updates become more complicated. The agent updates its own direct experience set, as well as the recommender experience set. Updating the recommender experience set requires calculating the difference between the recommendation received and the result of the actual direct experience, inserting the resulting adjustment value into the recommendation experience set. This adjustment value is analysed for future adjustments coming from a particular recommender agent. Although trust values do change over time, Abdul-Rahman and Hailes's model does not actively consider the impact of time on trust. Agents only update trust after direct interactions with other agents. This means that if an agent does not interact with another agent for a long time, its trust value for that

agent does not change. This is undesirable, since the other agent and the environment would have changed over time. Long periods without direct interaction with another agent do not affect the level of trust. Abdul-Rahman and Hailes (2000)do not define the way in which the agents reach a specific result and the feedback that is used.

8.2.2.4 Trust evaluation

Trust evaluation explores how an agent evaluates trust-related data to limit interaction and eventually to update trust. The approach of Abdul-Rahman and Hailes's (2000) trust reputation model towards trust evaluation is summarised in Table 8.4.

Table 8.4 Analysis of trust evaluation using Abdul-Rahman and Hailes's (2000) trust reputation model

CRITERIA	X	ADVANTAGES AND DISADVANTAGES
TRUST EVALUATION		
Categorisation		
Reputation		
Stereotyping	**X**	ADVANTAGE: Agents make assumptions according to category. DISADVANTAGE: The categories are limited to those that have been defined or by the rules used to define them.
Context evaluation		
Takes context into consideration	**X**	ADVANTAGE: Evaluating context results in a more accurate trust evaluation. DISADVANTAGE: Additional processing is required to analyse context.
Does not take context into consideration		
Risk		
Fallback mechanism		
Constraints	**X**	ADVANTAGE: Constraints control the risk associated with an interaction. DISADVANTAGE: Constraints limit the interaction.
Risk evaluation during update	**X**	ADVANTAGE: This allows for updating trust information after each interaction and a more accurate trust evaluation. DISADVANTAGE: It requires more processing.
Dynamic vs. approximate evaluation		
Dynamic	**X**	ADVANTAGE: It allows for changes to influence trust level.

		DISADVANTAGE: One needs more processing power.
Approximate		

As can be seen in Table 8.4, Abdul-Rahman and Hailes's (2000) approach to trust updates is rather simplistic. The specific classification details of the trust model are discussed below.

Categorisation: The type of categorisation this model uses is stereotype-based. The specific categories are uniquely defined and they depend on context. Agents are not only identified, but interactions with them are categorised by the context in which they occur. Agents may be trusted in some contexts and distrusted in others. The contexts thus form the category of trust in which an agent can be trusted. This form of categorisation results in the formation of multiple data sets for a single agent. Evaluation of a particular agent is a simple lookup to match the agent identified and the context. This approach allows agents to record an indefinite number of agent and context pairs, which leads to the problem of scalability, as already discussed.

Context: This model specifically takes context into consideration when dealing with other agents. In fact, context is seen as so important that the same agent has varying trust values in different contexts. The context is left undefined, which allows anyone who wishes to implement this model to define the context according to the environment in which this model is expected to be implemented. Since context is taken into consideration, the trust evaluation process becomes more process-intensive, requiring not only the agent, but also the context to be identified.

Risk: This model has no explicit consideration of risk, but it contains several implicit risk management mechanisms. It is explicitly defined as a reputation-based model. By definition, such models usually rely on recommendations to determine a trust value for another agent. The method inherently carries the risk of receiving false information from another agent. This risk is addressed by allowing agents to incorporate both direct and indirect information. Agents are also required to analyse the worth of another agent's recommendations by comparing a decision the recommender has made about another agent in a given context with the decision the agent itself has made about the same other agent in the same context. The result of this analysis is an adjustment value that an agent uses to adjust all recommendations coming from that specific recommender. An interaction is analysed at the end to see the difference in direct experience

from the recommendation given. This difference is stored as an adjustment value for the recommender agent. Another way of handling risk is to allow an agent to assign an uncertainty trust level if there is not enough information, or if information the agent has gathered is contradictory. Although mechanisms to handle risk are included, these mechanisms contain flaws. Uncertainty is assigned only if multiple counters have the same value, and not when these counters differ by a small percentage. Furthermore, as the number of recommendations received from agents increases, so does the number of adjustment values that are stored. The model does not limit the number of adjustment values stored. Increasing numbers of adjustment values have an impact on both the space and the processing needed. Space becomes scarce and processing becomes more intensive with large numbers of adjustment values.

Dynamic versus approximate evaluation: This is a dynamic approach that allows an agent to record a set of experiences that it later uses to determine a trust value. Experiences are updated and the updates influence further trust evaluations for agents in specific contexts. This carries the disadvantage that all dynamic approaches require intensive trust processing.

8.2.3 Identified strengths and weaknesses of Abdul-Rahman and Hailes's trust reputation model

Using the trust model criteria to analyse Abdul-Rahman and Hailes's (2000) trust reputation model has resulted in the identification of several strengths and weaknesses. Abdul-Rahman and Hailes's trust reputation model is well structured, allowing agents to form trust both through direct and through indirect means. This model aims to give an accurate evaluation of trust and updates trust dynamically. Context is also taken into consideration when trust is evaluated, and this results in trust values that are influenced by the context in which the interactions occur. Recommendations are used to establish a reputation for other agents. Agents are allowed to adjust the recommendations they receive to resemble their own perceptions more closely, and this results in a reputation value that is influenced by the agent seeking recommendations and that is more likely to be accurate from the agent's own perspective. This is a more personal approach towards reputation, where an agent knows the source of the recommendations that is used to form reputation and is allowed to influence the recommendations to suit its own perceptions.

Although Abdul-Rahman and Hailes take into consideration the dynamic nature of trust, the situation nature of trust (context) and the fact that perception influences trust, their trust reputation model contains a few flaws. Regardless of the values of the individual counters, Abdul-Rahman and Hailes's trust reputation model allows an agent to make a trust decision based on the maximum counter value. This is an inaccurate approximation if the difference in the counters that indicate trust and those that indicate distrust is a rather small number such as 1. A second flaw identified in the model is that it is not very scalable, due to the fact that no limits are placed on the potentially vast quantities of data the model wishes to store and analyse in order to determine recommender-based trust. The model also requires agents to interact with one another even if no trust information is known. This occurs under conditions of uncertainty and can have both a positive and negative impact. The positive impact is that the trust model allows for new opportunities for interaction, and the negative impact is that it exposes agents to risk.

Overall, Abdul-Rahman and Hailes's (2000) trust reputation model is a good model to implement in small, fairly stable environments, where the expected contexts in which agents are expected to interact are clearly defined. However, the first flaw identified in this model is a critical one and it makes this model undesirable for environments where trust changes intermittently.

8.2.4 Extending Abdul-Rahman and Hailes's trust reputation model by using prejudice filters

Prejudice filters were proposed to increase the number of successful interactions that are participated in and to lower the network traffic required to reach a trust evaluation where an agent rejects an interaction. The simplest way of including prejudice in Abdul-Rahman and Hailes's (2000) reputation trust model is through recommendation, as illustrated in Table 8.5.

Table 8.5 Prejudice filter extension using Abdul-Rahman and Hailes's (2000) trust reputation model

CRITERIA	X	ADVANTAGES AND DISADVANTAGES
PREJUDICE		
Categorisation		
Recommendation	**X**	ADVANTAGE: Information is received from trusted sources. DISADVANTAGE: This is limited to trusted recommenders.

Policy		
Path		
Learning		

The recommendation prejudice filter was chosen because of the fact that Abdul-Rahman and Hailes's (2000) trust reputation model relies on a form of recommendation to initialise and update trust. The specific reasoning for choosing this filter and the impact the filter is expected to have on the model are discussed below.

Recommendation: Currently, the trust model allows an agent to accept recommendations from any agent that wishes to send a recommendation. Each recommender is analysed so that the agent receiving the recommendations can get an adjustment value for each recommendation it receives. When several recommendations for the same agent have been received, the agent assigns weights to each recommendation according to the adjustment values it possesses for the recommender agent that sent the recommendation. The agent initialises a counter for the four possible outcomes of each recommendation. These outcomes refer to the four possible trust values (vt, t, u, vu) that a recommender can possibly send. The agent adjusts each recommendation it has received according to the adjustment value it possesses for the recommender agent that sent the recommendation. The adjusted recommendations are then analysed to increment the appropriate counters. This is done by using the values of the weights that were assigned to each recommendation. The counter with the highest value determines the trust assigned. The weights associated with the adjustment values defined by Abdul-Rahman and Hailes are illustrated in Table 8.6.

Table 8.6 Table of weights assigned to adjustment values as defined by Abdul-Rahman and Hailes

Adjustment value	0	1	2	3	unknown
Weight	9	5	3	1	0

From Table 8.6 it is clear that recommendations that require no adjustment are assigned the most weight, while the recommendations that require the largest adjustment are assigned the smallest weight. These weights influence the final trust value that is assigned from multiple recommendations and give preference to recommendations that require no adjustment. For instance, in a scenario where Agent A wants recommendations for Agent B and receives multiple

recommendations from Agents C, D and E, Agent A will analyse the adjustments it requires to make to Agents C, D and E's recommendations. If the adjustments it is required to make are 0, 3 and 1, Agent A will assign the weights 9, 1 and 5 to each agent respectively. Once the weights have been assigned, Agent A adjusts the recommendations of each agent. If it is assumed that all the agents' recommendations were very trustworthy (vt), then they will be adjusted to very trustworthy (vt) for Agent C, very untrustworthy (vu) for Agent D and trustworthy (t) for Agent E. These adjusted values are subsequently used to increment the counters associated with the values. In the example, the counter values will be 9, 5, 0 and 1 for vg, g, u and vu respectively.

This method of merging recommendations can lead to undesirable trust evaluations. An agent seeks recommendations that most closely represent its own perceptions. The weights allow agents to attach more importance to recommendations that mirror their own perceptions. However, if only a few recommenders that require small adjustments and a large number of recommenders that require large adjustments give recommendations, then the counter values will lean towards the opinions of the many recommenders instead of the few more reliable ones. Incorporating recommendation prejudice addresses this problem by allowing agents to trust only recommender agents whose perceptions are close to their own, discarding recommendations from recommenders that require large adjustment values.

8.3 Conclusion

This chapter illustrated an application of the trust model criteria identified in Chapters 3, 4 and 5 by means of a sample analysis. The model chosen for detailed analysis was Abdul-Rahman and Hailes's (2000) trust reputation model. The trust model criteria did not only succeed in identifying the type of environment for which the trust model has been proposed, but also tested the trust model's inherent strengths and weaknesses. Approaching the analysis of Abdul-Rahman and Hailes's trust reputation model in a methodical way by using the trust model criteria as a guideline helped to identify a critical flaw in the trust model. This critical flaw is the way in which the four trust counters are evaluated.

Chapter 8 also investigated the way in which prejudice filters can be included into the trust model that was analysed. Abdul-Rahman and Hailes's (2000) trust reputation model can include

prejudice filters using recommendation. This analysis and extension illustrated the way in which the trust model criteria can be used to analyse a reputation-based model.

In order to test the categories and associated criteria more fully, further applications are required. Chapter 9 analyses two more case studies and uses the first category of the trust model criteria defined earlier in this study, namely trust representation.

9. Example analysis using trust representation

9.1 Introduction

Chapter 8 presented a detailed analysis of Abdul-Rahman and Hailes's (2000) trust reputation model. The proposed trust model criteria were used to identify the properties of Abdul-Rahman and Hailes's model, as well as the environment that the model is best suited for. This served to prove the concept for one case study. However, further examples are required to be able to consolidate the concept across various types of trust models. The same analysis as that done in Chapter 8 can be done for other trust models. The current chapter expands on Chapter 8 and looks at a simplified analysis of another trust model. Section 9.2 takes two categories of the trust model criteria defined and illustrates an analysis of an alternative model using the criteria categories of trust representation and prejudice filters.

As various trust models have been defined, trust-based implementations that make use of these trust models have also been defined. Trust-based implementations work in conjunction with trust models to achieve a goal and, consequently, rely on specific types of trust models to operate. One such trust-based implementation is Dash, Ramchurn and Jennings's (2004) trust-based mechanism design. The trust model criteria can also be used to analyse these trust-based implementations so as to identify the strengths and weaknesses of the type of trust model that these trust-based implementations rely on. Section 9.3 demonstrates a sample analysis of Dash et al.'s implementation with regard to trust representation. The impact of prejudice filters on this implementation is also considered. Trust representation was chosen because it influences the rest of the trust model in that it illustrates how the environment in which a trust model exists requires particular characteristics from a trust model. Prejudice filters were chosen because they are a new concept introduced in this study and further investigation is needed into the way in which these filters influence trust models.

9.2 Other sample evaluations

The previous chapter provided a full sample analysis by using the trust model criteria defined earlier in this study. In order to expand on the sample analysis already provided and to test the

trust model criteria against different implementations, this chapter provides a partial analysis using two case studies. The criteria that influence trust representation have been chosen because they greatly influence the core of trust models' implementation. Furthermore, this partial analysis includes an exploration of the best ways in which to include prejudice filters into the trust model or trust-based implementation.

The first subsection looks at Schillo, Funk and Rovatsos's (2000) socionic trust model. This model merges the fields of artificial intelligence and trust models that allow artificial agents to rely on trust. Schillo et al.'s trust model was chosen for analysis because it implements concepts that are different from those implemented by Abdul-Rahman and Hailes's (2000) model (already analysed in the previous chapter). Schillo et al.'s model implements trust through a process of observation as opposed to recommendation and reputation. Furthermore, this model makes an important contribution to the research field of trust models by merging the fields of trust and artificial intelligence. The model is frequently cited and is recognised by experts such as Yu and Singh (2002), Ramchurn, Huynh and Jennings (2004), as well as Mui, Halberstadt and Mohtashemi (2002).

The second subsection looks at a trust-based implementation that relies on a particular type of trust model in order to operate. The chosen trust-based implementation is Dash et al.'s (2004) trust-based mechanism design. Dash et al. (2004:748) define mechanism design as follows: 'Mechanism design (MD) is the field of microeconomics that studies how to devise systems such that the interactions between strategic, autonomous and rational agents lead to outcomes that have socially desirable global properties.' Microeconomics refers to the study of economics in terms of individual parts. This includes studies on the distribution of goods and services, the way in which prices are determined and production factors (http://www.mcwdn.org/ECONOMICS/EcoGlossary.html). Trust-based mechanism design merges mechanism design with the principle of trust so that agents take the degree of trust into consideration when determining allocations of system counterparts. The properties of the trust model on which Dash et al.'s (2004) trust-based mechanism design relies, affect the trust values used by this implementation. Consequently they have an impact on the implementation itself. It is this impact that makes analysis of the trust models used by this implementation important. A trust-based implementation was chosen for

analysis to illustrate that the trust model criteria can also be used to analyse implementations that use trust models. The criteria are used to analyse the type of trust model that the trust-based implementation relies on. Dash et al.'s trust-based mechanism design was chosen especially because it requires the trust model on which it relies to provide rational approaches to trust, whereas the other trust models analysed rely on passionate approaches to trust.

9.2.1 Schillo et al.'s socionic trust model

Schillo et al. (2000) define a socionic approach to trust modelling. Socionics is an effort initiated by the German sociologist Thomas Malsch (Malsch et al. 1996) to merge research from distributed artificial intelligence and from sociology (Schillo et al. 2000). Distributed artificial intelligence is a subfield of artificial intelligence that focuses on solving complex problems by dividing the problem into smaller, more manageable parts (http://framework.v2.nl/archive/archive/node/text/default.xslt/nodenr-156647). Schillo et al.'s (2000) socionic model has been proposed as a means to find deceitful agents in an environment. For this purpose, Schillo et al. (2000) defined an algorithm and data structure known as TrustNet, which can be used to evaluate trust.

Schillo et al. (2000) approach trust by modelling an algorithm that achieves partner selection for agents who wish to play the so-called prisoner's dilemma game, focusing on the semantics of the partner selection process. Agents in the environment are required to declare their intentions during the partner selection process and are also allowed to lie about their intentions. The rules of the prisoner's dilemma game can be found in the following literature identified by Schillo et al. (2000): Luce and Raiffa (1957) and Axelrod (1984). Agents receive witness information from other agents about the results of this game and integrate this information into their own TrustNet. Schillo et al. (2000) define TrustNet as a graph containing a series of nodes and connections between these nodes that represent agents' trust levels and altruisms. Each agent has its own unique TrustNet for the environment and the agents it knows about. Agents may also receive witness information from malicious agents. If an agent has limited knowledge about a particular witness, the agent seeks witness information about the witness. This level of trust influences the trust which the agent attributes to the witness's reports.

Witness reports along with direct experiences and observations are merged to form a single trust value between 0 and 1. This trust value indicates the probability of a successful interaction with a particular agent. The results of interactions are compared to the initial intentions that an agent declared and used to test for benevolence. These results are then announced to other agents in an agent's neighbourhood, thus creating small communities of agents with knowledge about one another. Table 9.1 provides an evaluation of Schillo et al.'s (2000) trust representation.

Table 9.1 Analysis of trust representation using Schillo et al.'s (2000) socionic approach

CRITERIA	X	ADVANTAGES AND DISADVANTAGES
TRUST REPRESENTATION		
Trust outlook		
Blindly positive		
Blindly negative		
Slow negative, fast positive		
Slow positive, fast negative		
Balanced slow	**X**	ADVANTAGE: The process balances both positive and negative experiences. Due to the slow changes in values, agent is exposed to more interactions and hence has more opportunities before reaching a state of distrust. DISADVANTAGE: Agent is exposed to greater risk when the values decrease slowly.
Balanced fast		
Passionate vs. rational		
Passionate	**X**	ADVANTAGE: A dynamic, more adaptive approach. DISADVANTAGE: Exposes agents to higher risk and has higher processing overheads.
Rational		
Centralised vs. decentralised		
Centralised		
Decentralised	**X**	ADVANTAGE: A dynamic approach that includes differing perceptions. DISADVANTAGE: Every agent needs to keep its own trust-based information and requires more processing and storage capabilities.
Trust vs. distrust		
Trust only	**X**	ADVANTAGE: A simple representation that requires fewer calculations than when including distrust as a separate concept.

		DISADVANTAGE: Difficult to differentiate between low trust as a result of bad experience and low trust as a result of lack of information.
Distrust		
Scalability		
Scalable	**X**	ADVANTAGE: A scalable system can be expanded dynamically. DISADVANTAGE: A scalable system often requires higher maintenance.
Not scalable		

Trust outlook: This is a balanced slow approach. It relies on a probability measure to establish trust. This probability includes both positive and negative experiences, along with witness information to form a single trust value. Positive and negative experiences are given an equal weight, making it a balanced approach. The fact that agents collect a set of information makes it a slow approach. To influence a drastic change in trust in either direction, several witness reports or experiences are required.

Passionate vs. rational: This is a passionate approach. As stated before, agents can be both altruistic and malicious when behaving in a passionate way. Witness information is incorporated along with an agent's own information and agents form communities. Trust is represented by a single value that illustrates the probability of interaction success. This probability value does not determine any limitations, but only the likelihood of interaction success with another agent.

Centralised vs. decentralised: This is a decentralised approach. Each agent has its own unique perception(s). Any agent establishing a trust value takes into consideration its own community's witness information along with its own. When communicating with an agent from another community, the agent seeks its witness information from the agents in the other agent's community.

Trust vs. distrust: This approach has no explicit values for handling distrust. It assumes that a low probability value indicates distrust, but it does not know if this is due to too few interactions with the agent to make an accurate evaluation or as a result of several bad experiences.

Scalability: This model scales well, as agents store only a single probability of success value for other agents. Results of experiences are broadcast only to agents within an agent's own neighbourhood, limiting the number of agents an agent has to update. This allows agents to build small communities and store trust information about the community itself, as opposed to information about the entire environment.

Prejudice: Prejudice filters can be used to improve the performance of trust evaluation. The simplest way of implementing prejudice in this model is through categorisation. Agents can be categorised by the communities to which they belong, while the communities are again defined by the neighbourhood to which they belong. Information about an agent in a particular neighbourhood is obtained from the other agents that reside in that particular neighbourhood. If an agent receives trust-related information from a particular neighbourhood that tends to lean towards false information, an agent should be allowed to cease interactions with that specific neighbourhood. Another possibility is that of learning. Artificial intelligence agents are by nature agents that are defined by learning and they are expected to learn in order to adapt.

9.2.2 Dash et al.'s trust-based mechanism design

Dash et al. (2004) propose a trust-based mechanism design. As was defined above, it is the convergence of mechanism design with the principle of trust so that trust values are taken into consideration when various counterparts of a system are considered for allocation. Mechanism design allows a centre to allocate tasks in the most advantageous way. A centre is an agent that handles task allocations. Agents are not always successful in interactions, so trust is relied upon to test for probability of success (POS). Mechanism design incorporates trust model architecture in order to be efficient. Dash et al. (2004) have subsequently explored the properties that mechanism design requires from trust models.

The two properties of trust models as generalised by Dash et al. (2004) are the following: an agent's trust value in another agent is influenced both by its own perceptions and by those of other agents in the environment. This property incorporates recommendation- and reputation-based systems. The second property of trust involves the fact that trust is monotonic, meaning it increases with the possibility of success. The more successful an agent is, the higher the trust

value assigned to that agent. Agents use these two properties to update their level of trust over time.

Trust models that implement the two properties generalised by Dash et al. (2004) can then be incorporated into mechanism design by making a key change. Instead of calculating its own trust value and assigning its own tasks, each agent reports its POS value and a calculation function to a centre. The centre then uses this information to calculate the trust a particular agent possesses for other agents in the same environment. This trust is calculated on behalf of the agent and in the same way as the agent would have calculated it if it were calculating its own trust values. The agent that provides the centre with a POS value and a calculation function has no reason to lie to the centre, as the centre allocates task requests that originate from the agent by using the trust values it has obtained on behalf of the agent. Thus, the centre uses the trust function provided by the agent to evaluate the POS, which is a value between 0 and 1.

Mechanism design allows a centre to measure similarity between various agents and to allocate tasks based on this similarity. This similarity influences the POS value. Agents that are closer to one another in similarity possess similar POS values. Dash et al. (2004) rely on specific characteristics of trust models for success and these are summarised in Table 9.2.

Table 9.2 Analysis of trust representation using Dash et al.'s (2004) trust-based mechanism design

CRITERIA	X	ADVANTAGES AND DISADVANTAGES
TRUST REPRESENTATION		
Trust outlook		
Blindly positive		
Blindly negative		
Slow negative, fast positive		
Slow positive, fast negative		
Balanced slow	**X**	ADVANTAGE: The process balances both positive and negative experiences. Due to the slow changes in values, agent is exposed to more interactions and hence has more opportunities before reaching a state of distrust. DISADVANTAGE: Agent is exposed to greater risk when the values decrease slowly.
Balanced fast	**X**	ADVANTAGE: The approach balances both positive and negative experiences. Due to rapid changes in trust,

		agent is exposed to less risk when trust values are decreasing. DISADVANTAGE: Agent is exposed to fewer opportunities when values decrease quickly.
Passionate vs. rational		
Passionate		
Rational	**X**	ADVANTAGE: Rational agents are less likely to deviate in behaviour. This is a simple form of trust that is easy to represent and calculate. DISADVANTAGE: Rational agents are static agents and do not incorporate more intuitive forms of trust.
Centralised vs. decentralised		
Centralised		
Decentralised	**X**	ADVANTAGE: A dynamic approach that includes differing perceptions. DISADVANTAGE: Every agent needs to keep its own trust-based information and requires more processing and storage capabilities.
Trust vs. distrust		
Trust only	**X**	ADVANTAGE: A simple representation that requires fewer calculations than when including distrust as a separate concept. DISADVANTAGE: Difficult to differentiate between low trust as a result of bad experience and low trust as a result of lack of information.
Distrust		
Scalability		
Scalable	**X**	ADVANTAGE: A scalable system can be expanded dynamically. DISADVANTAGE: A scalable system often requires higher maintenance.
Not scalable		

Trust outlook: This model assumes a balanced trust approach. The trust value is associated with a probability of success value. This value is assumed to be a reflection of an agent's capability at any given time. Hence, keeping a trust value indefinitely in any position is undesirable. A slow update of any value is also undesirable, as the centre that allocates the tasks looks for the best current capability. Although a slow update of trust is undesirable, Dash et al.'s trust-based mechanism design will work with any trust model that takes a balanced approach.

Passionate vs. rational: Agents are assumed to be rational. In other words, it is assumed that an agent's behaviour is consistent. Agents are expected to resist malicious manipulation and an agent's behaviour is considered to be predictable. Dash et al. (2004) assume that the trust models measure trust by using a probability value. Passionate ways of using labels to represent trust are excluded. This approach requires that a trust model returns a single value between 0 and 1 that can be correlated to the probability of success.

Centralised vs. decentralised: This approach assumes a decentralised environment that allows agents to have their own perceptions. Mechanism design further measures similarity (also similarity in perceptions) between agents in order to establish POS values for unknown agents. Agents are even allowed to have their own unique trust functions that they forward to the centre. The agent that allocates the tasks, known as the centre, then uses the information provided by the trust function to assign tasks to other agents.

Trust vs. distrust: This approach has the same flaw as that of Schillo et al. (2000). It has no explicit values for handling distrust. It assumes that a low probability value indicates distrust, but does not know whether this low value is due to too few interactions with the agent to make an accurate evaluation, or to several bad experiences.

Scalability: This model is scalable and can even be implemented across varying trust model implementations, provided that the trust models adhere to the identified properties. Agents do not store vast quantities of data and require only a single value to establish trust with another agent.

Prejudice: The trust models used by Dash et al.'s (2004) trust-based mechanism design are assumed to incorporate reputation and recommendation principles, using both an agent's own perceptions and other agents' perceptions to arrive at a single trust value. Prejudice is best implemented within the trust model used by the trust-based mechanism design of Dash et al. Hence the best form of prejudice will be related to the particular trust model in place. Because of the assumptions made, the prejudice filters that are most likely to be relevant will be the recommendation- and categorisation-based filters. These allow agents to create categories for agents that they do not trust. The categories can be created based on the similarity between

agents, known as the similarity measure. Agents can be programmed to distrust agents that are very different from them in the similarity measure.

9.3 Conclusion

This chapter expanded on the analysis done in Chapter 8 by investigating two additional case studies and analysing them with regard to trust representation and prejudice filters. One trust model and one trust-based implementation that relies on trust models were analysed in this chapter. The trust model that was analysed was Schillo et al.'s socionic trust model, while the trust-based implementation analysed was the trust-based mechanism design of Dash et al.

Schillo et al.'s (2000) socionic trust model merges the fields of artificial intelligence and trust models to allow artificial agents to incorporate trust. This is a passionate, balanced slow approach, ideally suited to decentralised environments. It possesses no explicit values to indicate distrust and assumes that a low probability of success value indicates distrust. It is furthermore a quantitative numeric-driven approach that allows for scalability, due to the fact that agents are expected to form communities. Dash et al.'s (2004) trust-based mechanism design model relies heavily on trust models for success. The trust models required by the approach need to adhere to quite specific properties identified by the analysis. The trust model is expected to be balanced, rational and decentralised. It is expected to have a single value to indicate trust only and this value is expected to represent the probability of success. It is expected to be quantitative and scalable, storing only a single value for a trust relationship.

The Schillo et al. model (2000) can be extended to include prejudice filters, either by domain categorisation or learning due to the nature of the artificial intelligence agents. The mechanism design of Dash et al. (2004) should incorporate one of the recommendation or categorisation prejudice filters. Incorporating these prejudice filters is expected to improve the performance of trust evaluation.

Chapters 10, 11 and 12 implement a prototype trust model to illustrate the improvements that are possible with prejudice filters.

PART 5

PROTOTYPE AND SIMULATION

10. Prototype implementation

10.1 Introduction

The prejudice filters introduced in Chapters 6 and 7 are meant to improve the performance of trust models by decreasing the number of interactions a trust model attempts or evaluates in order to obtain a trust value that results in distrust. These filters are consequently aimed at increasing the number of successful interactions a model participates in. Several ways in which prejudice filters can be incorporated into trust models have been proposed in Chapters 6 and 7. All these ways rely on concepts that already exist in trust model architecture and that are meant to be an extension of existing trust models. The new abstract concept of prejudice filters still requires testing and evaluation. Chapter 10 takes the first step towards testing this concept by proposing a prototype with which to test the prejudice filters.

The proposed prototype is based on the trust reputation model of Abdul-Rahman and Hailes (2000). The simplest way in which to incorporate prejudice into this model is by means of recommendation, but the prototype is modified so as to include the domain prejudice filters defined in Chapter 6. Section 10.2 gives a structural overview of the prototype. A functional description of the prototype is given in Section 10.3. The various interactions that the prototype instigates are discussed in Section 10.4. Instructions to execute the prototype and interpret the results are given in Section 10.5 and Section 10.6 concludes the chapter.

10.2 Structural overview of the prototype

The prototype is a console-based application written in C#. The prototype executes on a Microsoft .NET Platform and is based on the trust reputation model of Abdul-Rahman and Hailes (2000). This model implements both a direct form and an indirect form of trust building. The direct form of trust building requires that an agent interacts directly with another in order to obtain and form a trust level. The indirect form of trust chosen by Abdul-Rahman and Hailes (2000) relies on recommendation. Only direct trust was implemented so as to scale down the implementation. This particular model was chosen because its direct trust building allows an agent to count and keep track of both successful and unsuccessful interactions, which makes it

possible for the researcher in the current study to use these values as benchmark values. The direct form of trust implemented by Abdul-Rahman and Hailes (2000) in their trust reputation model was modified to include constructs that can be used by the prejudice filters. This is accomplished by storing the domain an agent belongs to, as well as an agent's identity (ID) and context. A broad overview of the structure of the prototype is given in Figure 10.1.

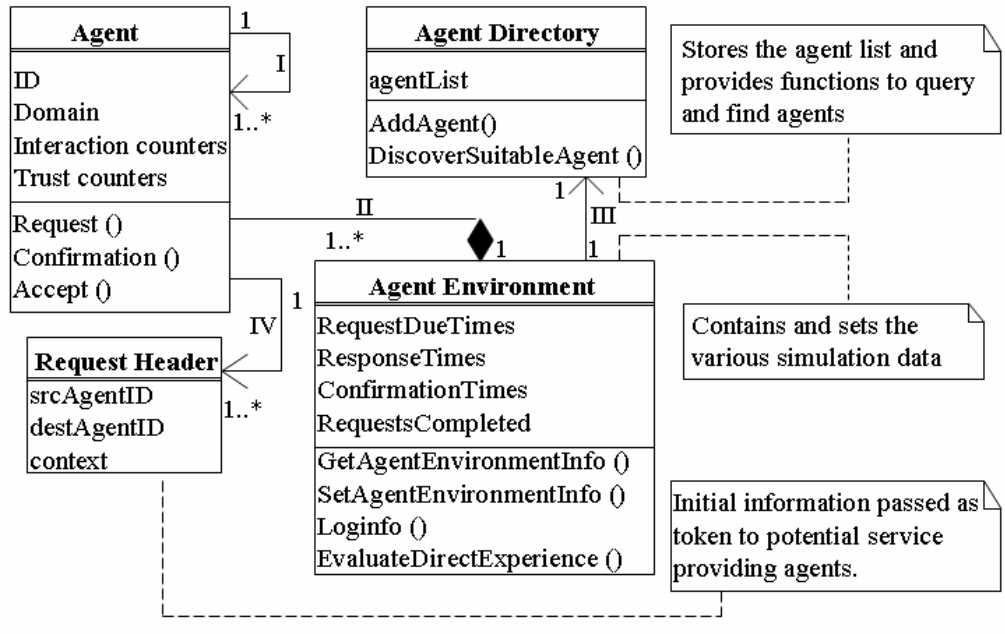

Figure 10.1 Class diagram of prototype implementation

The structure of the prototype consists of four key classes and the relationships between them, which are labelled with the roman numerals I to IV. These are referenced in the text that follows when they are discussed. The four key classes are Agent, Agent Directory, Request Header and Agent Environment. The attributes and methods that are used have been generalised, grouping all methods involved in a single task under a general method name that describes the task.

The Agent class defines an agent's attributes and methods. An agent possesses `ID` and `Domain` attributes. These attributes identify the agent to other agents. An agent's `ID` needs to be unique. The `Domain` value determines the environment an agent resides in. Hence, the domain value determines environmental factors that influence an agent's behaviour. In the case of the

prototype, the domain influences the agent's response times. Hence, the domain an agent resides in is used to implement the prejudice filter.

An agent also possesses two sets of counters. The first set of counters is labelled as interaction counters and this first set is used to record statistical information. These counters include counters for the following:

- `totalRequestsCompleted;`
- `totalAcceptsCompleted;`
- `RequestsSent;`
- `sentRequestsDenied;`
- `sentRequestsCompleted;`
- `requestsReceived;`
- `receivedRequestsDenied;` and
- `receivedRequestsCompleted`.

The above counters are used to track an agent's behaviour throughout the simulation and have been used for debugging as well as for analytical purposes. The first two counters keep track of the total number of interactions that have been completed – the first counting the number of requests that an agent has made and that have been completed by other agents, and the second counting the requests an agent has accepted from other agents and completed on their behalf. These counters give an indication of an agent's activity. The rest of the counters are used for the statistical information that is found in the output file named *AgentStatistics.tab* and they correspond to the table headings of the same name. They are all discussed in Section 10.5, which investigates the interpretation of results.

The second set of counters (trust counters) keeps track of the agent's trust-related data and contains counters for *vg* (very good), *g* (good), *b* (bad) and *vb* (very bad) respectively. The four counters record the experience grades of an interaction. A specific experience grade is the way in which an experience is interpreted and it refers to the counter that most closely resembles the result of the interaction. The prototype makes use of a timer to determine an experience grade. The experience deteriorates as time increases. The exact times used to determine the experience grades are defined in *DefaultAgentEnvironment.xml*. Each agent possesses a unique set of trust

counters for each agent it has knowledge about and only updates the counter set of the agent it is interacting with. For example, if Agent B is interacting with Agent A and has a *b* (bad) experience result, Agent B will update the *b* counter it has associated with Agent A. Experiences are evaluated and increment the counter that best describes the resulting experience. For example, if an interaction was good, the *g* counter gets incremented.

Agents communicate and interact with other agents, as illustrated by the relationship labelled 'I' in Figure 10.1. A single agent may interact with one or more other agents to accomplish a given task. An agent's methods are defined so as to allow that agent to communicate with other agents. The prejudice filters are implemented on the agent side and are contained in some of the agent's methods. An agent possesses `Request(), Confirmation()` and `Accept()` methods.

The `Request()` methods allow an agent to request a service. To do this, an agent looks for a suitable agent by checking the trust counters it possesses for every agent. If a suitable agent does not exist, the agent contacts the agent directory class and requests that the agent directory class return agents in the environment that it does not have trust counters for. Prejudice filters are implemented on the request side by allowing an agent to filter agents based on information it receives from the Agent Environment class, which in turn receives this information from the Agent Directory class.

The `Accept()` methods allow an agent to accept service requests from other agents. Prejudice filters are again implemented on the accept side (the methods that allow an agent to accept a request from another agent). Prejudice filters are implemented by allowing an agent to check the domain that a request is coming from. If the request is coming from an untrusted domain, the `Accept()` methods deny the request. Should the `Accept()` methods accept the request, the request is serviced.

The `Confirmation()` methods allow agents to confirm receipt of a service and ultimately terminate a connection. Trust updates are performed on the request and the accept side, as both the agent that requests a service and the agent that accepts a service are required to update the trust relationship according to their own perceptions.

The Agent Environment class is made up of one or more agents, and it controls the simulation data (labelled 'II' in Figure 10.1). The Agent Environment class controls the agents that have been defined, and it contains the various time-related data that the agents use. This time-related data includes `RequestDueTimes` that determine the exact time an agent makes a request, the `ResponseTimes` that determine the time it takes an agent to respond to a request and the `ConfirmationTimes` that determine the time taken to confirm a service received. The Agent Environment class also keeps track of the number of requests completed (`RequestsCompleted`). The Agent Environment class has four methods, namely `GetAgentEnvironmentInfo()`, `SetAgentEnvironmentInfo()`, `Loginfo()` and `EvaluateDirectExperience()`. The first two methods set up the environment in which the simulation executes. The third method keeps track of all the logging required by the simulation and records the information that is logged. The last method evaluates the result of interactions as all agents' evaluation processes work in the same way.

The Agent Environment class uses an Agent Directory class (labelled 'III' in Figure 10.1) to keep track of all the agents within the environment. The specific attributes that are initialised by the Agent Environment class are discussed in detail in the next section. When an agent is looking for other agents to interact with, it requests the Agent Environment class to inform it of the other agents that exist within the simulation. The Agent Environment class receives this information from the Agent Directory class by using the `DiscoverSuitableAgent()` method. The `Add()` method within the Agent Directory class is used during the initialisation of the Agent Directory class and adds all the agents that exist within that environment to the directory.

The last class is the Request Header class. This is simply a token that every agent possesses (labelled 'IV' in Figure 10.1) when that agent interacts with another agent. Agents may use more than one of these tokens if the agent interacts with more than one agent during the simulation.

10.3 Functional description of the prototype

The prototype implementation emulates several agents and executes on a single machine. The prototype also allows the user to control the number of iterations through which a single execution of a particular simulation will go. A single iteration allows each agent in the environment to make a single request. Thus, two iterations allow each agent in the environment to make two requests, and so on. Trust develops over time as agents learn the behaviour of other agents. Increasing the number of requests each agent makes illustrates the development of trust. To illustrate this development, the prototype allows the number of iterations to increase, thus increasing the number of requests an agent makes.

If an agent's request for a service is denied by the agent initially requested to perform the service, that agent is allowed to resend the request to other agents until it finds an agent willing to accept the request. This is still considered a single request. Even though an agent will request a service only once during every iteration, it may service a request several times. The service that the agent requests in the context of this prototype implementation is a booking. Since an agent is not allowed to perform its own booking, it is required to request the service from another agent. If the other agent accepts this request, the booking request is serviced.

Each agent possesses a set of trust counters for agents that it knows about and has interacted with. These counters determine the level of trust the agent has in the agents it knows about (as discussed) and they are evaluated when wishing to determine the trust level for another agent. The counter that returns the highest value when analysed using a `MAX` function determines the trust level given to an interaction. The four trust values associated with the counters are *vt* (very trustworthy), *t* (trustworthy), *u* (untrustworthy) and *vu* (very untrustworthy) and link to the four *vg*, *g*, *b* and *vb* experience counters respectively. Hence, if the *b* (bad) counter returns the highest value, the interaction is considered to be *u* (untrustworthy). Each of these counters is associated with a specific context. For simplicity's sake, the simulation was executed for a single context.

The appropriate counters are updated after an interaction has occurred between two agents. To identify these relevant counters, each agent needs to evaluate the experience. Experience

evaluation occurs by analysing the timers each agent possesses. If Agent A requests an interaction, it initiates a timer on its own side. It times the agent performing the request and stops the timer when it receives a response. The time taken to receive the response defines the way in which Agent A perceives the experience. The agent performing the service also has a timer of its own, because trust is considered to be unique to each agent participating in an interaction. The agent that performs a service initialises its timer when it sends a message confirming completion of the service and waits for acknowledgement. The time taken to receive the acknowledgement determines the experience result.

The prototype can be executed both with and without prejudice filters. Executing the prototype with prejudice filters requires that each agent filter out agents either before requesting a service or before accepting a service, as was discussed earlier. Specific properties for each agent are initialised in an xml file entitled *DefaultAgentEnvironment.xml*.

10.4 Definition of agent environment

The agent environment properties defined by *DefaultAgentEnvironment.xml* are illustrated in Figure 10.2. The xml tags in this file define units of data that are used by the prototype, as well as the various attributes used by the simulation. These tags are used to set the values required to execute a single iteration of the simulation.

The time at which an agent initiates a request is predefined by the `RequestDueTime` xml tag. Each agent has an exact moment at which to start performing the service request and the time that elapses is measured in milliseconds from the start of the simulation. Each agent possesses a unique ID and a `domain` to which it belongs. These are given by `agentID` and `agentDomain`. The `agentID` allows the agents to be paired into domains. The domains into which they are paired allow their general behaviour to be defined by the domain to which they belong. The `RequestDueTime` tells the agent at what time during the simulation it is expected to initiate a request. The `RequestDueTime` is for simulation purposes only. In a real-life situation, these requests occur when an agent needs a service. The initiation occurs some time

between `minTime` and `maxTime`. The `RequestDueTime` is counted from the start of the initialisation.

```
<Agent id='agentID' domain='agentDomain'>
      <RequestDueTime min='minTime' max='maxTime' />
      <ResponseTime min=' minTime ' max=' maxTime ' />
      <ConfirmationTime min=' minTime ' max=' maxTime ' />
      <ExperienceGradeTimes context='Booking' VeryGood='t1'
      Good='t2' Bad='t3' VeryBad='t4' />
      <ConfirmationExperienceGradeTimes context='Booking'
VeryGood='t1'
      Good='t2' Bad='t3' VeryBad='t4' />
      <DirectExperience agentId='agentID' context='Context'
      counter='ExperienceGrade' value='experienceValue'/>
      <TrustedExperienceGrade context='Booking' grade='VeryGood' />
      <UntrustedDomain context='Booking' domain='Book King' />
      </Agent>
```

Figure 10.2 Structure of an agent as defined by DefaultEnvironment.xml

The `ResponseTime` xml tag tells the agent how long it should wait after receiving a request before responding. This xml tag is used to emulate how long an agent takes to perform a given service. Some agents are faster than others. The `responseTime` xml tag allows the user of the simulation to emulate time and performance differences between agents that execute in different environments and on different platforms, on a single machine. In a real-life situation, this xml tag would not exist, as the agents themselves and the environments in which they exist would determine their response times. The agent would then respond at some time between `minTime` and `maxTime`.

The `ConfirmationTime` xml tag, also bound by `minTime` and `maxTime`, informs an agent how long it must wait before acknowledging the successful completion of a task. The `ConfirmationTime` xml tag is to allow the agent that performs a request to also have a simple time-based way to evaluate the agent it performs the service for. The agent that performs

the request sets its timer once it sends a request completed message to the agent that made the request. It then measures the time taken to receive a confirmation message, updating its own experiences accordingly. The `ExperienceGradeTimes` xml tag allows for the initialisation of what an agent considers to be the various results of an experience. This tag is initialised according to the time an agent is expected to take to complete a task. More specifically, `<ExperienceGradeTimes context='Booking' VeryGood='t1' Good='t2' Bad='t3' VeryBad='t4'/>`

This determines how an agent experiences a service and how long an agent is willing to wait for the completion of a request. The `ExperienceGradeTimes` xml tag value allows the agent that initialises the request to evaluate the agent that services the request. If an agent's request has been completed within t1 to t2 milliseconds, then the experience it had will be `VeryGood.` For example, note that

0 <= t1 <= t2 <= t3 <= t4, therefore,

[t1, t2) = VeryGood,

[t2, t3) = Good,

[t3, t4) = Bad, and

[t4, ∞)= VeryBad

`The ConfirmExperienceGradeTimes` tag (in Figure 10.2) works in the same way as `ExperienceGradeTimes`, only it refers to the time duration an agent is waiting to receive a confirmation and not the time duration an agent is waiting for completion of a request. This timer is initialised by the agent performing the service once it sends the result of the service to the agent that requested the service. The timer is stopped once the agent receives confirmation that the other agent received the result of the service performed. Inclusion of this timer allows the agent that services a request to evaluate the agent that initialises the request. The `DirectExperience` is for bootstrapping purposes and it allows for the initialisation of any values that an agent may already possess about other agents in the environment. This xml tag specifies the initial trust value and context for which the trust value exists. It initialises the agent, the context, the counter and the value of the counter for which the direct experiences exist. For example, assuming Agent A is given the following:

```
<DirectExperience agentId='B' context='Booking'counter='VeryGood'
value='1'/>
<DirectExperience agentId='B' context='Booking' counter='Good'
value='8'/>
```

Agent A will add a direct experience value entry for Agent B in the Booking context. It will then assign the value 1 to the counter counting *vg* experiences and 8 to the counter counting *g* experiences. Counter values that have not been explicitly specified using the `DirectExperience` xml tag will default to 0.

The `TrustedExperienceGrade` defines which levels of trust an agent is willing to give. Only agents of a defined experience grade are trusted. These experience grades are *vg*, *g*, *b* and *vb* and they refer to the counters that keep track of experiences an agent has had with another. The success of an interaction determines how an interaction is perceived. In the context of this simulation, an interaction's success is measured using the time taken to perform the service that has been requested. Longer time delays degrade the experience grade value. The `UntrustedDomain` xml tag is the xml tag related to the prejudice filter and it influences the simulation only when the prejudice filter is activated. This xml tag allows for the definition of untrusted domains. An agent then distrusts any agent that belongs to the untrusted domain. This approach relies on the assumption that an agent's environment in part influences its performance.

10.5 Interaction possibilities

The prototype allows for three basic interaction possibilities: one without the prejudice filter and two with the prejudice filter. The basic interaction without the prejudice filter assumes that no agents are filtered out and that one agent will interact with another; provided it trusts the agent within the given context. When an agent is blocked by the prejudice filter, so are any further interactions with it. A UML sequence diagram of this basic interaction without the prejudice filter is given in Figure 10.3.

The figure assumes that two agents are communicating with one another. Agent B wishes to perform an interaction and requires assistance from another agent. Agent B searches for suitable agents to service its request. First Agent B looks at its own trust counters and looks for agents

that it may trust. In an attempt to determine the trust values Agent B has for other agents, it runs a MAX function on the four counters associated with each agent.

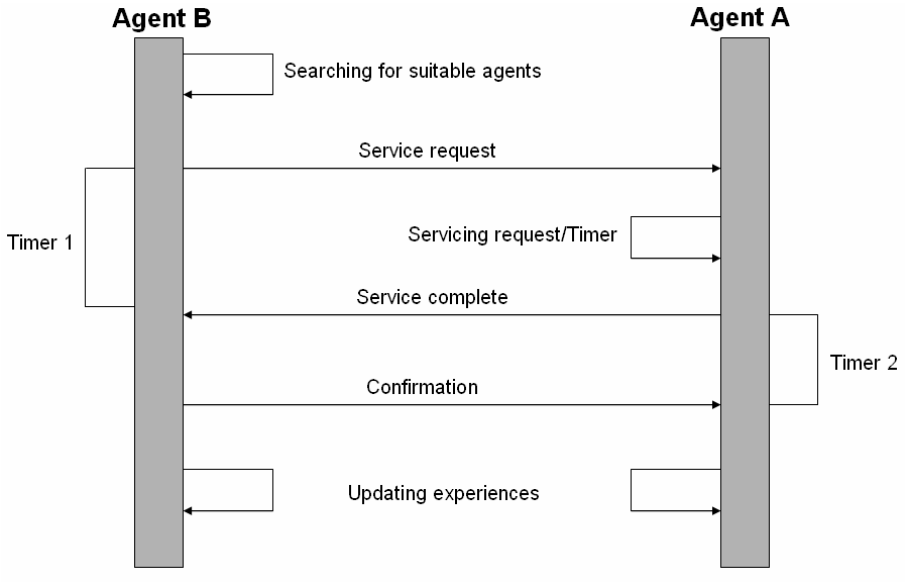

Figure 10.3 Simulation interaction without prejudice

The MAX function returns the counter with the highest value for each agent that Agent B knows about. Each agent has a predefined standard for trust. An agent only trusts other agents that meet the agent's own standards. These standards are defined by the `TrustedExperienceGrade` xml tag from Figure 10.2, contained in *DefaultEnvironment.xml*. This xml tag defines which counters an agent will accept as returns of the MAX function before it trusts another agent. Agent A's `TrustedExperienceGrade` is defined as follows:

```
<TrustedExperienceGrade context='Booking' grade='VeryGood' />
<TrustedExperienceGrade context='Booking' grade='Good' />
<TrustedExperienceGrade context='Booking' grade='MostlyGood' />
```

Agent A therefore trusts another agent, provided the MAX function returns either the *vg* or *g* counter for a specific agent. Conflicts, where more than one counter is returned by the MAX function, are resolved according to the following logical rules:

If MAX = *vg* & *g*

Then `MostlyGood`

Else if MAX = *b* & *vb*

Then `MostlyBad`

Else Uncertainty

Consequently, Agent A also trusts agents where the `MAX` function returns the *vg* and the *g* counter.

If an Agent B is not able to find another agent which it can trust, it consults the agent directory class. Once a suitable agent has been found (in the case of Figure 10.3 this is Agent A), Agent B sends a service request to the suitable agent. The suitable agent services the request and informs Agent B that the request has been completed. In the meantime, Agent B is timing Agent A. The time that Agent A takes to complete the given request defines the experience value Agent B will assign to the interaction. The specific time thresholds are also defined in *DefaultEnvironment.xml*. Agent B subsequently sends an acknowledgement to Agent A. This acknowledgement is considered to be a confirmation by the prototype. Agent A measures the time taken to receive the acknowledgement, so that it can update its own experiences with Agent B. Once the acknowledgement has been both sent and received, both agents update their respective experience grades.

The interaction discussed above defines how the agents communicate with one another if there are no prejudice filters in place. The next two UML sequence diagrams look at two alternative possibilities that occur when prejudice filters are implemented. The prejudice filter allows for two different interactions, because the prejudice filter allows agents either to accept or to deny a request. An agent may filter out another agent either on the request or on the accept side. On the request side, an agent seeking a new interaction can filter out agents that it receives from the agent directory class before attempting to communicate with them, simply because of the domain to which they belong. On the accept side, an agent receiving a request may deny the request based on the fact that the request comes from a domain which it does not trust.

Figure 10.4 illustrates a successful interaction with prejudice filters, while Figure 10.5 illustrates an interaction denial. In both the prejudice filter-based possibilities, Agent B looks for other agents in the same way as it did in Figure 10.3. Agent B finds all other possible suitable agents in the simulation and then filters out undesirable agents by checking the domains of the new agents

it has found. Agent B then only communicates with the agents that have passed this filter test and consequently chooses an agent from an acceptable domain.

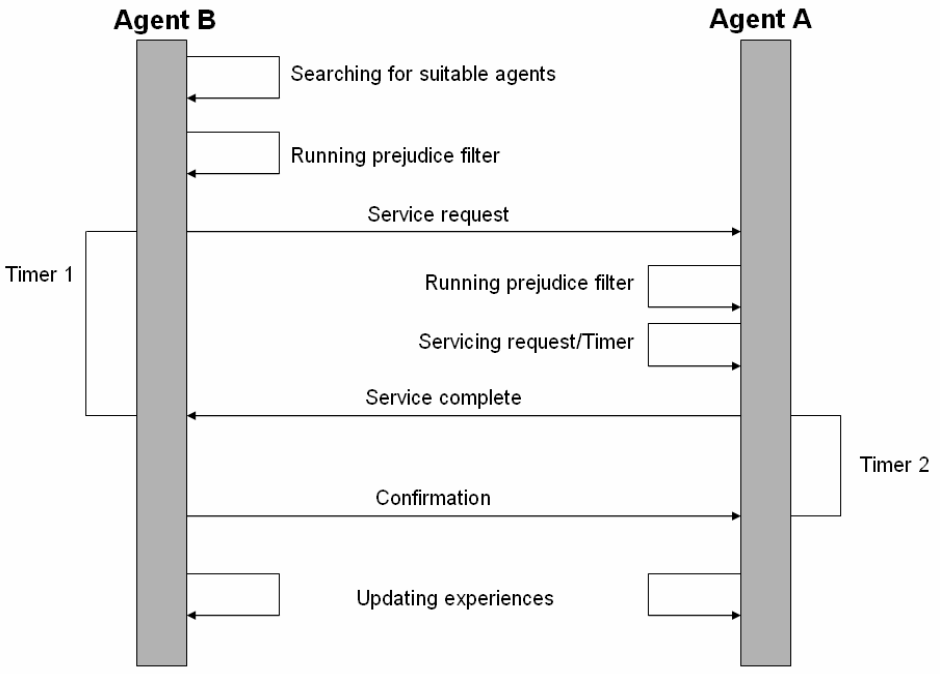

Figure 10.4 Successful interaction with prejudice filter

Agent B next sends the service request to Agent A, who executes its own prejudice filter and checks whether the request is coming in from a trusted domain. Should Agent B pass through the prejudice filter, the interaction continues as illustrated in Figure 10.4. If not, then the interaction continues as illustrated in Figure 10.5.

Once Agent A has accepted the request, Agent A services the request, while Agent B measures the time duration of the interaction. Agent A sends a service complete message to Agent B and waits for an acknowledgement message from Agent B while setting its own timer. The prototype implements the acknowledgement as a confirmation. Once Agent B has sent its acknowledgement to Agent A, both agents use their respective timers to update their experience counters. In the second prejudice filter UML sequence diagram, Figure 10.5, Agent A denies Agent B's request, based on its own prejudice filter results. If this happens, Agent A sends a service denied message to Agent B. Agent B then contacts the next suitable agent. If no suitable agents can be found, the interaction fails.

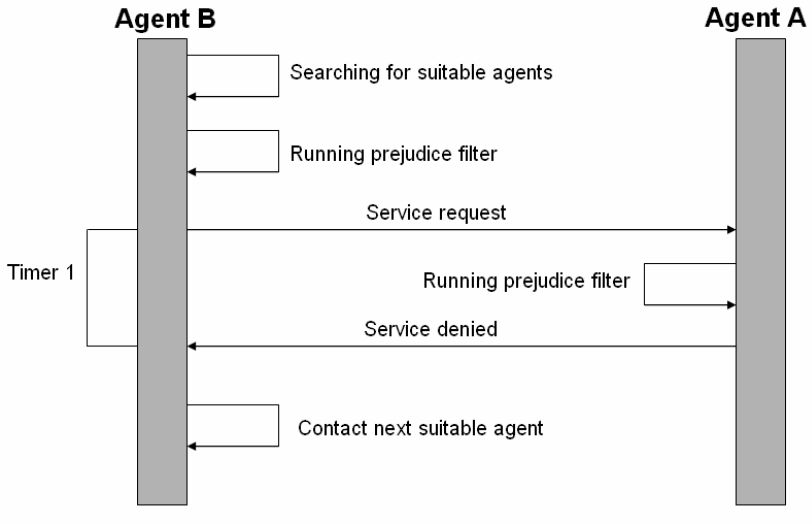

Figure 10.5 Denied interaction with prejudice filter

10.6 Executing the prototype

Detailed instructions on executing the prototype can be found in the file README.txt found on the compact disk accompanying the dissertation; however, a summary of how to execute the prototype is provided in this section. In order to run a particular scenario, various attributes need to be set for each agent. These attributes are defined in *DefaultAgentEnvironment.xml*. The console should be used to browse to the directory where the executable of the prototype is stored. The prototype is executed using various commands from the console. Examples can be found in Figure 10.6.

The prototype requires two compulsory parameters and three optional parameters. The two compulsory parameters are the trust environment defined by an xml file and the number of iterations the simulation must go through, defined by an integer. In the case of the given simulations, the xml file refers to *DefaultAgentEnvironment.xml*. The three optional parameters control the way in which the simulation behaves. The `nofilter` parameter is the most important one, and it is used for the case study's purposes. When it is used, the `nofilter` parameter results in the simulation executing without a prejudice filter. When this option is omitted, the simulation executes with the prejudice filter.

```
TrustSimulation.exe DefaultAgentEnvironment.xml 1
TrustSimulation.exe DefaultAgentEnvironment.xml 1 nofilter
TrustSimulation.exe DefaultAgentEnvironment.xml 1 noupdates
TrustSimulation.exe DefaultAgentEnvironment.xml 1 noupdates nofilter
TrustSimulation.exe DefaultAgentEnvironment.xml 1 nofilter noupdates
TrustSimulation.exe DefaultAgentEnvironment.xml 1 nofilter noupdates
nologging
TrustSimulation.exe DefaultAgentEnvironment.xml 1 nologging
```

Figure 10.6 Various commands that can be used to execute the prototype simulations

The last two parameters are a result of development and are for control purposes. The `noupdates` parameter means that the direct experiences are not to be updated during a simulation. This allows the simulation to be executed in such a way that agents do not take into consideration the change in trust after interactions. Since the current research is interested in the change in trust, this option is not used by the simulations investigated in this dissertation. The `nologging` parameter means that the simulation will not log any of the simulation output to increase the speed at which the prototype executes. Since we are actually interested in comparing the results, this parameter is not used during the case studies. Figures 10.7 and 10.8 demonstrate the console output that is generated when the simulation is executed with and without prejudice filters respectively. In both figures, the simulation is executed for one iteration for illustrative purposes. Increasing the number of iterations simply increments the number of requests that each agent in the environment makes during the simulation and results in the same output on the console as shown in Figures 10.7 and 10.8.

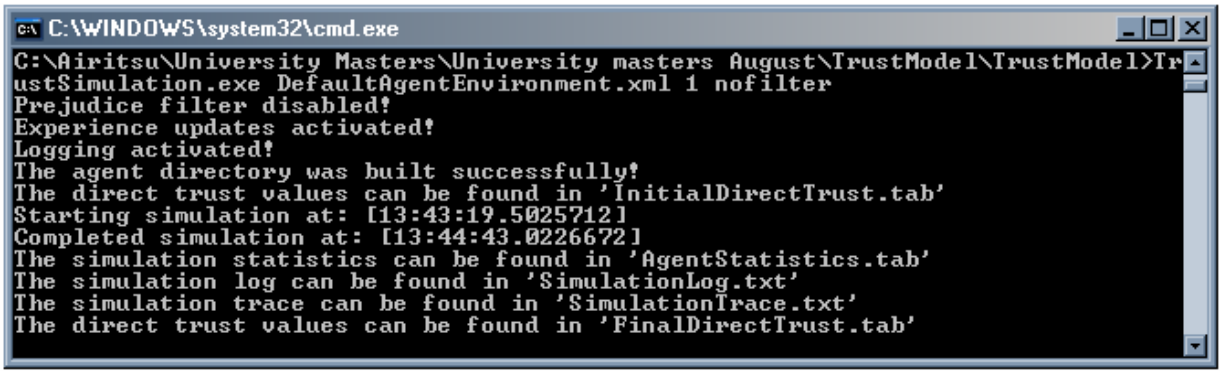

Figure 10.7 Simulation output without prejudice filter

As can be seen by the console output, the simulation generates various output files that store the results of the simulation. The files generated include the following: *InitialDirectTrust.tab*, *AgentStatistics.tab*, *SimulationLog.txt*, *SimulationTrace.txt* and *FinalDirectTrust.tab*.

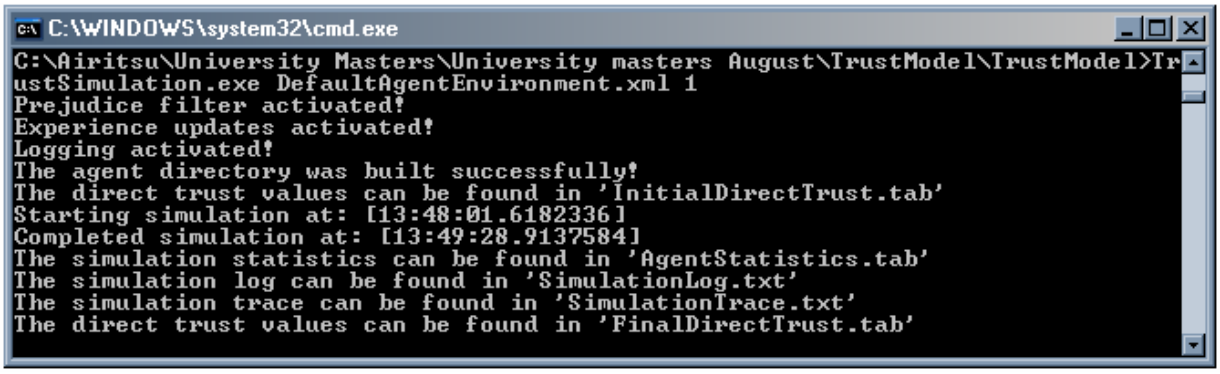

Figure 10.8 Simulation output with prejudice filter

10.7 Interpreting the results

The prototype disables or enables the prejudice filter, activates logging and creates the agent directory that is used to identify other agents in the environment. It then creates a file *InitialDirectTrust.tab*, which stores the initial trust values for every agent in the environment. Sample output is given in Table 10.1.

Table 10.1 Sample output stored in *InitialDirectTrust.tab*

srcAgent	destAgent	Context	vg	g	b	vb	Evaluation
f	a	Booking	1	0	0	0	VeryGood
f	e	Booking	0	1	0	0	Good

The first column, srcAgent, displayed when opening the *InitialDirectTrust.tab* file in a tab-delimited interpreter such as Microsoft Excel, refers to the agent that is storing the trust-related information. In the example given, the agent storing the information is Agent F. The second column, destAgent, refers to the agents about which the trust information is stored – Agents A and E in the given example. The third column refers to the context for which these values exist. The next four columns store counter values as defined by Abdul-Rahman and Hailes (2000). The values are *vg* (very good), *g* (good), *b* (bad) and *vb* (very bad) respectively. The final column, Evaluation, gives the trust level that will result from an analysis of the counters stored. After

execution of the simulation as concluded, four more files are created: *AgentStatistics.tab*, *SimulationLog.txt*, *SimulationTrace.txt* and *FinalDirectTrust.tab*.

AgentStatistics.tab stores various statistics for each agent. These statistics include the number of *Requests Sent*, *Sent Requests Denied*, *Sent Requests Completed*, *Received Requests*, *Received Requests Denied* and *Received Requests Completed* for each agent and context in which these occurred. Table 10.2 illustrates the data stored.

Table 10.2 Example output stored in AgentStatistics.tab

AgentId	Context	Requests Sent	Sent Requests Denied	Sent Requests Completed	Received Requests	Received Requests Denied	Received Requests Completed
b	Booking	1	1	1	0	0	0
a	Booking	1	0	1	6	3	3

The first column stores the agent identifier. The second stores the context for which there are trust statistics. The rest of the columns store the statistics. The first three statistics are for the number of requests that the particular agent has initiated. *Requests Sent* is the number of requests an agent has initiated. *Sent Requests Denied* refers to the number of times an agent's request has been denied. This number may be higher than that of *Requests Sent* because an agent may send a single request more than once, to more than one agent. Thus it is possible for the same request to be denied multiple times. The number of requests that an agent initiates and that have been successfully completed by other agents is stored in *Sent Requests Completed*. The last three statistics are for the requests that the agent is requested to accept. *Received Requests* refers to the number of requests an agent is required to service. *Received Requests Denied* refers to the number of received requests that the agent has denied. Finally, the number of requests that the agent has accepted and successfully serviced can be found in *Received Requests Completed.*

The *SimulationLog.txt* file stores the exact order in which the various events in the simulation occur. (An example of the output generated during a simulation without prejudice filters and with prejudice filters can be found in Appendices A and B respectively.) *SimulationLog.txt* stores the times and order in which an agent performs given events such as request initialisation, request acceptance, agent searching, the activation of prejudice filters, updating of experience grades and termination of interactions. *SimulationTrace.txt* stores information about the same events that

have been stored in *SimulationLog.txt*. The difference is that, while *SimulationLog.txt* orders the information by the time at which the various events occurred, *SimulationTrace.txt* orders the information by the agents that were participating in the various events. Hence, all the events that Agent A participates in will be listed, followed by those that Agent B participates in and so on. The *FinalDirectTrust.tab* file stores the final trust values that each agent possesses for other agents in the environment and it is to be interpreted in the same way as *InitialDirectTrust.tab.*

10.8 Conclusion

This chapter discussed the prototype used to test the impact of prejudice filters on trust. Abdul-Rahman and Hailes's (2000) implementation of direct trust within their trust reputation model has been implemented and modified to include prejudice filters. To adapt the direct trust implemented by Abdul-Rahman and Hailes so as to include domain-related filters, the agents were modified to store the domains that they do not trust.

This chapter furthermore investigated the structure of the prototype, the way in which it executes; the way in which agent properties are defined, the various interaction possibilities, as well as the way to execute the prototype and interpret the results given. A scenario is required to demonstrate the prototype effectively. Chapter 11 introduces a scenario that is to be used for the prototype's various simulations.

11. Prototype Scenario

11.1 Introduction

The prototype discussed in the previous chapter modifies and implements the direct trust defined by Abdul-Rahman and Hailes's (2000) trust reputation model. The prototype implements the domain filter, which relies on the fact that the environment in which an agent resides influences the agent's actions. In order to test the prototype, a simulation scenario is required. This scenario is used to execute the simulation and to compare the results obtained with, as well as without prejudice filters.

Chapter 11 introduces the scenario used to obtain simulation results for evaluation. A measurement mechanism is required to differentiate between good domains and bad domains. In the case of the prototype, and hence the scenario described in this chapter, the measurement mechanism is the time that agents take to respond to various actions, such as service requests and service confirmations. This time differs among domains. Section 11.2 introduces the scenario used for the simulation and for testing the impact of prejudice filters on trust. Section 11.3 concludes the chapter.

11.2 Scenario

The prototype uses a simulation to illustrate the impact of prejudice filters on trust. The current chapter provides the scenario which was then used to execute the different simulation case studies, the results of which are compared. The defined scenario remains the same throughout the various tests of the simulation and allows for the comparison of data results obtained from executing the scenario with and without prejudice filters.

The scenario defines three booksellers: *The Bookshoppe*, *Book King* and *Book*s. Each bookseller has its own branch stores and an agent that controls all the information related to the particular branch store in which the agent resides. The three booksellers work together to supply books to customers. When a particular branch store of a bookseller does not have a particular book in stock, it consults with its other branch stores as well as with the other booksellers that it knows

about to find and reserve the requested book on behalf of the customer. The process of reserving a book is a service provided by the booksellers. This service defines the transactional context in which the interaction takes place. The prototype labels this context as 'Booking' and sees the process of reserving a book as a booking process. The agents in charge of a particular branch store of a bookseller instigate the booking process. The agent of the store that does not have a book in stock instigates a booking request, asking one of the agents either from another branch store of its own store or from another bookseller to reserve the book requested by the customer. The agent that receives the booking request chooses either to handle the request or to deny it, based on its own trust evaluation process. Should the agent choose to handle the request, it reserves the book and then confirms the reservation. The agents involved in the booking process are required to keep track of the trust they have in other agents to ensure good service and customer satisfaction.

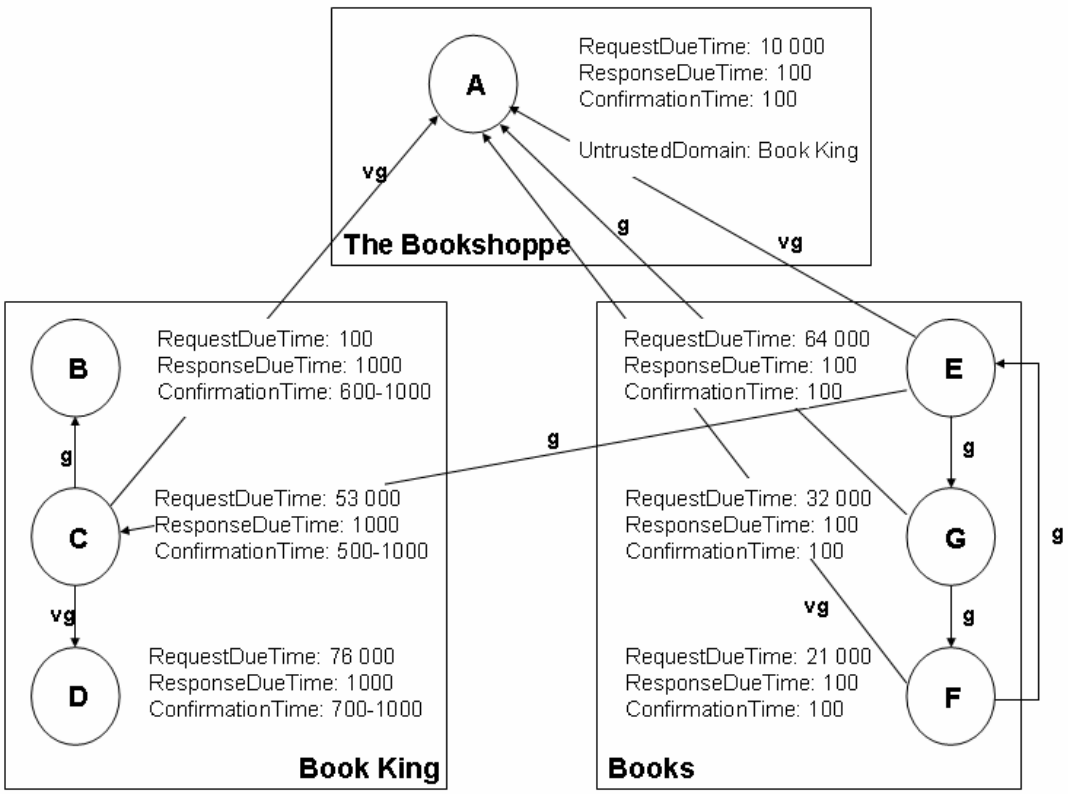

Figure 11.1 Simulation scenario with time delays represented in milliseconds

Figure 11.1 gives an overview of the general agent setup within this scenario and specifies a series of booking processes initialised by the three booksellers. The simulation scenario consists

of seven agents that represent branch stores (A-F) within three domains that represent the booksellers: *The Bookshoppe* (contains Agent *A*), *Book King* (contains Agents B, C and D) and *Books* (contains Agents E, F and G). Each agent possesses its own values for various tags defined within the simulation. Definitions of these tags were discussed in detail in Chapter 10. The specific detailed definitions for each agent can be found in *DefaultEnvironment.xml*. Figure 11.2 gives an example of such a definition by providing the definition for Agent A. The following information can be gained from the tags used to define Agent A:

```
      <Agent id='A' domain='The Bookshoppe'>
            <RequestDueTime min='10000' max='10000' />
            <ResponseTime min='100' max='100' />
            <ConfirmationTime min='100' max='100' />
            <ExperienceGradeTimes context='Booking' VeryGood='0'
Good='3900'
                        Bad='4100' VeryBad='4500'/>
            <ConfirmationExperienceGradeTimes context='Booking'
VeryGood='0'
                       Good='600' Bad='800' VeryBad='1000'/>
            <TrustedExperienceGrade context='Booking'
grade='VeryGood'/>
            <TrustedExperienceGrade context='Booking' grade='Good'/>
            <TrustedExperienceGrade context='Booking'
grade='MostlyGood'/>
            <UntrustedDomain context='Booking' domain='Book King'/>
      </Agent>
```

Figure 11.2 Value definition for Agent A

Agent A belongs to the domain: *The Bookshoppe*. Agent A makes a request 10 000 milliseconds after the beginning of the simulation and takes 100 milliseconds both to respond to a service request and to confirm the receipt of a request from another agent. The tags that have a `min` and `max` value defined in *DefaultEnvironment.xml* allow the simulation to be controlled to some extent. The `min` and `max` values define the time range in which an action is allowed to happen and introduce randomness into the simulation. The `ExperienceGradeTimes` tag indicates Agent A's tolerance to slow responses. If the service response from another agent takes less than 3 900 milliseconds, the experience result of the interaction is `VeryGood`. If the response takes 3 900 milliseconds and more but less than 4 100 milliseconds, then the response is considered Good. Responses that take 4 100 milliseconds and more, but less than 4 500 milliseconds are considered to be Bad, while responses that take 4 500 milliseconds and more are considered to be

`VeryBad`. The values for `ConfirmationExperienceGradeTimes` are interpreted in the same way as those for `ExperienceGradeTimes`. The difference is that `ConfirmationExperienceGradeTimes` applies when Agent A is waiting for a confirmation from another agent. Agent A trusts all agents whose trust values evaluate to `VeryGood, Good` and `MostlyGood`. Agent A distrusts agents from the domain *Book King*.

Figure 11.1 illustrates the values that have been assigned for the `RequestDueTime`, `ResponseDueTime` and `ConfirmationTime` tags for each agent. The `RequestDueTime` tag defines the exact moment after the beginning of the simulation at which a single agent will start executing a request for a service. A service in this context is a request for a booking to be performed. The `ResponseDueTime` tag defines how many milliseconds an agent waits before responding to a booking request, and the `ConfirmationTime` tag defines how long an agent will wait before sending a confirmation message.

The confirmation message is used to inform the agent that performs the service that the agent that requested the service has received the results of the service request. The confirmation message allows the agent that performs the request to evaluate the agent that requested the service by timing how long it takes the other agent to confirm the receipt of the service. This supports the trust-based assumption that each agent participating in an interaction has its own unique perception of trust. Hence, the agent performing the request is also expected to do a trust evaluation of an interaction. The delegation of services is enforced by the simulation itself as the simulation does not allow an agent to perform its own booking.

Each of the `RequestDueTime, ResponseDueTime` and `ConfirmationTime` actions can be defined within a range between a `min` and `max` value. The `min` value defines the minimum waiting time before an agent performs the desired action, while the `max` value defines the maximum time an agent will wait before performing a given action. The agent can perform the desired action at any time between the `min` and `max` values. When the `min` and `max` values are the same, the agent performs the desired action at exactly the specifically defined time. This precise time is counted from the beginning of the simulation when defined by `RequestDueTime`. In the case of the `ResponseDueTime` and `ConfirmationTime` tags,

this is an exact time delay in response to a request for a service and a confirmation of the service respectively. Figure 11.1 gives the time delay definitions defined for each agent. These values are given as ranges, except for the cases where both the `min` and `max` values are the same. When the `min` and `max` values are the same, there is only a single value beside the specific tag within the figure.

Agent A is the key agent in the simulation and all results are viewed from the perspective of Agent A. The directional arrows between agents illustrate the prior knowledge and trust levels that agents have in respect of each other. The agent at the root of the arrow is the agent in possession of that trust knowledge and the agent at the arrow head is the agent about whom the knowledge is possessed. Each arrow is labelled with the dominant trust level. For instance, the arrow between Agent F and Agent E is labelled *g*. This means that Agent F's trust in Agent E is good. These labels relate to the experience levels discussed above.

The domain *Book King* is considered to be the problematic domain that will be filtered out due to its slow `ResponseDueTime` and `ConfirmationTime`. This domain will tend to result in more failed interactions with Agent A because of its slow response time. Thus the prejudice filter filters out this domain, as illustrated by the `UntrustedDomain` tag. If Agent A receives a request for a service from Agent B, it checks its `UntrustedDomain` tag. From the `UntrustedDomain` tag, Agent A can see that it distrusts agents from the *Book King* domain. Since Agent B resides in the domain *Book King*, it is distrusted by Agent A, and Agent A denies Agent B the service. The results of executing this specific scenario several times with varying numbers of iterations will be graphically illustrated in the next chapter.

11.3 Conclusion

A scenario is needed to execute the simulation required for analysing the effect of prejudice filters on trust. This chapter has introduced such a scenario in which three booksellers and their branch stores work together to reserve books on behalf of customers. The branch stores of the booksellers, represented by agents, are expected to interact with one another to reserve books, a service that is provided by the bookings that agents instigate. In order to satisfy their customers,

these bookings must be fast, efficient and reliable. Consequently, agents need to establish trust relationships to guarantee customer satisfaction by dealing only with other agents that meet these specifications.

The environments in which the agents reside are controlled by predefined rules that influence the way in which agents operate. In the case of the given scenario, the bookseller to which the various branch stores belong defines the rules that those branch stores (represented by agents) are required to adhere to. Therefore, all the branch stores of a particular bookseller are expected to behave in a similar way. A different bookseller may find the behaviour of these branch stores undesirable and may wish to refrain from providing services to or requesting services from them. The bookseller consequently changes its trust value for the branch stores with the undesirable behaviour to distrust. Trust models use the trust rules that they have defined to establish these trust and distrust values, based on experience. Often this trust adjustment is done individually for each agent.

Prejudice filters are expected to simplify and assist this process by denying interactions with all the branch stores of a particular bookseller, rather than having to determine a level of distrust for each branch store individually, based on experience. Chapter 12 looks at the simulation results of various case studies performed on the scenario described in Chapter 11 and compares the results that are obtained when the simulation is executed with prejudice filters with those obtained when executed without prejudice filters.

12. Prototype simulation results

12.1 Introduction

Chapters 10 and 11 looked at a possible prototype and scenario that can be used to test the impact of prejudice filters on trust. Testing the impact of prejudice filters on trust implies looking for a performance improvement in the trust model that has been extended by the prejudice filters. As a prototype and a scenario have already been described, all that remains is a comparison of the simulation results.

This chapter shows the testing of the incorporation of domain prejudice into the trust reputation model of Abdul-Rahman and Hailes (2000). As stated in the previous chapters, the prototype implemented only the direct form of trust as defined by Abdul-Rahman and Hailes, and it modified the model to record domain-related information. This simple modification allowed for the incorporation of domain-based prejudice filters. Section 12.2 demonstrates and graphically depicts two case studies and their results, while Section 12.3 concludes this chapter.

12.2 Simulation results

Two main case studies have been conducted using the scenario defined in Chapter 11. As an agent interacts with other agents, it gains experience and the experience gained influences the trust an agent has in the other agents. The first case study explored the impact of the experience gained by allowing the number of interactions that agents participate in to increase with each simulation execution. The second case study demonstrated the situational effect on trust by allowing the various delays introduced by the operating system into the simulations to affect the simulation results. In the first case study, this impact was minimised by rebooting and clearing the operating system memory before each successive simulation execution.

12.2.1 Case study 1

The first case study executes the simulation ten times without the prejudice filter. Thereafter the simulation is executed ten times with the prejudice filter. The first execution of the simulation

sets the iteration value to one. Thereafter the iteration number is increased by one with each consecutive execution until the value reaches ten. The results of each execution were recorded.

12.2.1.1 Simulation results of case study 1

Table 12.1 records the results of the simulation without the prejudice filters and Table 12.2 records the results with the prejudice filters. All the results recorded in this chapter were recorded from Agent A's perspective and illustrate the trust data recorded by Agent A. This trust data includes results of interactions with all the other agents in the environment (Agents B, C, D, E, F, G and H) collectively. The columns of the tables represent the number of iterations through which the simulation was executed. During each iteration, each of the six agents in the environment (Agents B, C, D, E, F and G), made one request for a service. Hence, during four iterations, the six agents made four requests for a service each, resulting in a total of 24 interaction requests. The top row of each table records the number of iterations a simulation was set to execute. The data beneath a specific iteration number illustrates the results recorded when the simulation was run for that many iterations. The successive rows list the four counters that Agent A possesses. The value next to each counter represents the number of experiences resulting in a particular experience grade that Agent A had during a given simulation execution. The experience grade refers to the way in which Agent A experienced interactions and is a result of an analysis process that occurred once an interaction had concluded. These counters record the sum of all different experiences Agent A had with all other agents in the environment. This allows us to use analyse Agent A's total number of experiences, defined by a particular experience grade.

Table 12.1 Agent A's results for Case Study 1's simulation without prejudice filters

No. of iterations:	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Interaction result										
vb	3	3	4	4	2	3	3	3	14	6
b	1	1	0	0	2	1	1	1	2	1
g	0	0	0	1	0	0	0	0	0	1
vg	3	5	7	6	9	8	9	11	1	14

Table 12.2 is designed in the same way as Table 12.1. The only difference is the actual data represented by the table, as this data includes the results of the prototype executed with prejudice

filters. The simulation with prejudice filters filtered out the agents that had a long time delay before responding. Table 12.2 demonstrates a clear decrease in bad experiences and an increase in good experiences overall.

Table 12.2 Agent A's results for Case Study 1's simulation with prejudice filters

No. of iterations	1	2	3	4	5	6	7	8	9	10
Interaction result										
***vb*(f)**	0	0	0	0	0	0	0	0	0	1
***b*(f)**	0	0	0	0	0	0	0	0	0	1
***g*(f)**	0	0	0	0	0	0	0	0	1	0
***vg*(f)**	4	8	12	16	20	24	28	32	35	37

To better explain the data represented in the tables, the results demonstrated in Tables 12.1 and 12.2 are represented graphically in Figures 12.1 and 12.2 respectively.

From Figure 12.1, it is apparent that the number of very good experiences increased as the number of iterations as well as the number of requests increased. With the exception of the simulation that was executed through nine iterations, the very bad experiences did not increase as much as the very good ones. The bad experiences demonstrated a general stability, remaining more or less the same across the various iterations. As the number of iterations increased, the agents obviously gained more knowledge about their neighbours and adjusted their trust values accordingly, hence not increasing negative experiences as the number of iterations increased. This allowed agents to narrow down the number of agents they requested services from and to request services only from agents that gave favourable responses – thereby increasing the number of positive experiences over time. The good experiences were practically nonexistent.

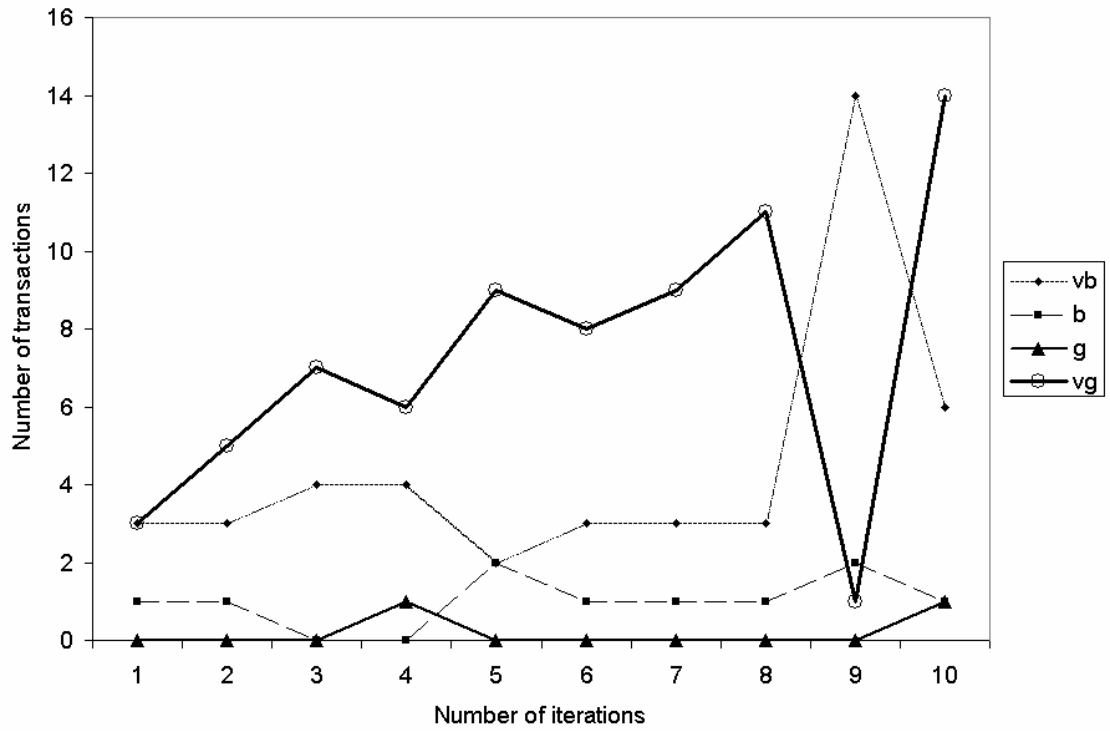

Figure 12.1 Graphic results for simulation of Case Study 1 without prejudice filters

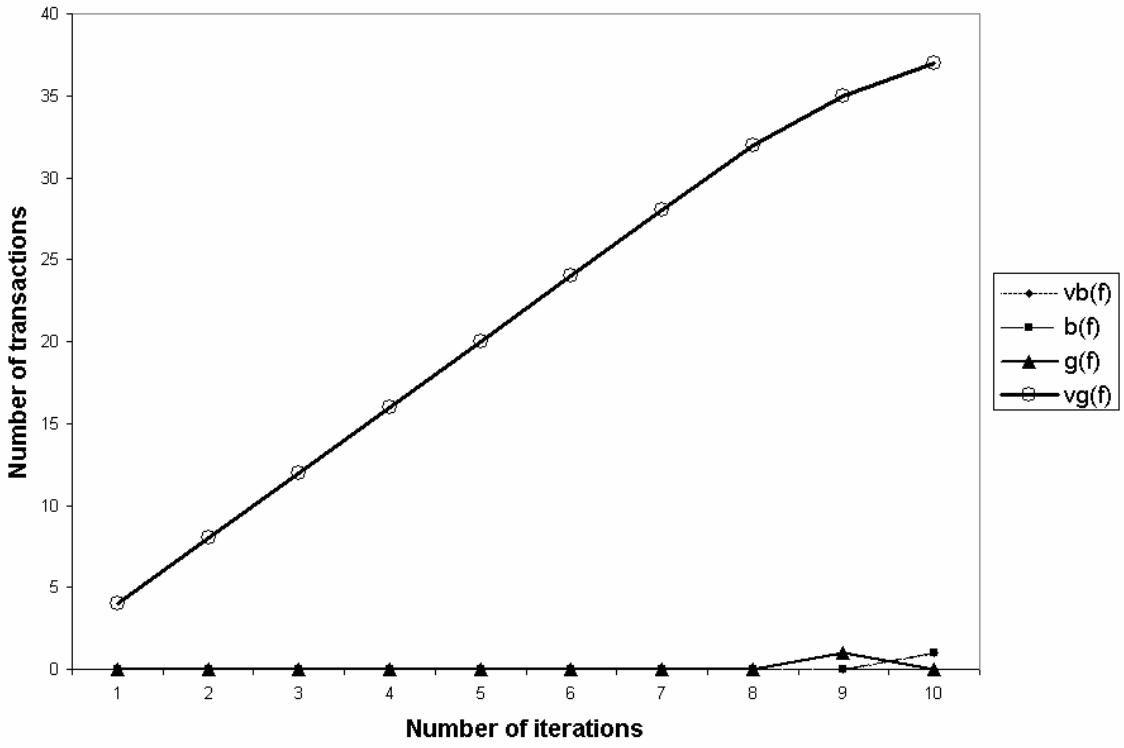

Figure 12.2 Graphic results for simulation of Case Study 1 with prejudice filters

Figure 12.2 demonstrates a clear improvement on the results given in Figure 12.1. Both the very bad and bad experiences were minimised to the point of being almost non-existent, while the very good experiences increased greatly. The good experiences were also minimised in favour of the very good ones. Each of the four lines, representing *vg*, *g*, *b* and *vg***,** were taken from the two graphs and placed into their own graph so that the difference for each experience grade could be illustrated diagrammatically. This allows for easy comparison between the results gained without the prejudice filter and the results gained with the prejudice filter. The experience grade refers to the four counters that determined the result of an experience and consequently the level of trust assigned to another agent.

12.2.1.2 Comparison of the individual counters for Case Study 1

Each of the graphs depicted in Figures 12.3 to 12.6 illustrates what happens to the number of interactions within each trust experience grade across the spectrum of *vb*, *b*, *g* and *vg*. The bold line represents the data recorded with the prejudice filter, while the dashed line represents the data recorded without the filter.

The number of very bad experiences, as illustrated by Figure 12.3, decreased drastically when prejudice filters were included in the simulation. They practically disappeared. When the simulation was executed without prejudice filters, the number of very bad experiences was a virtually consistent presence with only a few deviations as the number of iterations increased. The reason why the number of very bad experiences did not rise in general, but tended to remain the same or close to a specific value (3), is that the number of agents that an agent encounters is limited and controlled within the environment of the simulation. Also, the behaviour of the agents was kept predictable to some extent. Once an agent had a very bad experience with the distrusted agents, it no longer interacted with those agents in further iterations. The prejudice filter entirely excluded the agents that tended to give slow responses. Agent A denied all requests coming from those agents and did not request a service from them either, thus reducing the number of very bad experiences overall.

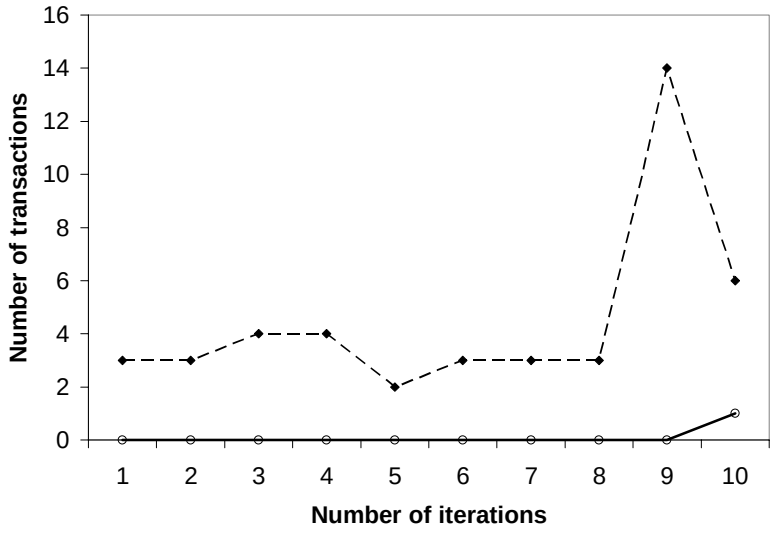

Figure 12.3 Comparing Case Study 1's *vb* counters with and without prejudice filters

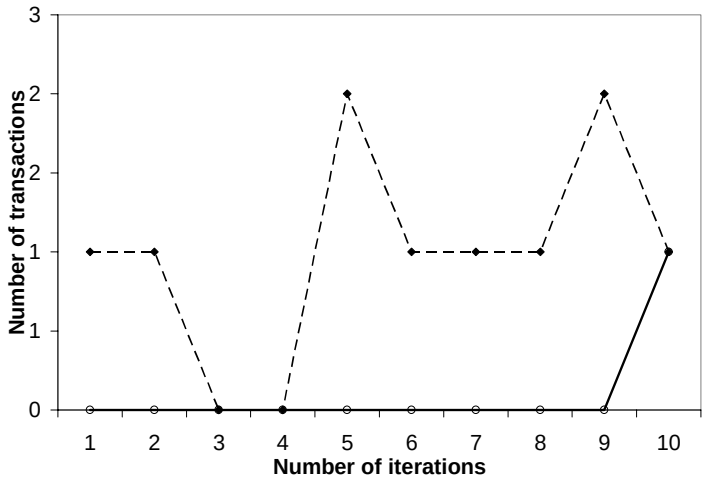

Figure 12.4 Comparing Case Study 1's *b* counters with and without prejudice filters

The bad experiences (Figure 12.4) also tended to remain near a single value when the prejudice filter was not used, for the same reasons that the very bad experiences tended towards a single value. The number of bad experiences was decreased (disappearing altogether at one point) when the prejudice filter was used, as interactions with the agents that gave bad experiences were denied.

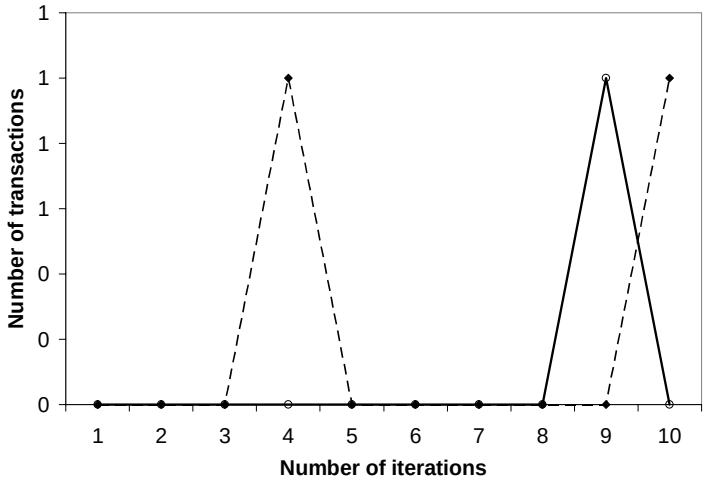

Figure 12.5 Comparing Case Study 1's *g* counters with and without prejudice filters

In both simulations – with the prejudice filter and without the prejudice filter – good experiences (Figure 12.5) were almost nonexistent, due to the fact that the experience results leaned towards the very bad or the very good. There was a slight improvement when the prejudice filter was used across the various simulations, but this can be considered negligible when time and the situational factors of the simulation are taken into consideration. These time and situation factors are the various delays introduced into the simulation by the operating system on which the simulation ran. The delays occurred as a result of the processor's sharing time with the simulation in order for it to accomplish background tasks inherent to the operating system on which the simulation was executed. The operating system on which the prototype implementation was executed is the Windows XP Home edition with Service Pack 2 installed.

Since several agents were simulated on a single machine, and time durations were measured, small delays in the operating system introduced random behaviour into the simulation. This random behaviour reflects real-life situations where the operating system and environment in which an agent resides influence the agent's performance. It is this reasoning that brings about the need for the second case study, which executed exactly the same simulation several times and investigated the patterns that emerged when the delays introduced by the operating system were also considered.

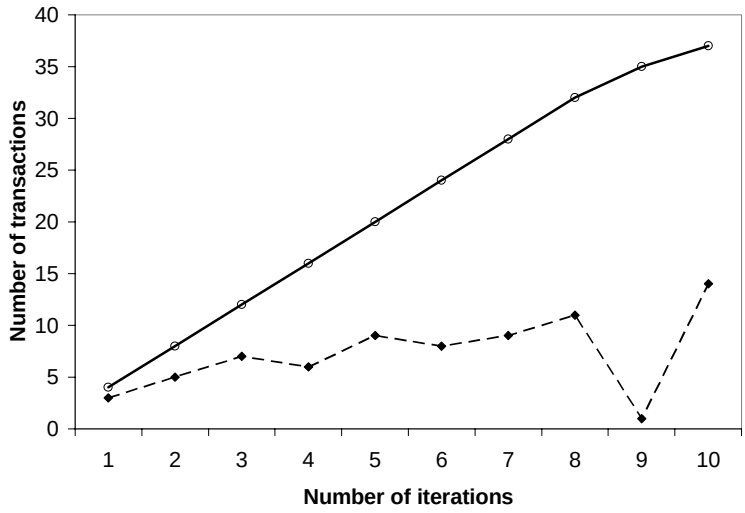

Figure 12.6 Comparing Case Study 1's *vg* counters with and without prejudice filters

The final diagram of the first case study – Figure 12.5 – illustrates a drastic increase in the number of very good experiences when the prejudice filters were used. As the number of iterations increased, the increase in the number of very good experiences was more dramatic when a prejudice filter was used than when not used.

12.2.2 Case Study 2

The second case study ran the simulation ten times without the prejudice filter and ten times with the prejudice filter, each simulation executing for five iterations at a time. This was done to emulate the behaviour exhibited by agents when operating in a dynamic environment. Changes in the environment affected the agent's performance. By allowing changes in the operating system on which the simulation was executed to influence the simulation results, the impact of real-time influences on agent behaviour was also considered. The iteration number chosen for the simulation was five, because it is enough to allow changes in the operating system to influence the simulation results. The operating system shared its time with the simulation and influenced the simulation results by running background processes at various times during the simulation. The simulation expects agents to react in a specific time period. Since the simulation is sharing processing time with the operating system, the operating system sometimes suspends the simulation while it is executing a background task. This suspension influences the results of the

simulation since a given agent may take longer than expected to respond when the simulation is suspended.

12.2.2.1 Case Study 2 simulation results

Tables 12.3 and 12.4 record the results of Case Study 2's simulations without and with prejudice filters respectively. These results were influenced by the operating system on which the simulation was executed and demonstrated changes in trust values.

Table 12.3 Agent A's results for Case Study 2's simulation without prejudice filters

Simulation number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Interaction result										
vb	7	6	6	7	9	7	7	3	8	2
b	0	1	1	0	0	0	0	0	0	2
g	3	3	0	2	4	1	4	3	2	0
vg	0	0	6	1	1	2	2	4	1	7

Table 12.4 Agent A's results for Case Study 2's simulation with prejudice filters

Simulation number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Transaction result										
***vb*(f)**	2	5	4	5	5	1	5	4	4	1
***b*(f)**	0	0	0	0	0	0	0	0	0	0
***g*(f)**	6	15	14	13	11	0	12	12	11	0
***vg*(f)**	12	0	2	2	4	19	3	4	5	19

The data in Tables 12.3 and 12.4 was interpreted in the same way as the data in Tables 12.1 and 12.2. The various values show the number of interactions that resulted in a particular experience classification for all four possible experience grades, as defined by Abdul-Rahman and Hailes (2000). Figures 12.7 and 12.8 illustrate the results graphically.

In the first case study, the graph illustrating the simulation study without the prejudice filters showed a large number of very bad experiences in general; however, the very good experiences still dominated. In this case study, that is no longer the case. Delays caused by the operating

system caused all the agents to respond more slowly, and this ultimately resulted in an increase in very bad experiences. This experiment clearly illustrates the situational influence of trust.

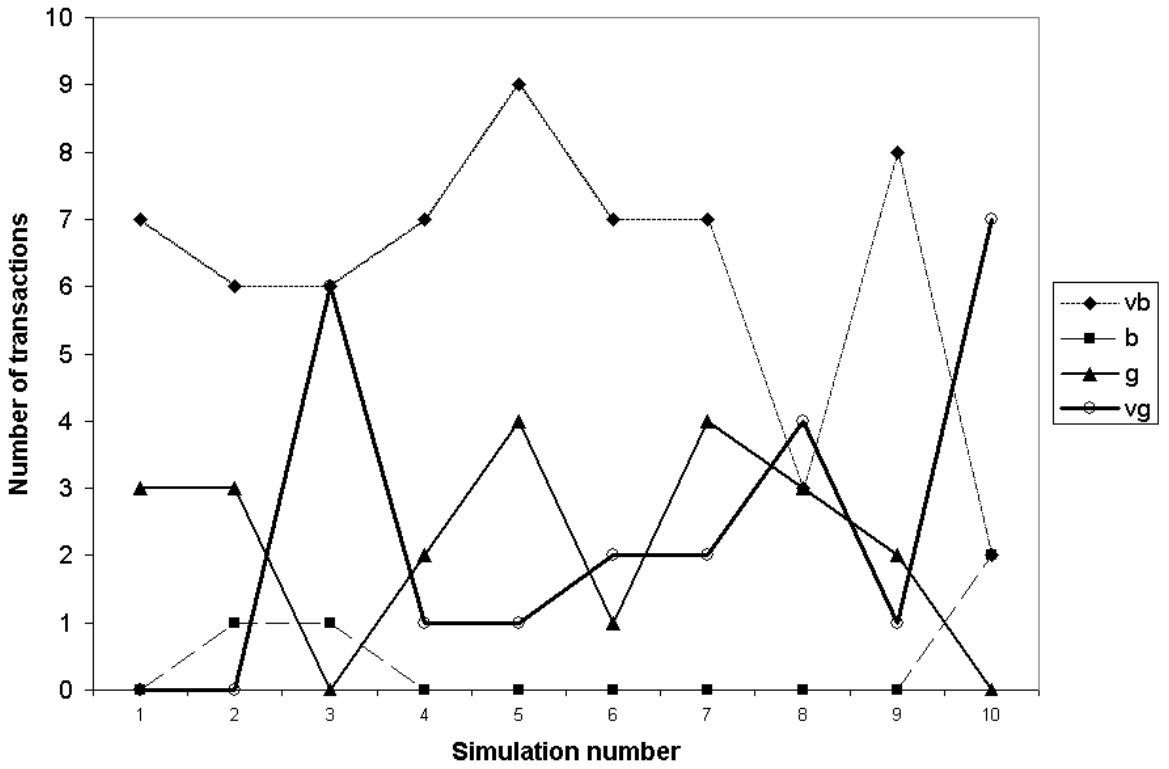

Figure 12.7 Graphic results for simulation Case Study 2 without prejudice filters

As can be seen in the graph in Figure 12.7, the very bad experiences dominated the results in the second case study. The large number of bad experiences in the first study was actually minimised in the second study. However, because delays caused by the operating system had a negative effect on the time it took agents to respond, the bad experiences from the first case study simply became very bad.

Just as in the previous case study, the use of prejudice filters as illustrated by the graph in Figure 12.8 demonstrated an increase in general performance. Because of the various delays that the operating system introduced into the simulation , the results were more unpredictable than in the first case study. However, a consistent increase in very good and good experiences and a decrease in bad and very bad experiences could be identified in both case studies when prejudice filters were used. When prejudice filters were used, both the very good and good experience grades had higher values than those that measured bad and very bad experiences.

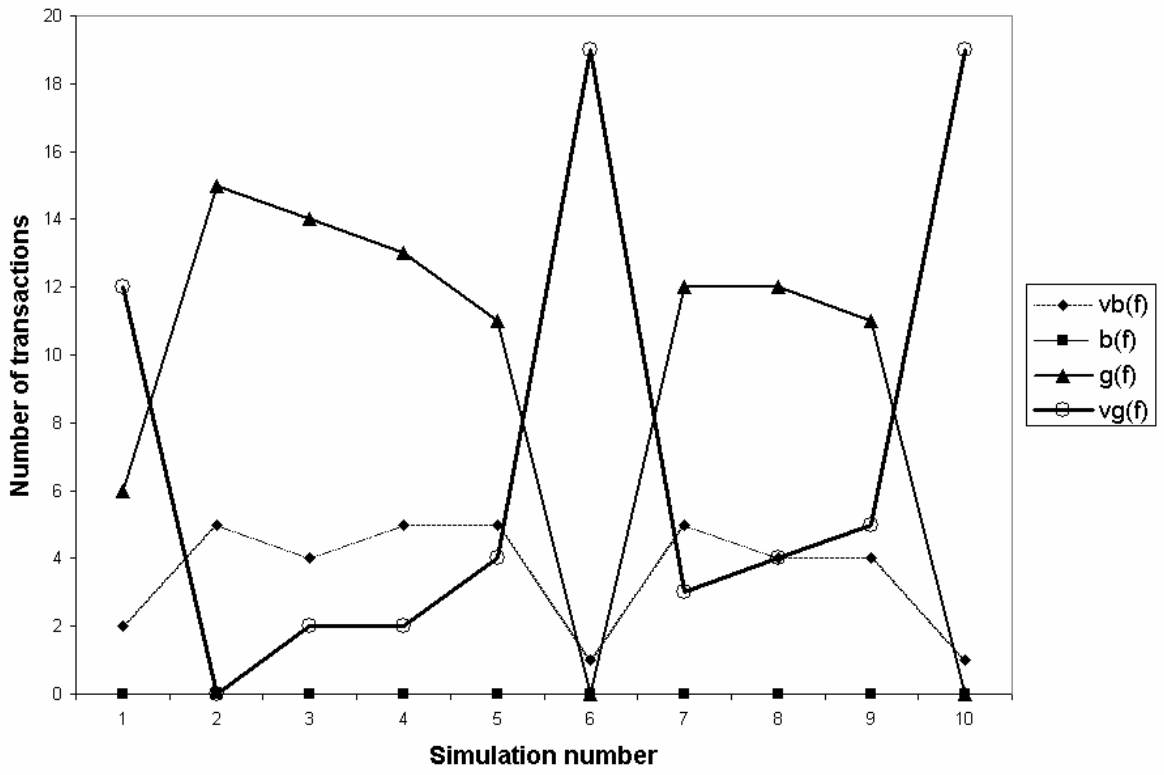

Figure 12.8 Graphic results for simulation Case Study 2 with prejudice filters

12.2.2.2 Comparison of the individual counters for Case Study 2

A comparison of the various experience grades for Case Study 2 is illustrated in Figures 12.9 to 12.12, illustrating how the prejudice filter affected each category. As in the graphs for the previous case study, the bold line represents data recorded with the prejudice filter and the dashed line represents the data recorded without the prejudice filter.

Just as in the first case study, the number of very bad experiences (Figure 12.9) was consistently higher when the simulation was executed without the prejudice filter than with the prejudice filter. Each time the same simulation was executed, the results differed. This is due to the fact that the operating system on which the simulation was executed was allowed to continue with its own tasks and these tasks changed at irregular intervals.

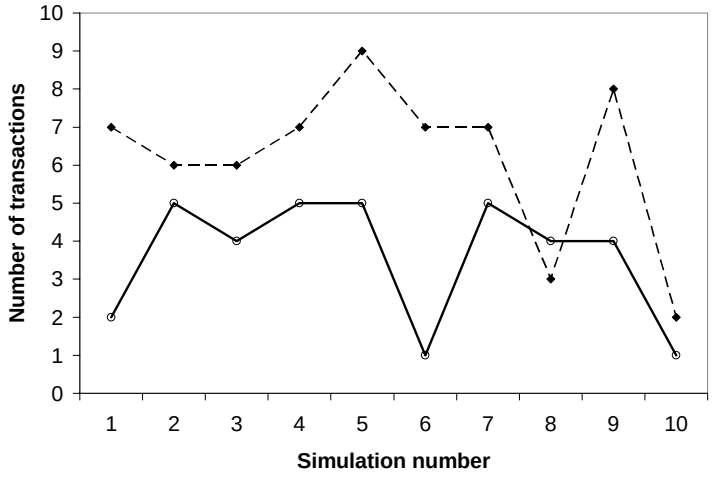

Figure 12.9 Comparing Case Study 2's *vb* counters with and without prejudice filters

The number of bad experiences (Figure 12.10) decreased from the first simulation. This was due to the increase in very bad experiences caused by delays in the operating system. Even though the number of bad experiences decreased, the simulations without the filter still demonstrated a higher tendency to result in bad experiences than those executed with the prejudice filter.

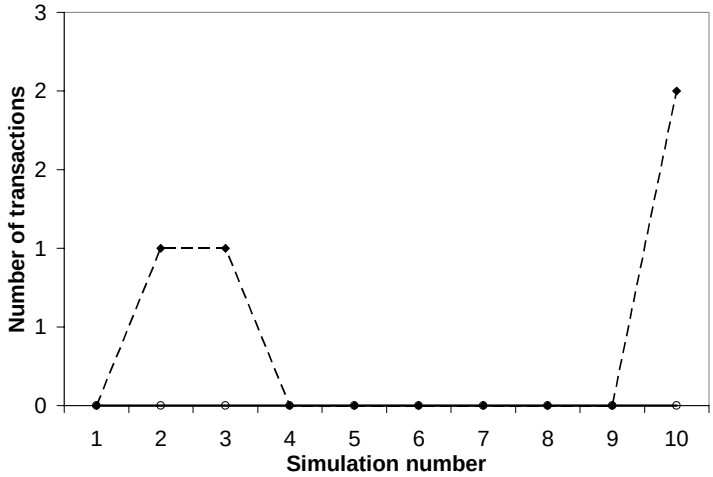

Figure 12.10 Comparing Case Study 2's *b* counters with and without prejudice filters

The number of good experiences, as illustrated in Figure 12.11, increase considerably when the prejudice filter was used. This is due to the fact that the agents that took a long time to respond were filtered out before an interaction with them was instigated. This in turn decreased the likelihood of an interaction that might have resulted in a bad experience.

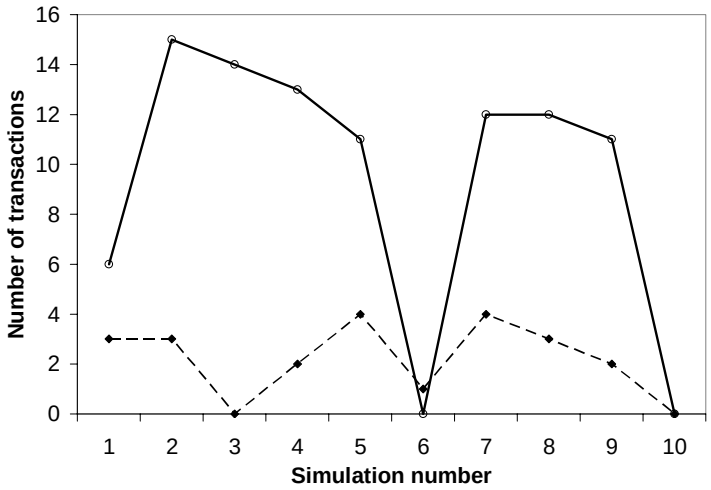

Figure 12.11 Comparing Case Study 2's *g* counters with and without prejudice filters

As with good experiences, the number of very good experiences (Figure 12.12) increased when the prejudice filter was used. Peaked moments of increase in experiences that were very good coincided with the drops in those that were good, resulting in a consistent increase in positive experiences over bad ones overall.

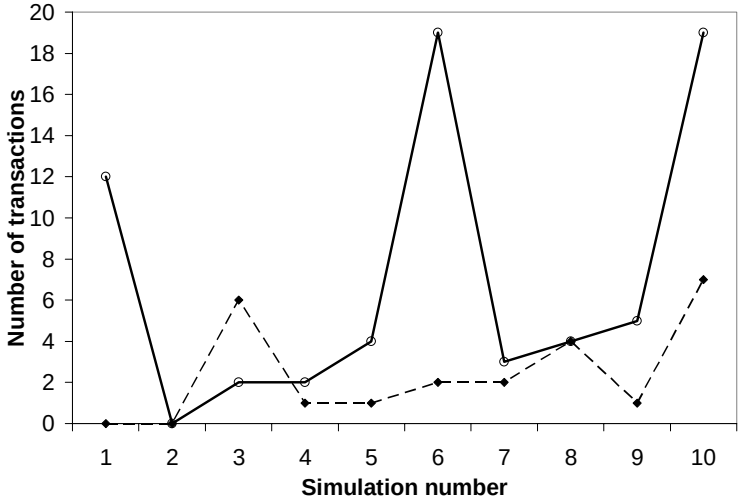

Figure 12.12 Comparing Case Study 2's *vg* counters with and without prejudice filters

Another interesting phenomenon to note was this – not only did the number of good experiences increase, but also the number of interactions that Agent A participated in. By filtering out interactions that resulted in bad experiences, Agent A was left free to handle requests from agents that were more likely to result in good experiences and thus it gained a better reputation among

the other agents. The fact is that Agent A's response to a trusted agent was not delayed by an attempt to service a distrusted agent.

12.2.3 Overall results

The results of both case studies consistently demonstrated an improvement in the interaction results. Each interaction resulted in a specific experience that incremented the counter associated with it. The general interaction results refer to the overall experience grades that resulted from the various interactions that occurred between two agents. In general, the number of experiences that were very good and good respectively increased due to the working of the prejudice filters, while those that were bad and very bad respectively decreased. When one looks at the comparisons of the individual experience grades themselves, one notes that the use of prejudice filters had a marked effect on the working of the trust model. The very good and good experience grade results increased consistently, while the bad and very bad ones decreased, because the domain *Book King*, which resulted in bad experiences due to the long time delays in `ResponseDueTime` and `ConfirmationTime`, was filtered out.

As noted in the second case study, the number of transactions that Agent A received and dealt with increased when the prejudice filters were used, because Agent A built a better reputation for itself among other agents it trusted by responding more quickly. Agent A was able to respond more quickly to trusted agents' requests because it filtered out and did not attempt to deal with distrusted agent's requests. It was consequently available to service the trusted agents as soon as their requests were received. By demonstrating an increase in both the number of interactions dealt with and the number of good and very good experiences, Agent A displayed a performance increase through the use of prejudice filters.

12.3 Conclusion

This chapter looked at simulations to illustrate the impact of prejudice filters on trust. The prototype was created using a modified version of the direct trust proposed in the trust reputation model of Abdul-Rahman and Hailes (2000). The scenario created in Chapter 11 was used in simulations for two case studies. The simulations were executed several times and the results

divided into two simulation case studies. The two case studies were created to illustrate the impact of prejudice filters in two different situations. One case study looked at the effect of trust development over time, while the second case study looked at the effect of the environment in which interactions occur.

The overall results indicate that prejudice filters have a positive impact on trust. The number of failed interactions dropped, and the number of successful interactions increased. The delays that the operating system added to the simulation have a great impact on trust itself, especially in the given simulation where success is measured in time. Even when the system delay, which can be seen as a situational and environmental factor, was allowed to have an impact on the simulation, the use of prejudice filters still improved performance.

PART 6

CONCLUSION

13. Conclusion

13.1 Summary

Several issues have been looked at and discussed in this dissertation, each ultimately working towards the single goal of solving the particular problem at hand. The dissertation is briefly summarised below.

Part 1 (containing the first two chapters) covered the introduction and background of this dissertation. Chapter 1 provided a broad overview of the research topic, the research problem and the structure of this dissertation, and identified the issues that had been addressed. Included in this chapter were definitions for the key terms used throughout the dissertation, as well as the rationale for this study. The theoretical background required for this dissertation was provided in Chapter 2. A basic overview of the vital concept of trust was provided from a psychological perspective, allowing for a better understanding of the concepts that have driven trust model implementations. A basic trust model architecture was defined by the researcher to provide an overview of the structure of trust models in general.

The trust model criteria proposed in this study were discussed in Part 2 of this dissertation. Four categories of trust model criteria were defined and discussed in detail in Chapters 3, 4 and 5, namely trust representation, initial trust, trust updates and trust evaluation. These criteria are based on current implementations of trust models and were defined in order to evaluate trust models. This evaluation identified environmental as well as implementation factors that influence the environment for which a particular trust model is best suited.

Part 3 expanded on the criteria identified in Part 2 and introduced a fifth category: prejudice filters. Prejudice filters are a new concept to trust models and rely on the cognitive definition of prejudice in order to filter out unwanted communications before the lengthy trust evaluation process can occur. Prejudice filters were proposed to lower both the process and communication load required by an agent simply to evaluate another agent which it will ultimately distrust.

Chapter 6 defined and discussed the various prejudice filters identified, and Chapter 7 discussed the various relationships that exist between them.

A more practical approach was taken in Parts 4 and 5 of this dissertation. Part 4, consisting of Chapters 8 and 9, took examples of trust models and used the identified trust model criteria to evaluate the trust model implementations. Reasoning as to why a particular implementation satisfies a particular criterion was also offered. Chapter 8 provided a detailed evaluation of Abdul-Rahman and Hailes's (2000) trust reputation model. Chapter 9 provided only a partial trust analysis of an additional trust model and of a trust-based mechanism design that relies on a particular type of trust model.

Part 5 (Chapters 10, 11 and 12) looked at a prototype, a scenario and simulation respectively in order to investigate the impact of prejudice filters on trust. A simple prototype implementation of trust based on the trust reputation model of Abdul-Rahman and Hailes (2000) was set out in Chapter 10. Chapter 11 provided a scenario to be used for testing the prototype. Two main case studies were conducted, as discussed in Chapter 12, and the results were graphically illustrated and discussed. These graphic results assisted in an analysis of the simulation results, thus illustrating the impact of prejudice filters on trust. It became apparent that prejudice filters have a markedly positive influence on trust, as they increase the number of positive experiences and decrease the number of negative experiences.

The various chapters of this dissertation have worked towards solving the problem identified in Chapter 1. This has been accomplished in an incremental way, where each chapter has built on concepts and solutions introduced in the chapters that preceded it.

13.2 Revisiting the problem

Due to the global growth of the Internet, businesses seek to create a virtual presence to expand their markets worldwide. This shift towards a virtual economy has raised the issue of trust. How do you trust an entity that in essence does not have a physical presence? How do you get others to trust your virtual presence? Numerous experts have proposed various trust models to address these problems (cf. Chapter 2). However, addressing the issue of trust by using trust models

creates its own problems, as evidenced by Chapter 1. This section serves to revisit these problems and to discuss the extent to which they have been addressed.

Trust models rely on human psychological concepts to establish and encourage trust in virtual environments. Trust can exist between any two entities participating in an interaction. The entities that participate in an interaction may be humans, computers and even other devices. The trust models proposed by the various experts focus on different concepts, creating the problem that they are difficult to compare. There was no standardisation between these models prior to this research, making it difficult to evaluate trust models and identify which trust models are better suited for particular environments. The problem created by the lack of standardisation among current trust model implementations was addressed in part in this study by creating a set of evaluation criteria. These criteria addressed the problem of having no means by which to categorise and evaluate trust models.

The criteria further assisted in the identification of strengths and weaknesses of specific trust model approaches. They provided a structure that can be used for further trust model implementations and identified five categories that had been identified as vital components in the implementation of trust models. However, the problem of no standardised structure among trust models was addressed only partially by the various criteria, since these criteria were based on an analysis of current trust models and consequently included only the concepts currently in implementation. The categories identified by the criteria still need to be tested as a framework for further trust model implementations.

Research into the problem identified a further flaw in various approaches towards trust modelling. The flaw identified was that of the processing load that each agent is required to deal with when establishing a trust value for another agent. More detailed trust analysis requires a heavier processing load. This is a costly endeavour, especially in cases where the often lengthy trust evaluation process results in a value that leads to distrust and a rejection of another agent. If a trust evaluation results in a distrust value, processing time and power that could have been used for a successful interaction is wasted on a distrusted agent. Spending time on analysing a large number of agents that result in a distrust value decreases the time an agent could have spent on

providing services to trusted nodes. If an agent is kept busy dealing with distrusted agents, this can also have an impact on the trust other agents have in the agent. This is due to the fact that other agents have to wait for the agent to finish evaluating the distrusted agents before the agent can even accept requests for services from trusted agents. In this study, prejudice filters are therefore proposed to prevent the lengthy evaluations that ultimately result in distrust values.. Prejudice filters are a means by which a quick evaluation can be done, even before a lengthy evaluation process begins, to test for the likelihood that the evaluation will end in distrust. This allows an agent to deny an incoming communication, based on a number of defined prejudiced characteristics, before the trust evaluation process begins, thereby lowering the amount of processing required that would any way result in a distrust value. This consequently allows an agent to dedicate itself to agents it trusts.

13.3 Future work

Even though several issues have been identified and addressed by this dissertation, many questions still remain. The most prominent issue that was addressed is the need for some sort of evaluation criteria for the wide and varied trust model implementations that have been proposed. However, as with any rather new research field, this still leaves much that remains to be done. Points to consider are the extent to which each criterion has been defined and tested. Future work should include more detailed inspection of each criterion. Further testing by evaluating even more varied trust models to test the versatility of the criteria is also required. Detailed research into each criterion needs to be expanded to include a definition of an empirical system that can actually identify the degree to which a particular trust model complies with a given criterion, as well as the efficiency of the particular implementation. Specific metrics that can be used as weights still require definition and will assist in differentiating between trust models that rely on similar basic principles for implementation. The categories identified by the criteria can be used as a generic framework to be used for implementing further trust models. However, this generic framework still needs to be defined explicitly and be tested.

The concept of prejudice filters is defined with current trust model architecture in mind so as to make it simpler to expand current trust models to include prejudice. This study proposes a solution for implementing prejudice filters in trust model architecture and provides a simplified

implementation to illustrate the expected performance increase. Future work in this regard includes defining more detailed and specific implementations of the various types of prejudice filters that have been identified in this study. This includes standardising the protocols required to carry trust-related information so that agents executing different trust model implementations can have a standard way of acquiring and identifying the data that is important to their specific implementations.

13.4 Final conclusion

Chapter 13 provided a summary of the work presented in this dissertation. The reader was reminded of the structure of the dissertation with regard to the content of the various chapters. The problem statement was revisited and discussed with regard to the way in which the problem has been addressed and solved. The solution presented by this dissertation was discussed and areas for future research were identified.

Several publications, which can be found in Appendices C to G, have resulted from this research (Wojcik, Eloff & Venter 2006a, 2006b; Wojcik, Venter & Eloff 2006a, 2006b; Wojcik, Venter, Eloff & Olivier 2005). A paper entitled *Trust-based forensics* (Wojcik, Venter, Eloff & Olivier 2006) was not specifically covered in this dissertation since its topic deviated from the main topic of the dissertation. However, that paper explored alternative uses for trust modelling in the field of Computer Forensics. It looked specifically at incorporating trust and trust models into forensics. Forensic investigators need to keep up with technological trends to obtain the evidence they require for prosecution. The paper sought to define a tool for analysing data in environments where trust model implementations are executed to assist the investigator in identifying the ways in which the trust environment itself influences the presence of evidence of criminal activity. The paper raised its own questions and suggested further research areas, which include a more detailed definition of the tools and issues involved in forensic investigations conducted in trust-based environments.

References

Aberer, K. & Despotovic, Z. 2001. Managing trust in a peer-2-peer information system. In: *Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM'01):* 310--317.

Abdul-Rahman, A. & Hailes, S. 1997. A distributed trust model. In: *New Security Paradigms Workshop, Proceedings of the 1997 Workshop on New security Paradigms*, Langdale, Cumbria, United Kingdom, 48-60.

Abdul-Rahman, A. & Hailes, S. 1999. Relying on trust to find reliable information. In: *Proceedings 1999 International Symposium on Database, Web and Cooperative Systems (DWACOS'99)*, Baden-Baden, Germany, 1-6.

Abdul-Rahman, A. & Hailes, S. 2000. Supporting trust in virtual communities. In: *Proceedings of the 33rd Hawaii International Conference on System Sciences*, Maui, Hawaii, 1-10.

Aberer, K. & Despotovic, Z. 2001. Managing trust in a peer-2-peer information system. In: *Proceedings of the 10th International Conference on Information and Knowledge Management*, New York, USA: ACM Press, 310-317.

Agarwal, D., Thompson, M., Perry, M. & Lorch, M. 2003. A new security model for collaborative environments, In: *Proceedings of the Workshop on Advanced Collaborative Environments*, Seattle, WA, 1-7.

Axelrod, R. 1984. *The evolution of cooperation*. New York: Basic Books.

Azzedin, F. & Maheswaran, M. 2003. Trust modeling for peer-to-peer based computing systems. In: *Proceedings of the International Parallel and Distributed Processing Symposium*, Washington, DC, USA: IEEE Computer Society, 1-10.

Bagley, C., Verma, G.K., Mallick, K. & Young, L. 1979. *Personality, self-esteem and prejudice*, Farnborough: Saxon House.

Barber, B. 1983. *Logic and limits of trust*. New Brunswick. New Jersey: Rutgers University Press.

Beth, T. et al. 1994. Valuation of Trust in Open Networks. In: *Proceedings of the European Symposium on Research in Computer Security, Lecture Notes in Computer Science* 875: 3-18.

Blaze, M., Feigenbaum, J., Ioannidis, J. & Keromytis, A.D. 1999. The role of trust management in distributed systems security. In: *Secure Internet Programming: Security Issues for Mobile and Distributed Objects*, Lecture Notes in Computer Science, London, UK: Springer-Verlag, 1603:185-210.

Bohnet, I. & Zechhauser, R. 2004. Trust, risk and betrayal. *Journal of Economic Behavior and Organization*,, 55(4):467-484.

Boon, S.D. & Holmes, J.G. 1991. The dynamics of interpersonal trust: resolving uncertainty in the face of risk. In: Hinde, R.A. & Groebel, J. (eds), *Cooperation and Prosocial Behaviour*, Cambridge: Cambridge University Press, 190-211.

Bowling, M. & Veloso, M. 2002. Multiagent learning using a variable learning rate. *Artificial Intelligence*, Elsevier, 136:215-250.

Britner, M.J. 1992. Servicescapes: the impact of physical surroundings on customers and employees. *Journal of Marketing*, 54(2):69-82.

Britner, M.J., & Zeithaml, V.A.. 2006. Servicescape. *Wikipedia*. Available from http://en.wikipedia.org/wiki/Servicescape [Cited 14 June 2006].

Buchegger, S. and Le Boudec, J.Y. 2004. A Robust Reputation System for P2P and Mobile Ad-hoc Networks. In: *Proceedings of the 2nd Workshop in the Economics of Peer-to-Peer Systems*: 1-6.

Carbone, M., Nielsen, M. & Sassone, V. 2003. A formal model for trust in dynamic networks. In: *Proceedings of the First International Conference on Software Engineering and Formal Methods*, Washington, DC, USA: IEEE Computer Society, 54-61.

Chakraborty, S. & Ray, I. 2007. p-Trust: A New Model of trust to Allow Finer Control Over Privacy in Peer-to-Peer Framework. In: *Journal of Computers* 2(2): 13-24.

Dasgupta, D. 1997. Artificial neural networks and artificial immune systems: similarities and differences. In: *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, Orlando, FL, USA, 12-15.

Dash, R.K., Ramchurn, S.D. & Jennings, N.R. 2004. Trust-based mechanism design. In: *Proceedings from the 3rd International Conference on Autonomous Agents and Multi- Agent Systems- Volume 2*, Washington, DC, USA: IEEE Computer Society, 748-755.

Datta, A., Hauswirth, M. & Aberer, K. 2003. Beyond 'web of trust': Enabling P2P E-commerce. In: *E-Commerce, IEEE International Conference*, Washington, DC, USA: IEEE Computer Society, 303-312.

Deutsch, M. 1962. Cooperation and trust: Some theoretical notes. In: Jones, M.R. (ed.), *Nebraska Symposium on Motivation*. Lincoln, NE: Nebraska University Press, 275-319.

Doney, P. & Cannon, J. 1997. An examination of the nature of trust in buyer-seller relationships. *Journal of Marketing*, 61:35-51.

Dragovic, B., Kotsovinos, E. & Pietzuch, P.R. 2003. XenoTrust: Event-based distributed trust management. In: *Proceedings of the Second IEEE International Workshop on Trust and Privacy*

in Digital Business (DEXA-TrustBus'03), Washington, DC, USA: IEEE Computer Society, 410-414.

Earley, P.C. & Gibson, C.B. 1998. Taking stock in our progress on individualism-collectivism: 100 years of solidarity and community. *Journal of Management*, Elsevier, 24(3):265-304.

English, C., Nixon, P., Terzis, S., McGettrick, A. & Lowe, H. 2002. Dynamic trust models for ubiquitous computing environments. In: *Workshop on Security in Ubiquitous Computing, UBICOMP 2002*, Göteborg, Sweden, 1-4.

Esfandiari, B. & Chandrasekharan, S. 2001. On how agents make friends: Mechanisms for trust acquisition. In: *4th Workshop on Deception, Fraud and Trust in Agent Societies*, Montreal, Canada, 27-34.

Fiske, S.T. 2000. Stereotyping, prejudice, and discrimination at the seam between the centuries: Evolution, culture, mind and brain. *European Journal of Social Psychology*, 30:299-322.

Forouzan, B.A. & Fegan, S.C. 2003. *TCP/IP Protocol Suit* (2nd ed.). New-York: McGraw-Hill.

Fujiwara, C. *Disintermediated!* 22 December 2000. Available from http://www.hermenaut.com /a54.shtml [Cited 14 June 2006].

Gambetta, D. 1990a. Can we trust trust? In: Gambetta, D. (ed.), *Trust*, Oxford: Blackwell, 213-237.

Gambetta, D. (ed.) 1990b. *Trust*. Oxford: Blackwell.

Glassner, B. 1980. *Essential interactionism*. London: Boston & Henley, Routledge & Kegan Paul.

Grandison, T. & Sloman, M. 2000. A survey of trust in Communications Surveys and Tutorials 3: 2–16

Guha, R., Kumar R., Raghaven, P. & Tomkins, A. 2004. Propagation of trust and distrust. In: *Proceedings of the 13th International Conference on World Wide Web,* New York, NY, USA, 403-412.

http://framework.v2.nl/archive/archive/node/text/default.xslt/nodenr-156647 [Cited 24 February 2007].

http://www.mcwdn.org/ECONOMICS/EcoGlossary.html [Cited 04 December 2006].

http://www.microsoft.com/windows2000/techinfo/howitworks/activedirectory/glossary.asp [Cited 04 December 2006].

http://www.pmostep.com/290.1TerminologyDefinitions.htm [Cited 24 February 2007].

Huberman, B. and Wu F. 2004. The Dynamics of Reputation. In: *Journal of Statistical Mechanics: Theory and Experiment,* April 2004: 1-17.

Hultkrantz, O. & Lumsden, K. 2001. E-commerce and consequences for the logistics industry. In: *Proceedings for Seminar on the Impact of E-Commerce on Transport,* Paris, France, 1-13.

Huynh, T.D., Jennings, N.R. & Shadbolt, N.R. 2004. FIRE: An integrated trust and reputation model for open multi-agent systems. In: *Proceedings of 16th European Conference on Artificial Intelligence*, Valencia, Spain, 18-22.

Jones, S. 1990. A discussion of issues and systems relevant to computer supported cooperative work. In: *Tech. rept. 64*. University of Stirling, Department of Computing Science and Mathematics.

Jones, A. and Firozabadi, B. 2000. On the Characterization of a Trusting Agent - Aspects of a Formal Approach. Castelfranchi and Tan eds., Trust and Deception in Virtual Societies, Kluwer Academic Publishers 2001: 157-168.

Jonker, C.M. & Treur, J. 1997. Compositional verification of multi-agent systems: a formal analysis of pro-activeness and reactiveness. In: *Lecture Notes in Computer Science, Revised Lectures from the International Symposium on Compositionality: The Significant Difference,* London, UK: Springer-Verlag, 1536:350-380.

Jonker, C.M. & Treur, J. 1999. Formal analysis of models for the dynamics of trust based on experiences. In: *Proceedings of the 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World : Multi-Agent System Engineering.* London, UK: Springer-Verlag, 1647:221-231.

Jøsang, A. 1996. The right type of trust for distributed systems. In: *New Security Paradigms Workshop: Proceedings of the 1996 New Security Paradigms Workshop,* Lake Arrowhead, California, United States, 119-131.

Jøsang, A. 1997. Prospectives for modelling trust in information security. In: *Australasian Conference on Information Security and Privacy*, Sydney, Australia, 2-13.

Jøsang, A. & Tran, N. 2000. Trust Management for E-commerce. Available from http://citeseer.ist.psu.edu/375908.html [Cited 9 October 2006].

Jøsang, A. et al. 2000. PKI Seeks a Trusting Relationship. In: *Ed Dawson, Andrew Clark, and Colin Boyd, editors, Proceedings of the 2000 Australasian Conference on Information Security and Privacy (ACISP2000)*: 1-14.

Kamvar, S. et al. 2003. The Eigen Trust Algorithm for Reputation Management in P2P Networks. In: *Proceedings of 12th International Conference on World Wide Web*: 640-651.

Khare, R. & Rifkin, A. 1997. Weaving a web of trust. *World Wide Journal*, Sebastopol, CA, USA: O'Reilly & Associates, Inc., 2(3):77-112.

Lamsal, P. 2001. Understanding trust and security. Available from http://wiki.uni.lu/secan-lab/Unders UnderstandingTrustAndSecurity.pdf [Cited 25 April 2005].

Langheinrich, M. 2003. When trust does not compute – the role of trust in ubiquitous computing. In: *Proceedings of the Privacy Workshop of Ubicomp*, Seattle, Washington, 1-8.

Li, X. & Liu, L. 2004. PeerTrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, Washington, DC, USA: IEEE Computer Society, 16(7):843-857.

Li, X., Lyu, M.R. & Liu, J. 2004. A trust model based routing protocol for secure ad hoc networks. In: *IEEE Aerospace Conference Proceedings*, Washington, DC, USA: IEEE Computer Society, 2:1286-1295.

Li, X., Valacich, J.S. & Hess, T.J. 2004. Predicting user trust in information systems: A comparison of competing trust models. In: *Proceedings of the 37th Hawaii International Conference on Systems Sciences*, Predicting user trust in information systems, 1-10.

Liang, Z. & Shi, W. 2005. PET: A PErsonalized Trust model with reputation and risk evaluation for P2P resource sharing. In: *Proceedings of the 38th Hawaii International Conference on Systems Sciences*, Hawaii, 1-10.

Linn, J. 2000. Trust models and management in public-key infrastructures. In: *Technical Report, RSA Data Security*, Inc., Redwood City, CA, USA.

Luce, R.D. & Raiffa, H. 1957. *Games and decisions : Introduction and critical survey*. New York: Wiley.

Luhmann, N. 1979. *Trust and power*. Chichester: Wiley.

Luo, H., Zerfos, P., Kong, J., Lu, S. & Zhang, L. 2002. Self-securing ad hoc wireless networks. In: *Proceedings of the Seventh IEEE Symposium on Computers and Communications*, Washington, DC, USA: IEEE Computer Society, 567-574.

Malsch, T., Florian, M., Jonas, M. & Schulz-Schäfer, I. 1996. Expeditionen ins Grenzgebiet zwischen Soziologie und Künstlicher Intelligenz. In: *Künstliche Intelligenz*, 2:6-12.

Manchala, D. 2000. E-Commerce trust metrics and models. In: *IEEE Internet Computing*, Washington, DC, USA: IEEE Computer Society, 36-44.

Marsh, S.P. 1994. Formalising trust as a computational concept. Dissertation for the Department of Computing Science and Mathematics, University of Stirling, 1-184.

Marx, M. & Treur, J. 2001. Trust dynamics formalised in temporal logic. In: *Proceedings of the Third International Conference on Cognitive Science*, Beijing, China, 359-362.

McKnight, D.H., Choudhury, V. & Kacmar, C. 2002. Developing and validating trust measures for e-commerce: An integrative typology. *Information Systems Research*, Linthicum, Maryland, USA: Institute for Operations Research and the Management Sciences, 13(3):334-359.

Montaner, M., Lopez, B. & Lluis de la Rosa, J. 2002. Opinion-based filtering through trust. In: *Proceedings of the 6th International Workshop on Cooperative Information Agents VI.* Lecture Notes in Computer Science, London UK: Springer-Verlag, 2446:164-178.

Mui, L., Halberstadt, A. & Mohtashemi, M. 2002. Notions of reputation in multi-agents systems: A review. In: *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems*, New York, USA: ACM Press, 280-287.

Mui, L., Mohtashemi, M. & Halberstadt, A. 2002. A computational model of trust and reputation. In: *Proceedings of the 35th Hawaii International Conference on System Sciences*, Washington, DC, USA: IEEE Computer Society, 2431-2439.

Nilsson, N.J. 1998. *Artificial Intelligence: A new synthesis*. San Francisco: Morgan Kauffmann.

Nooteboom, B. 2002. *Trust: forms, foundations, functions, failures, and figures.* Cheltenham: Edward Elgar.

Panther, S., Erwin, G. & Remenyi, D. 2003. Measuring e-commerce effectiveness: a conceptual model. In: *Proceedings of the 2003 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on Enablement through Technology*, New York, USA: ACM Press, 47:143-152.

Papadopoulou, P., Andreou, A., Kanellis, P. & Martakos, D. 2001. Trust and relationship building in electronic commerce. *Internet Research: Electronic Networking Applications and Policy*, MCB UP Ltd, 11(4):322-332.

Patton, M.A. & Jøsang, A. 2004. Technologies for trust in electronic commerce. *Electronic Commerce Research*, Norwell, MA, USA: Kluwer Academic Publishers, 4:9-21.

Perlman, R. 1999. An overview of PKI trust models. *IEEE Network*, Washington, DC, USA: IEEE Computer Society,13(6):38-43.

Pirzada, A.A. & McDonald, C. 2004. Establishing trust in pure ad-hoc networks. In: *Proceedings of the 27th Australasian Conference on Computer Science.* Darlinghurst, Australia, Australia: Australian Computer Society, Inc.,,26:47-54.

Purser, S. 2001. A Simple Graphical Tool for Modeling Trust. In: *Computers and Security* 20(6): 479-484.

Ramchurn S.R., Sierra, C., Jennings, N.R. & Godo, L. 2003. A computational trust model for multi-agent interactions based on confidence and reputation. In: *Proceedings of 6th International Workshop of Deception, Fraud and Trust in Agent Societies*, Melbourne, Australia, 69-75.

Ramchurn, S.D., Huynh, D. & Jennings, N.R. 2004. Trust in multi-agent systems. In: *The Knowledge Engineering Review*, New York: Cambridge University Press, 19(1):1-25.

Ramchurn, S.R., Jennings, N.R., Sierra, C. & Godo, L. 2004. Devising a trust model for multiagent interactions using confidence and reputation. *Applied Artificial Intelligence*, New York, USA: ACM Press, 18(9-10):833-852.

Rangan, P. 1988. An Axiomatic Basis of Trust in Distributed Systems. In: *Proceedings of the 1988 IEEE Security and Privacy Symposium*: 204-211.

Ray, I. and Chakraborty, C. 2004. A Vector Model of Trust for Developing trustworthy Systems. In: *Proceedings of the European Symposium on Research in Computer Security, Lecture Notes in Computer Science* 3193: 260-275.

Schillo, M., Funk, P. & Rovatsos, M. 2000. Using trust for detecting deceitful agents in artificial societies. *Applied Artficial Intelligence, Special Issue on Trust, Deception and Fraud in Agent Societies*, Taylor and Francis Ltd,14(8):825 848.

Schmidt, A., Beigl, M. & Gellersen, H.W. 1999. There is more to context than location. In: *Computers and Graphics*, Elsevier, 23(6):893-901.

Shand, B., Dimmock, N. & Bacon, J. 2004. Trust for ubiquitous, transparent collaboration. In: *Wireless Networks*, 10(6):711-721.

Shrobe, H., Doyle, J. & Szolovitz, P. 1999. Active trust management for autonomous adaptive survivable systems. In: *Proposal to the Defense Advanced Research Projects Agency in response*

to BAA #00-15, Information Assurance and Survivability (IA&S) of the Next Generation Information Ifrastructure (NGII), 1-29.

Simpson, G.E. & Yinger, J.M. 1985. *Racial and cultural minorities: An analysis of prejudice and discrimination*. (5th ed.). New York: Plenum.

Siyal, M.Y. & Barkat, B. 2002. A novel trust service provider for the internet based commerce applications. *Internet Research: Electronic Networking Applications and Policy*, MCB UP Ltd, 12(1):55-65.

Spekman, R.E. & Davis, E.W. 2004. Risky business: expanding the discussion on risk and the extended enterprise. *International Journal of Physical & Logistics Management*, Emerald Group Publishing Limited, 34(5):414-433.

Stahl, G.K & Sitkin, S.B. 2005. Trust in mergers and acquisitions. In: Stahl, G.K. & Mendenhall, M. (eds) *Mergers and acquisitions: Managing culture and human resources*. Stanford University Press, 1-18.

Stakhanova, N. et al. 2004. A Reputation Based trust Management in Peer-to-Peer Network Systems. In: *Proceedings of Parallel and Distributed Computing conference*: 1-6.

Stoneburner, G. 2001. Underlying technical models for information technology security. In: *National Institute of Standards and Technology Special Publication 800-33, Draft Version 0.2*, Washington DC, USA: U.S. Government Printing Oce.

Sung, H. & Yuan, S.T. 2001. A learning-enabled integrative trust model for e-markets. In: *Papers from the Fifth International Conference on Autonomous Agents Workshop on Deception, Fraud and Trust in Agent Societies*, 81-96.

Teacy, W.T.L., Patel, J., Jennings, N.R & Luck, M. 2005. Coping with inaccurate reputation sources: Experimental analysis of a probabilistic trust model. In: *Proceedings of the Fourth*

International Joint Conference on Autonomous Agents and Multiagent Systems, Utrecht, Netherlands, 997-1004.

Tenenbaum, J.M., Chowdhry T.S. & Hughes, K. 1997. Eco System: an Internet commerce architecture. In: *Computer*, Los Alamitos, CA, USA: EEE Computer Society Press, 30(5):48-55.

Twigg, A. & Dimmock, N. 2003. Attack-resistance of computational trust models. In: *Proceedings of the Twelfth IEEE Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises* WET, Washington, DC, USA: IEEE Computer Society, 275-280.

Wang, Y. & Vassileva, J. 2004. Bayesian network-based trust model. In: *Proceedings of Web Intelligence*, Washington, DC, USA: IEEE Computer Society, 341-348.

Wen, W. & Mizoguchi, F. s.a. An authorization-based trust model for multi-agent systems. Available from http://www.istc.cnr.it/T3/download/aamas1999/Wen-Mizoguchi.pdf [Cited 9 October 2006].

Whitby, A. et al. 2005. Filtering out Unfair Ratings in Bayesian Reputation Systems. In: *Infain Journal of Management Research* 4(2): 48-64, February 2005.

Witkowski, M., Artikis, A. & Pitt, J. 2000. Trust and cooperation in a trading society of objective-trust based agents. In: *Workshop on Deception, Fraud and Trust in Agent Societies at Autonomous Agents*, Barcelona, Spain, 127-135.

Wojcik, M., Eloff, J.H.P. & Venter, H.S. 2006. Trust model architecture: Defining prejudice by learning. In: *Lecture Notes in Computer Science: Trust, Privacy, and Security in Digital Business*, London UK: Springer-Verlag, 4083:182-191.

Wojcik, M., Eloff, J.H.P. & Venter, H.S. 2006. Trust model architecture: Defining prejudice by learning. In: *Proceedings of the 3rd International Conference on Trust, Privacy & Security in Digital Business (TrustBus)*, Krakow, Poland, 1-10.

Wojcik, M., Venter, H.S., Eloff, J.H.P. & Olivier, M.S. 2005. Incorporating prejudice into trust models to reduce network overload. In: *Proceedings of SATNAC: South African Telecommunication Networks & Applications Conference*, Champagne Sports Resort, Drakensberg, South Africa, 1-6.

Wojcik, M., Venter, H.S., Eloff, J.H.P. & Olivier, M.S. 2006. Trust-based forensics: Applying machine trust models to forensic investigation. In: *Fifth Annual Digital Forensics Conference*, Orlando, Florida, USA, 1-20.

Wojcik, M., Venter, H.S. & Eloff, J.H.P. 2006a. Trust model evaluation criteria: A detailed analysis of trust evaluation. WIP. In: *Proceedings of ISSA: Information Security*, Sandton, South Africa, 1-10.

Wojcik, M., Venter, H.S. & Eloff, J.H.P. 2006b. A detailed analysis of trust representation as a trust model evaluation criterion. In: *Proceedings of SATNAC: South African Telecommunication Networks & Applications Conference*, Spier Conference Centre, Stellenbosch, South Africa, 1-6.

Xiong, L. & Liu, L. 2003. A reputation-based trust model for peer-to-peer ecommerce communities. In: *Proceedings of the IEEE Conference on Electronic Commerce*, Newport Beach, CA, USA, 275-284.

Yang, Y., Brown, L. & Lewis, E. 2001. eCommerce Trust via the Proposed W3 Trust Model. In: *PACCS01Conference Proceedings*, 9-14.

Yi, B.C., Corbitt, B. & Thanasankit, T. 2002. Trust and Consumers in B2C eCommerce. In: *School Working Papers Series*. Available from http://www.deakin.edu.au/buslaw/infosys/docs/workingpapers/archive/Working_Papers_2002/2002_05_Corbitt.pdf [Cited 9 October 2006].

Yu, B. & Singh, M.P. 2002. Distributed reputation management for electronic commerce. In: *Computational Intelligence*, 18(4):535-549.

Appendix A

Output without prejudice filters

SimulationLog.txt without prejudice filters

Starting simulation at: [17:38:37.7591824]

[17:38:37.8993840] ITERATION 1

All agents have been pooled and are waiting to request...
[17:38:38.1197008]: [b] REQUEST QUEUE: Begin
[17:38:38.4902336]: [b] REQUEST QUEUE: End
[17:38:38.7205648]: [b] REQUEST 1: Begin
[17:38:38.9208528]: [b] REQUEST 1: Searching for all known suitable agents.
[17:38:39.0610544]: [b] REQUEST 1: Found no known suitable agents.
[17:38:39.2012560]: [b] REQUEST 1: No known agents could service the request.
[17:38:39.3014000]: [b] REQUEST 1: Searching for all unknown suitable agents.
[17:38:39.4716448]: [b] REQUEST 1: Found 6 unknown suitable agent(s).
[17:38:39.6218608]: [b] REQUEST 1: Unknown suitable agents: {a d c g e f}.
[17:38:39.7921056]: [a] ACCEPT 1: Begin
[17:38:40.3328832]: [a] ACCEPT 1: Servicing request from b
[17:38:40.5031280]: [a] ACCEPT 1: Start Service for b [Estimated Response Duration: 100 ms]
[17:38:40.7434736]: [a] ACCEPT 1: End Service for b [Actual Response Duration: 240.3456 ms]
[17:38:41.0439056]: [a] ACCEPT 1: Waiting for confirmation from b
[17:38:41.2542080]: [b] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 902 ms]
[17:38:42.3057200]: [b] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 911.3104 ms]
[17:38:42.4959936]: [a] ACCEPT 1: Confirmation received from b [Total Duration: 1241.7856 ms]
[17:38:42.6361952]: [a] ACCEPT 1: Updating direct experience
[17:38:42.8264688]: [a] ACCEPT 1: Experience with b was VeryBad
[17:38:43.0367712]: [a] ACCEPT 1: Previous experience:
[17:38:43.8779808]: [a] ACCEPT 1: Updated experience: a b Booking 0 0 0 1 VeryBad
[17:38:44.0081680]: [a] ACCEPT 1: End (Successful) [Total Duration: 4055.832 ms]
[17:38:44.1283408]: [b] REQUEST 1: Accepted and serviced by a [Service Duration: 4336.2352 ms]
[17:38:44.2384992]: [b] REQUEST 1: Experience with a was VeryBad
[17:38:44.3486576]: [b] REQUEST 1: Previous experience:
[17:38:45.1998816]: [b] REQUEST 1: Updated experience: b a Booking 0 0 0 1 VeryBad
[17:38:45.4902992]: [b] REQUEST 1: Completed
[17:38:45.6805728]: [b] REQUEST 1: End (Successful) [Total Duration: 6960.008 ms]
[17:38:48.0239424]: [a] REQUEST QUEUE: Begin
[17:38:48.3944752]: [a] REQUEST QUEUE: End
[17:38:48.5146480]: [a] REQUEST 1: Begin
[17:38:48.6848928]: [a] REQUEST 1: Searching for all known suitable agents.

[17:38:48.8651520]: [a] REQUEST 1: Found no known suitable agents.
[17:38:49.0053536]: [a] REQUEST 1: No known agents could service the request.
[17:38:49.1755984]: [a] REQUEST 1: Searching for all unknown suitable agents.
[17:38:49.3358288]: [a] REQUEST 1: Found 6 unknown suitable agent(s).
[17:38:49.4860448]: [a] REQUEST 1: Unknown suitable agents: {d b c g e f}.
[17:38:49.7163760]: [d] ACCEPT 1: Begin
[17:38:50.2170960]: [d] ACCEPT 1: Servicing request from a
[17:38:50.4173840]: [d] ACCEPT 1: Start Service for a [Estimated Response Duration: 1000 ms]
[17:38:51.6391408]: [d] ACCEPT 1: End Service for a [Actual Response Duration: 1221.7568 ms]
[17:38:51.9395728]: [d] ACCEPT 1: Waiting for confirmation from a
[17:38:52.1198320]: [a] REQUEST 1: Start Confirmation for d [Estimated Confirmation Duration: 100 ms]
[17:38:52.3802064]: [a] REQUEST 1: End Confirmation for d [Actual Confirmation Duration: 100.144 ms]
[17:38:52.5404368]: [d] ACCEPT 1: Confirmation received from a [Total Duration: 420.6048 ms]
[17:38:52.7807824]: [d] ACCEPT 1: Updating direct experience
[17:38:52.9309984]: [d] ACCEPT 1: Experience with a was VeryGood
[17:38:53.1112576]: [d] ACCEPT 1: Previous experience:
[17:38:53.8923808]: [d] ACCEPT 1: Updated experience: d a Booking 1 0 0 0 VeryGood
[17:38:54.0425968]: [d] ACCEPT 1: End (Successful) [Total Duration: 4165.9904 ms]
[17:38:54.1928128]: [a] REQUEST 1: Accepted and serviced by d [Service Duration: 4476.4368 ms]
[17:38:54.3430288]: [a] REQUEST 1: Experience with d was Bad
[17:38:54.4932448]: [a] REQUEST 1: Previous experience:
[17:38:55.3644976]: [a] REQUEST 1: Updated experience: a d Booking 0 0 1 0 Bad
[17:38:55.6348864]: [a] REQUEST 1: Completed
[17:38:55.7650736]: [a] REQUEST 1: End (Successful) [Total Duration: 7250.4256 ms]
[17:38:59.0297680]: [f] REQUEST QUEUE: Begin
[17:38:59.5204736]: [f] REQUEST QUEUE: End
[17:38:59.6907184]: [f] REQUEST 1: Begin
[17:38:59.8509488]: [f] REQUEST 1: Searching for all known suitable agents.
[17:39:00.0011648]: [f] REQUEST 1: Found 1 known suitable agent(s).
[17:39:00.1714096]: [f] REQUEST 1: Known suitable agents: {a}.
[17:39:00.3917264]: [a] ACCEPT 2: Begin
[17:39:00.7121872]: [a] ACCEPT 2: Servicing request from f
[17:39:00.9024608]: [a] ACCEPT 2: Start Service for f [Estimated Response Duration: 100 ms]
[17:39:01.2028928]: [a] ACCEPT 2: End Service for f [Actual Response Duration: 300.432 ms]
[17:39:01.5533968]: [a] ACCEPT 2: Waiting for confirmation from f
[17:39:01.6835840]: [f] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 100 ms]
[17:39:01.9339440]: [f] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 100.144 ms]
[17:39:02.1442464]: [a] ACCEPT 2: Confirmation received from f [Total Duration: 460.6624 ms]
[17:39:02.2844480]: [a] ACCEPT 2: Updating direct experience

[17:39:02.4546928]: [a] ACCEPT 2: Experience with f was VeryGood
[17:39:02.6049088]: [a] ACCEPT 2: Previous experience:
[17:39:03.3960464]: [a] ACCEPT 2: Updated experience: a f Booking 1 0 0 0 VeryGood
[17:39:03.5462624]: [a] ACCEPT 2: End (Successful) [Total Duration: 2984.2912 ms]
[17:39:03.6664352]: [f] REQUEST 1: Accepted and serviced by a [Service Duration: 3274.7088 ms]
[17:39:03.8867520]: [f] REQUEST 1: Experience with a was VeryBad
[17:39:04.0269536]: [f] REQUEST 1: Previous experience: f a Booking 1 0 0 0 VeryGood
[17:39:04.4575728]: [f] REQUEST 1: Updated experience: f a Booking 1 0 0 1 EquallyGoodAndBad
[17:39:04.7379760]: [f] REQUEST 1: Completed
[17:39:04.9783216]: [f] REQUEST 1: End (Successful) [Total Duration: 5287.6032 ms]
[17:39:10.0255792]: [g] REQUEST QUEUE: Begin
[17:39:10.4662128]: [g] REQUEST QUEUE: End
[17:39:10.6064144]: [g] REQUEST 1: Begin
[17:39:10.7866736]: [g] REQUEST 1: Searching for all known suitable agents.
[17:39:10.9869616]: [g] REQUEST 1: Found no known suitable agents.
[17:39:11.1572064]: [g] REQUEST 1: No known agents could service the request.
[17:39:11.3174368]: [g] REQUEST 1: Searching for all unknown suitable agents.
[17:39:11.5077104]: [g] REQUEST 1: Found 3 unknown suitable agent(s).
[17:39:11.6579264]: [g] REQUEST 1: Unknown suitable agents: {a d f}.
[17:39:11.7981280]: [a] ACCEPT 3: Begin
[17:39:12.1286032]: [a] ACCEPT 3: Servicing request from g
[17:39:12.2788192]: [a] ACCEPT 3: Start Service for g [Estimated Response Duration: 100 ms]
[17:39:12.5992800]: [a] ACCEPT 3: End Service for g [Actual Response Duration: 320.4608 ms]
[17:39:12.9197408]: [a] ACCEPT 3: Waiting for confirmation from g
[17:39:13.0599424]: [g] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 100 ms]
[17:39:13.3203168]: [g] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 100.144 ms]
[17:39:13.4905616]: [a] ACCEPT 3: Confirmation received from g [Total Duration: 430.6192 ms]
[17:39:13.6407776]: [a] ACCEPT 3: Updating direct experience
[17:39:13.7909936]: [a] ACCEPT 3: Experience with g was VeryGood
[17:39:13.9311952]: [a] ACCEPT 3: Previous experience:
[17:39:14.8825632]: [a] ACCEPT 3: Updated experience: a g Booking 1 0 0 0 VeryGood
[17:39:15.0227648]: [a] ACCEPT 3: End (Successful) [Total Duration: 3074.4208 ms]
[17:39:15.1829952]: [g] REQUEST 1: Accepted and serviced by a [Service Duration: 3384.8672 ms]
[17:39:15.3131824]: [g] REQUEST 1: Experience with a was VeryBad
[17:39:15.4533840]: [g] REQUEST 1: Previous experience: g a Booking 0 1 0 0 Good
[17:39:15.9741328]: [g] REQUEST 1: Updated experience: g a Booking 0 1 0 1 EquallyGoodAndBad
[17:39:16.2845792]: [g] REQUEST 1: Completed
[17:39:16.4247808]: [g] REQUEST 1: End (Successful) [Total Duration: 5818.3664 ms]
[17:39:31.0257760]: [c] REQUEST QUEUE: Begin
[17:39:31.5465248]: [c] REQUEST QUEUE: End

[17:39:31.7367984]: [c] REQUEST 1: Begin
[17:39:31.8770000]: [c] REQUEST 1: Searching for all known suitable agents.
[17:39:32.0172016]: [c] REQUEST 1: Found 2 known suitable agent(s).
[17:39:32.1774320]: [c] REQUEST 1: Known suitable agents: {a d}.
[17:39:32.3176336]: [a] ACCEPT 4: Begin
[17:39:32.6080512]: [a] ACCEPT 4: Servicing request from c
[17:39:32.7482528]: [a] ACCEPT 4: Start Service for c [Estimated Response Duration: 100 ms]
[17:39:33.0086272]: [a] ACCEPT 4: End Service for c [Actual Response Duration: 260.3744 ms]
[17:39:33.2890304]: [a] ACCEPT 4: Waiting for confirmation from c
[17:39:33.4292320]: [c] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 751 ms]
[17:39:34.3104992]: [c] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 751.08 ms]
[17:39:34.4707296]: [a] ACCEPT 4: Confirmation received from c [Total Duration: 1041.4976 ms]
[17:39:34.6309600]: [a] ACCEPT 4: Updating direct experience
[17:39:34.8012048]: [a] ACCEPT 4: Experience with c was VeryBad
[17:39:34.9013488]: [a] ACCEPT 4: Previous experience:
[17:39:35.6324000]: [a] ACCEPT 4: Updated experience: a c Booking 0 0 0 1 VeryBad
[17:39:35.7726016]: [a] ACCEPT 4: End (Successful) [Total Duration: 3294.7376 ms]
[17:39:35.9027888]: [c] REQUEST 1: Accepted and serviced by a [Service Duration: 3585.1552 ms]
[17:39:36.1531488]: [c] REQUEST 1: Experience with a was VeryBad
[17:39:36.2733216]: [c] REQUEST 1: Previous experience: c a Booking 1 0 0 0 VeryGood
[17:39:36.6538688]: [c] REQUEST 1: Updated experience: c a Booking 1 0 0 1 EquallyGoodAndBad
[17:39:37.0143872]: [c] REQUEST 1: Completed
[17:39:37.1846320]: [c] REQUEST 1: End (Successful) [Total Duration: 5447.8336 ms]
[17:39:42.0215872]: [e] REQUEST QUEUE: Begin
[17:39:42.4622208]: [e] REQUEST QUEUE: End
[17:39:42.5924080]: [e] REQUEST 1: Begin
[17:39:42.7626528]: [e] REQUEST 1: Searching for all known suitable agents.
[17:39:42.9328976]: [e] REQUEST 1: Found 1 known suitable agent(s).
[17:39:43.1131568]: [e] REQUEST 1: Known suitable agents: {a}.
[17:39:43.2533584]: [a] ACCEPT 5: Begin
[17:39:43.6639488]: [a] ACCEPT 5: Servicing request from e
[17:39:43.8041504]: [a] ACCEPT 5: Start Service for e [Estimated Response Duration: 100 ms]
[17:39:44.0244672]: [a] ACCEPT 5: End Service for e [Actual Response Duration: 220.3168 ms]
[17:39:44.3148848]: [a] ACCEPT 5: Waiting for confirmation from e
[17:39:44.4450720]: [e] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 100 ms]
[17:39:44.6754032]: [e] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 100.144 ms]
[17:39:44.8256192]: [a] ACCEPT 5: Confirmation received from e [Total Duration: 380.5472 ms]
[17:39:44.9758352]: [a] ACCEPT 5: Updating direct experience
[17:39:45.1160368]: [a] ACCEPT 5: Experience with e was VeryGood

[17:39:45.2562384]: [a] ACCEPT 5: Previous experience:
[17:39:46.0473760]: [a] ACCEPT 5: Updated experience: a e Booking 1 0 0 0 VeryGood
[17:39:46.1875776]: [a] ACCEPT 5: End (Successful) [Total Duration: 2763.9744 ms]
[17:39:46.3177648]: [e] REQUEST 1: Accepted and serviced by a [Service Duration: 3064.4064 ms]
[17:39:46.4679808]: [e] REQUEST 1: Experience with a was VeryBad
[17:39:46.6181968]: [e] REQUEST 1: Previous experience: e a Booking 1 0 0 0 VeryGood
[17:39:47.0988880]: [e] REQUEST 1: Updated experience: e a Booking 1 0 0 1 EquallyGoodAndBad
[17:39:47.3993200]: [e] REQUEST 1: Completed
[17:39:47.5695648]: [e] REQUEST 1: End (Successful) [Total Duration: 4977.1568 ms]
[17:39:54.0288528]: [d] REQUEST QUEUE: Begin
[17:39:54.4094000]: [d] REQUEST QUEUE: End
[17:39:54.5596160]: [d] REQUEST 1: Begin
[17:39:54.8500336]: [d] REQUEST 1: Searching for all known suitable agents.
[17:39:55.0102640]: [d] REQUEST 1: Found 1 known suitable agent(s).
[17:39:55.1504656]: [d] REQUEST 1: Known suitable agents: {a}.
[17:39:55.3307248]: [a] ACCEPT 6: Begin
[17:39:55.6612000]: [a] ACCEPT 6: Servicing request from d
[17:39:55.7813728]: [a] ACCEPT 6: Start Service for d [Estimated Response Duration: 100 ms]
[17:39:56.0016896]: [a] ACCEPT 6: End Service for d [Actual Response Duration: 230.3312 ms]
[17:39:56.2420352]: [a] ACCEPT 6: Waiting for confirmation from d
[17:39:56.4523376]: [d] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 803 ms]
[17:39:57.4037056]: [d] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 811.1664 ms]
[17:39:57.5739504]: [a] ACCEPT 6: Confirmation received from d [Total Duration: 1121.6128 ms]
[17:39:57.7141520]: [a] ACCEPT 6: Updating direct experience
[17:39:57.8843968]: [a] ACCEPT 6: Experience with d was VeryBad
[17:39:58.0746704]: [a] ACCEPT 6: Previous experience: a d Booking 0 0 1 0 Bad
[17:39:58.5153040]: [a] ACCEPT 6: Updated experience: a d Booking 0 0 1 1 MostlyBad
[17:39:58.6555056]: [a] ACCEPT 6: End (Successful) [Total Duration: 3104.464 ms]
[17:39:58.8557936]: [d] REQUEST 1: Accepted and serviced by a [Service Duration: 3525.0688 ms]
[17:39:58.9859808]: [d] REQUEST 1: Experience with a was VeryBad
[17:39:59.1462112]: [d] REQUEST 1: Previous experience: d a Booking 1 0 0 0 VeryGood
[17:39:59.6669600]: [d] REQUEST 1: Updated experience: d a Booking 1 0 0 1 EquallyGoodAndBad
[17:39:59.9573776]: [d] REQUEST 1: Completed
[17:40:00.1176080]: [d] REQUEST 1: End (Successful) [Total Duration: 5557.992 ms]
Completed simulation at: [17:40:00.1176080]

SimulationTrace.txt without prejudice filters

Trace of Agent 'a':

[17:38:39.7921056]: [a] ACCEPT 1: Begin
[17:38:40.3328832]: [a] ACCEPT 1: Servicing request from b
[17:38:40.5031280]: [a] ACCEPT 1: Start Service for b [Estimated Response Duration: 100 ms]
[17:38:40.7434736]: [a] ACCEPT 1: End Service for b [Actual Response Duration: 240.3456 ms]
[17:38:41.0439056]: [a] ACCEPT 1: Waiting for confirmation from b
[17:38:42.4959936]: [a] ACCEPT 1: Confirmation received from b [Total Duration: 1241.7856 ms]
[17:38:42.6361952]: [a] ACCEPT 1: Updating direct experience
[17:38:42.8264688]: [a] ACCEPT 1: Experience with b was VeryBad
[17:38:43.0367712]: [a] ACCEPT 1: Previous experience:
[17:38:43.8779808]: [a] ACCEPT 1: Updated experience: a b Booking 0 0 0 1 VeryBad
[17:38:44.0081680]: [a] ACCEPT 1: End (Successful) [Total Duration: 4055.832 ms]
[17:38:48.0239424]: [a] REQUEST QUEUE: Begin
[17:38:48.3944752]: [a] REQUEST QUEUE: End
[17:38:48.5146480]: [a] REQUEST 1: Begin
[17:38:48.6848928]: [a] REQUEST 1: Searching for all known suitable agents.
[17:38:48.8651520]: [a] REQUEST 1: Found no known suitable agents.
[17:38:49.0053536]: [a] REQUEST 1: No known agents could service the request.
[17:38:49.1755984]: [a] REQUEST 1: Searching for all unknown suitable agents.
[17:38:49.3358288]: [a] REQUEST 1: Found 6 unknown suitable agent(s).
[17:38:49.4860448]: [a] REQUEST 1: Unknown suitable agents: {d b c g e f}.
[17:38:52.1198320]: [a] REQUEST 1: Start Confirmation for d [Estimated Confirmation Duration: 100 ms]
[17:38:52.3802064]: [a] REQUEST 1: End Confirmation for d [Actual Confirmation Duration: 100.144 ms]
[17:38:54.1928128]: [a] REQUEST 1: Accepted and serviced by d [Service Duration: 4476.4368 ms]
[17:38:54.3430288]: [a] REQUEST 1: Experience with d was Bad
[17:38:54.4932448]: [a] REQUEST 1: Previous experience:
[17:38:55.3644976]: [a] REQUEST 1: Updated experience: a d Booking 0 0 1 0 Bad
[17:38:55.6348864]: [a] REQUEST 1: Completed
[17:38:55.7650736]: [a] REQUEST 1: End (Successful) [Total Duration: 7250.4256 ms]
[17:39:00.3917264]: [a] ACCEPT 2: Begin
[17:39:00.7121872]: [a] ACCEPT 2: Servicing request from f
[17:39:00.9024608]: [a] ACCEPT 2: Start Service for f [Estimated Response Duration: 100 ms]
[17:39:01.2028928]: [a] ACCEPT 2: End Service for f [Actual Response Duration: 300.432 ms]
[17:39:01.5533968]: [a] ACCEPT 2: Waiting for confirmation from f
[17:39:02.1442464]: [a] ACCEPT 2: Confirmation received from f [Total Duration: 460.6624 ms]
[17:39:02.2844480]: [a] ACCEPT 2: Updating direct experience
[17:39:02.4546928]: [a] ACCEPT 2: Experience with f was VeryGood
[17:39:02.6049088]: [a] ACCEPT 2: Previous experience:

[17:39:03.3960464]: [a] ACCEPT 2: Updated experience: a f Booking 1 0 0 0 VeryGood
[17:39:03.5462624]: [a] ACCEPT 2: End (Successful) [Total Duration: 2984.2912 ms]
[17:39:11.7981280]: [a] ACCEPT 3: Begin
[17:39:12.1286032]: [a] ACCEPT 3: Servicing request from g
[17:39:12.2788192]: [a] ACCEPT 3: Start Service for g [Estimated Response Duration: 100 ms]
[17:39:12.5992800]: [a] ACCEPT 3: End Service for g [Actual Response Duration: 320.4608 ms]
[17:39:12.9197408]: [a] ACCEPT 3: Waiting for confirmation from g
[17:39:13.4905616]: [a] ACCEPT 3: Confirmation received from g [Total Duration: 430.6192 ms]
[17:39:13.6407776]: [a] ACCEPT 3: Updating direct experience
[17:39:13.7909936]: [a] ACCEPT 3: Experience with g was VeryGood
[17:39:13.9311952]: [a] ACCEPT 3: Previous experience:
[17:39:14.8825632]: [a] ACCEPT 3: Updated experience: a g Booking 1 0 0 0 VeryGood
[17:39:15.0227648]: [a] ACCEPT 3: End (Successful) [Total Duration: 3074.4208 ms]
[17:39:32.3176336]: [a] ACCEPT 4: Begin
[17:39:32.6080512]: [a] ACCEPT 4: Servicing request from c
[17:39:32.7482528]: [a] ACCEPT 4: Start Service for c [Estimated Response Duration: 100 ms]
[17:39:33.0086272]: [a] ACCEPT 4: End Service for c [Actual Response Duration: 260.3744 ms]
[17:39:33.2890304]: [a] ACCEPT 4: Waiting for confirmation from c
[17:39:34.4707296]: [a] ACCEPT 4: Confirmation received from c [Total Duration: 1041.4976 ms]
[17:39:34.6309600]: [a] ACCEPT 4: Updating direct experience
[17:39:34.8012048]: [a] ACCEPT 4: Experience with c was VeryBad
[17:39:34.9013488]: [a] ACCEPT 4: Previous experience:
[17:39:35.6324000]: [a] ACCEPT 4: Updated experience: a c Booking 0 0 0 1 VeryBad
[17:39:35.7726016]: [a] ACCEPT 4: End (Successful) [Total Duration: 3294.7376 ms]
[17:39:43.2533584]: [a] ACCEPT 5: Begin
[17:39:43.6639488]: [a] ACCEPT 5: Servicing request from e
[17:39:43.8041504]: [a] ACCEPT 5: Start Service for e [Estimated Response Duration: 100 ms]
[17:39:44.0244672]: [a] ACCEPT 5: End Service for e [Actual Response Duration: 220.3168 ms]
[17:39:44.3148848]: [a] ACCEPT 5: Waiting for confirmation from e
[17:39:44.8256192]: [a] ACCEPT 5: Confirmation received from e [Total Duration: 380.5472 ms]
[17:39:44.9758352]: [a] ACCEPT 5: Updating direct experience
[17:39:45.1160368]: [a] ACCEPT 5: Experience with e was VeryGood
[17:39:45.2562384]: [a] ACCEPT 5: Previous experience:
[17:39:46.0473760]: [a] ACCEPT 5: Updated experience: a e Booking 1 0 0 0 VeryGood
[17:39:46.1875776]: [a] ACCEPT 5: End (Successful) [Total Duration: 2763.9744 ms]
[17:39:55.3307248]: [a] ACCEPT 6: Begin
[17:39:55.6612000]: [a] ACCEPT 6: Servicing request from d
[17:39:55.7813728]: [a] ACCEPT 6: Start Service for d [Estimated Response Duration: 100 ms]
[17:39:56.0016896]: [a] ACCEPT 6: End Service for d [Actual Response Duration: 230.3312 ms]
[17:39:56.2420352]: [a] ACCEPT 6: Waiting for confirmation from d
[17:39:57.5739504]: [a] ACCEPT 6: Confirmation received from d [Total Duration: 1121.6128 ms]
[17:39:57.7141520]: [a] ACCEPT 6: Updating direct experience
[17:39:57.8843968]: [a] ACCEPT 6: Experience with d was VeryBad

[17:39:58.0746704]: [a] ACCEPT 6: Previous experience: a d Booking 0 0 1 0 Bad
[17:39:58.5153040]: [a] ACCEPT 6: Updated experience: a d Booking 0 0 1 1 MostlyBad
[17:39:58.6555056]: [a] ACCEPT 6: End (Successful) [Total Duration: 3104.464 ms]

Trace of Agent 'f':

[17:38:59.0297680]: [f] REQUEST QUEUE: Begin
[17:38:59.5204736]: [f] REQUEST QUEUE: End
[17:38:59.6907184]: [f] REQUEST 1: Begin
[17:38:59.8509488]: [f] REQUEST 1: Searching for all known suitable agents.
[17:39:00.0011648]: [f] REQUEST 1: Found 1 known suitable agent(s).
[17:39:00.1714096]: [f] REQUEST 1: Known suitable agents: {a}.
[17:39:01.6835840]: [f] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 100 ms]
[17:39:01.9339440]: [f] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 100.144 ms]
[17:39:03.6664352]: [f] REQUEST 1: Accepted and serviced by a [Service Duration: 3274.7088 ms]
[17:39:03.8867520]: [f] REQUEST 1: Experience with a was VeryBad
[17:39:04.0269536]: [f] REQUEST 1: Previous experience: f a Booking 1 0 0 0 VeryGood
[17:39:04.4575728]: [f] REQUEST 1: Updated experience: f a Booking 1 0 0 1 EquallyGoodAndBad
[17:39:04.7379760]: [f] REQUEST 1: Completed
[17:39:04.9783216]: [f] REQUEST 1: End (Successful) [Total Duration: 5287.6032 ms]

Trace of Agent 'g':

[17:39:10.0255792]: [g] REQUEST QUEUE: Begin
[17:39:10.4662128]: [g] REQUEST QUEUE: End
[17:39:10.6064144]: [g] REQUEST 1: Begin
[17:39:10.7866736]: [g] REQUEST 1: Searching for all known suitable agents.
[17:39:10.9869616]: [g] REQUEST 1: Found no known suitable agents.
[17:39:11.1572064]: [g] REQUEST 1: No known agents could service the request.
[17:39:11.3174368]: [g] REQUEST 1: Searching for all unknown suitable agents.
[17:39:11.5077104]: [g] REQUEST 1: Found 3 unknown suitable agent(s).
[17:39:11.6579264]: [g] REQUEST 1: Unknown suitable agents: {a d f}.
[17:39:13.0599424]: [g] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 100 ms]
[17:39:13.3203168]: [g] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 100.144 ms]
[17:39:15.1829952]: [g] REQUEST 1: Accepted and serviced by a [Service Duration: 3384.8672 ms]
[17:39:15.3131824]: [g] REQUEST 1: Experience with a was VeryBad
[17:39:15.4533840]: [g] REQUEST 1: Previous experience: g a Booking 0 1 0 0 Good
[17:39:15.9741328]: [g] REQUEST 1: Updated experience: g a Booking 0 1 0 1 EquallyGoodAndBad
[17:39:16.2845792]: [g] REQUEST 1: Completed

[17:39:16.4247808]: [g] REQUEST 1: End (Successful) [Total Duration: 5818.3664 ms]

Trace of Agent 'd':

[17:38:49.7163760]: [d] ACCEPT 1: Begin
[17:38:50.2170960]: [d] ACCEPT 1: Servicing request from a
[17:38:50.4173840]: [d] ACCEPT 1: Start Service for a [Estimated Response Duration: 1000 ms]
[17:38:51.6391408]: [d] ACCEPT 1: End Service for a [Actual Response Duration: 1221.7568 ms]
[17:38:51.9395728]: [d] ACCEPT 1: Waiting for confirmation from a
[17:38:52.5404368]: [d] ACCEPT 1: Confirmation received from a [Total Duration: 420.6048 ms]
[17:38:52.7807824]: [d] ACCEPT 1: Updating direct experience
[17:38:52.9309984]: [d] ACCEPT 1: Experience with a was VeryGood
[17:38:53.1112576]: [d] ACCEPT 1: Previous experience:
[17:38:53.8923808]: [d] ACCEPT 1: Updated experience: d a Booking 1 0 0 0 VeryGood
[17:38:54.0425968]: [d] ACCEPT 1: End (Successful) [Total Duration: 4165.9904 ms]
[17:39:54.0288528]: [d] REQUEST QUEUE: Begin
[17:39:54.4094000]: [d] REQUEST QUEUE: End
[17:39:54.5596160]: [d] REQUEST 1: Begin
[17:39:54.8500336]: [d] REQUEST 1: Searching for all known suitable agents.
[17:39:55.0102640]: [d] REQUEST 1: Found 1 known suitable agent(s).
[17:39:55.1504656]: [d] REQUEST 1: Known suitable agents: {a}.
[17:39:56.4523376]: [d] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 803 ms]
[17:39:57.4037056]: [d] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 811.1664 ms]
[17:39:58.8557936]: [d] REQUEST 1: Accepted and serviced by a [Service Duration: 3525.0688 ms]
[17:39:58.9859808]: [d] REQUEST 1: Experience with a was VeryBad
[17:39:59.1462112]: [d] REQUEST 1: Previous experience: d a Booking 1 0 0 0 VeryGood
[17:39:59.6669600]: [d] REQUEST 1: Updated experience: d a Booking 1 0 0 1 EquallyGoodAndBad
[17:39:59.9573776]: [d] REQUEST 1: Completed
[17:40:00.1176080]: [d] REQUEST 1: End (Successful) [Total Duration: 5557.992 ms]

Trace of Agent 'e':

[17:39:42.0215872]: [e] REQUEST QUEUE: Begin
[17:39:42.4622208]: [e] REQUEST QUEUE: End
[17:39:42.5924080]: [e] REQUEST 1: Begin
[17:39:42.7626528]: [e] REQUEST 1: Searching for all known suitable agents.
[17:39:42.9328976]: [e] REQUEST 1: Found 1 known suitable agent(s).
[17:39:43.1131568]: [e] REQUEST 1: Known suitable agents: {a}.
[17:39:44.4450720]: [e] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 100 ms]

[17:39:44.6754032]: [e] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 100.144 ms]
[17:39:46.3177648]: [e] REQUEST 1: Accepted and serviced by a [Service Duration: 3064.4064 ms]
[17:39:46.4679808]: [e] REQUEST 1: Experience with a was VeryBad
[17:39:46.6181968]: [e] REQUEST 1: Previous experience: e a Booking 1 0 0 0 VeryGood
[17:39:47.0988880]: [e] REQUEST 1: Updated experience: e a Booking 1 0 0 1 EquallyGoodAndBad
[17:39:47.3993200]: [e] REQUEST 1: Completed
[17:39:47.5695648]: [e] REQUEST 1: End (Successful) [Total Duration: 4977.1568 ms]

Trace of Agent 'b':

[17:38:38.1197008]: [b] REQUEST QUEUE: Begin
[17:38:38.4902336]: [b] REQUEST QUEUE: End
[17:38:38.7205648]: [b] REQUEST 1: Begin
[17:38:38.9208528]: [b] REQUEST 1: Searching for all known suitable agents.
[17:38:39.0610544]: [b] REQUEST 1: Found no known suitable agents.
[17:38:39.2012560]: [b] REQUEST 1: No known agents could service the request.
[17:38:39.3014000]: [b] REQUEST 1: Searching for all unknown suitable agents.
[17:38:39.4716448]: [b] REQUEST 1: Found 6 unknown suitable agent(s).
[17:38:39.6218608]: [b] REQUEST 1: Unknown suitable agents: {a d c g e f}.
[17:38:41.2542080]: [b] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 902 ms]
[17:38:42.3057200]: [b] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 911.3104 ms]
[17:38:44.1283408]: [b] REQUEST 1: Accepted and serviced by a [Service Duration: 4336.2352 ms]
[17:38:44.2384992]: [b] REQUEST 1: Experience with a was VeryBad
[17:38:44.3486576]: [b] REQUEST 1: Previous experience:
[17:38:45.1998816]: [b] REQUEST 1: Updated experience: b a Booking 0 0 0 1 VeryBad
[17:38:45.4902992]: [b] REQUEST 1: Completed
[17:38:45.6805728]: [b] REQUEST 1: End (Successful) [Total Duration: 6960.008 ms]

Trace of Agent 'c':

[17:39:31.0257760]: [c] REQUEST QUEUE: Begin
[17:39:31.5465248]: [c] REQUEST QUEUE: End
[17:39:31.7367984]: [c] REQUEST 1: Begin
[17:39:31.8770000]: [c] REQUEST 1: Searching for all known suitable agents.
[17:39:32.0172016]: [c] REQUEST 1: Found 2 known suitable agent(s).
[17:39:32.1774320]: [c] REQUEST 1: Known suitable agents: {a d}.
[17:39:33.4292320]: [c] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 751 ms]
[17:39:34.3104992]: [c] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 751.08 ms]

[17:39:35.9027888]: [c] REQUEST 1: Accepted and serviced by a [Service Duration: 3585.1552 ms]
[17:39:36.1531488]: [c] REQUEST 1: Experience with a was VeryBad
[17:39:36.2733216]: [c] REQUEST 1: Previous experience: c a Booking 1 0 0 0 VeryGood
[17:39:36.6538688]: [c] REQUEST 1: Updated experience: c a Booking 1 0 0 1 EquallyGoodAndBad
[17:39:37.0143872]: [c] REQUEST 1: Completed
[17:39:37.1846320]: [c] REQUEST 1: End (Successful) [Total Duration: 5447.8336 ms]

Appendix B

Output with prejudice filters

SimulationLog.txt with prejudice filters

Starting simulation at: [17:36:27.8924432]

[17:36:28.0326448] ITERATION 1

All agents have been pooled and are waiting to request...
[17:36:28.2830048]: [b] REQUEST QUEUE: Begin
[17:36:28.7837248]: [b] REQUEST QUEUE: End
[17:36:28.9239264]: [b] REQUEST 1: Begin
[17:36:29.0541136]: [b] REQUEST 1: Searching for all known suitable agents.
[17:36:29.1943152]: [b] REQUEST 1: Found no known suitable agents.
[17:36:29.3745744]: [b] REQUEST 1: No known agents could service the request.
[17:36:29.5448192]: [b] REQUEST 1: Searching for all unknown suitable agents.
[17:36:29.7350928]: [b] REQUEST 1: Found 6 unknown suitable agent(s).
[17:36:29.8652800]: [b] REQUEST 1: Unknown suitable agents: {a d c g e f}.
[17:36:29.9854528]: [b] REQUEST 1: Running prejudice filter on unknown suitable agents.
[17:36:30.1056256]: [b] REQUEST 1: Unknown suitable agents (after filtering): {a g e f}
[17:36:30.2458272]: [a] ACCEPT 1: Begin
[17:36:30.7165040]: [a] ACCEPT 1: Running prejudice filter on b
[17:36:30.8466912]: [a] ACCEPT 1: Request received from an untrusted domain.
[17:36:30.9468352]: [a] ACCEPT 1: Denying request from b
[17:36:31.2873248]: [a] ACCEPT 1: End (Denied) [Total Duration: 881.2672 ms]
[17:36:31.4475552]: [b] REQUEST 1: Denied by a [Interaction Duration: 1211.7424 ms]
[17:36:31.7479872]: [g] ACCEPT 1: Begin
[17:36:32.2787504]: [g] ACCEPT 1: Running prejudice filter on b
[17:36:32.4289664]: [g] ACCEPT 1: Request received from a trusted domain.
[17:36:32.5591536]: [g] ACCEPT 1: Servicing request from b
[17:36:32.7293984]: [g] ACCEPT 1: Start Service for b [Estimated Response Duration: 100 ms]
[17:36:32.9597296]: [g] ACCEPT 1: End Service for b [Actual Response Duration: 230.3312 ms]
[17:36:33.2401328]: [g] ACCEPT 1: Waiting for confirmation from b
[17:36:33.4604496]: [b] REQUEST 1: Start Confirmation for g [Estimated Confirmation Duration: 710 ms]
[17:36:34.3517312]: [b] REQUEST 1: End Confirmation for g [Actual Confirmation Duration: 711.0224 ms]
[17:36:34.5119616]: [g] ACCEPT 1: Confirmation received from b [Total Duration: 1051.512 ms]
[17:36:34.6621776]: [g] ACCEPT 1: Updating direct experience
[17:36:34.8925088]: [g] ACCEPT 1: Experience with b was VeryBad
[17:36:35.0427248]: [g] ACCEPT 1: Previous experience:
[17:36:35.8839344]: [g] ACCEPT 1: Updated experience: g b Booking 0 0 0 1 VeryBad
[17:36:36.0942368]: [g] ACCEPT 1: End (Successful) [Total Duration: 4186.0192 ms]

[17:36:36.2144096]: [b] REQUEST 1: Accepted and serviced by g [Service Duration: 4466.4224 ms]
[17:36:36.3345824]: [b] REQUEST 1: Experience with g was VeryBad
[17:36:36.4747840]: [b] REQUEST 1: Previous experience:
[17:36:37.2859504]: [b] REQUEST 1: Updated experience: b g Booking 0 0 0 1 VeryBad
[17:36:37.6064112]: [b] REQUEST 1: Completed
[17:36:37.7566272]: [b] REQUEST 1: End (Successful) [Total Duration: 8832.7008 ms]
[17:36:38.1872464]: [a] REQUEST QUEUE: Begin
[17:36:38.4776640]: [a] REQUEST QUEUE: End
[17:36:38.6679376]: [a] REQUEST 1: Begin
[17:36:38.9082832]: [a] REQUEST 1: Searching for all known suitable agents.
[17:36:39.0484848]: [a] REQUEST 1: Found no known suitable agents.
[17:36:39.1987008]: [a] REQUEST 1: No known agents could service the request.
[17:36:39.4190176]: [a] REQUEST 1: Searching for all unknown suitable agents.
[17:36:39.5492048]: [a] REQUEST 1: Found 6 unknown suitable agent(s).
[17:36:39.7194496]: [a] REQUEST 1: Unknown suitable agents: {d b c g e f}.
[17:36:39.8696656]: [a] REQUEST 1: Running prejudice filter on unknown suitable agents.
[17:36:40.0499248]: [a] REQUEST 1: Unknown suitable agents (after filtering): {g e f}
[17:36:40.2502128]: [g] ACCEPT 2: Begin
[17:36:40.6107312]: [g] ACCEPT 2: Running prejudice filter on a
[17:36:40.7509328]: [g] ACCEPT 2: Request received from a trusted domain.
[17:36:40.9111632]: [g] ACCEPT 2: Servicing request from a
[17:36:41.0613792]: [g] ACCEPT 2: Start Service for a [Estimated Response Duration: 100 ms]
[17:36:41.3217536]: [g] ACCEPT 2: End Service for a [Actual Response Duration: 260.3744 ms]
[17:36:41.6522288]: [g] ACCEPT 2: Waiting for confirmation from a
[17:36:41.7924304]: [a] REQUEST 1: Start Confirmation for g [Estimated Confirmation Duration: 100 ms]
[17:36:42.0327760]: [a] REQUEST 1: End Confirmation for g [Actual Confirmation Duration: 100.144 ms]
[17:36:42.1729776]: [g] ACCEPT 2: Confirmation received from a [Total Duration: 380.5472 ms]
[17:36:42.3031648]: [g] ACCEPT 2: Updating direct experience
[17:36:42.5034528]: [g] ACCEPT 2: Experience with a was VeryGood
[17:36:42.6937264]: [g] ACCEPT 2: Previous experience: g a Booking 0 1 0 0 Good
[17:36:43.0943024]: [g] ACCEPT 2: Updated experience: g a Booking 1 1 0 0 MostlyGood
[17:36:43.2144752]: [g] ACCEPT 2: End (Successful) [Total Duration: 2784.0032 ms]
[17:36:43.3446624]: [a] REQUEST 1: Accepted and serviced by g [Service Duration: 3094.4496 ms]
[17:36:43.5449504]: [a] REQUEST 1: Experience with g was VeryGood
[17:36:43.6851520]: [a] REQUEST 1: Previous experience:
[17:36:44.6164912]: [a] REQUEST 1: Updated experience: a g Booking 1 0 0 0 VeryGood
[17:36:44.9269376]: [a] REQUEST 1: Completed
[17:36:45.0771536]: [a] REQUEST 1: End (Successful) [Total Duration: 6409.216 ms]
[17:36:49.1830576]: [f] REQUEST QUEUE: Begin
[17:36:49.6937920]: [f] REQUEST QUEUE: End
[17:36:49.8239792]: [f] REQUEST 1: Begin
[17:36:49.9541664]: [f] REQUEST 1: Searching for all known suitable agents.

[17:36:50.0843536]: [f] REQUEST 1: Found 1 known suitable agent(s).
[17:36:50.2846416]: [f] REQUEST 1: Known suitable agents: {a}.
[17:36:50.4248432]: [a] ACCEPT 2: Begin
[17:36:50.7753472]: [a] ACCEPT 2: Running prejudice filter on f
[17:36:50.9255632]: [a] ACCEPT 2: Request received from a trusted domain.
[17:36:51.0958080]: [a] ACCEPT 2: Servicing request from f
[17:36:51.2059664]: [a] ACCEPT 2: Start Service for f [Estimated Response Duration: 100 ms]
[17:36:51.4663408]: [a] ACCEPT 2: End Service for f [Actual Response Duration: 260.3744 ms]
[17:36:51.7367296]: [a] ACCEPT 2: Waiting for confirmation from f
[17:36:51.8869456]: [f] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 100 ms]
[17:36:52.1172768]: [f] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 100.144 ms]
[17:36:52.2374496]: [a] ACCEPT 2: Confirmation received from f [Total Duration: 350.504 ms]
[17:36:52.3676368]: [a] ACCEPT 2: Updating direct experience
[17:36:52.4978240]: [a] ACCEPT 2: Experience with f was VeryGood
[17:36:52.6480400]: [a] ACCEPT 2: Previous experience:
[17:36:53.4892496]: [a] ACCEPT 2: Updated experience: a f Booking 1 0 0 0 VeryGood
[17:36:53.6294512]: [a] ACCEPT 2: End (Successful) [Total Duration: 3034.3632 ms]
[17:36:53.8097104]: [f] REQUEST 1: Accepted and serviced by a [Service Duration: 3384.8672 ms]
[17:36:53.9599264]: [f] REQUEST 1: Experience with a was VeryBad
[17:36:54.1101424]: [f] REQUEST 1: Previous experience: f a Booking 1 0 0 0 VeryGood
[17:36:54.5808192]: [f] REQUEST 1: Updated experience: f a Booking 1 0 0 1 EquallyGoodAndBad
[17:36:54.8612224]: [f] REQUEST 1: Completed
[17:36:55.0014240]: [f] REQUEST 1: End (Successful) [Total Duration: 5177.4448 ms]
[17:37:00.1888832]: [g] REQUEST QUEUE: Begin
[17:37:00.4692864]: [g] REQUEST QUEUE: End
[17:37:00.6996176]: [g] REQUEST 1: Begin
[17:37:00.8698624]: [g] REQUEST 1: Searching for all known suitable agents.
[17:37:01.0200784]: [g] REQUEST 1: Found no known suitable agents.
[17:37:01.1402512]: [g] REQUEST 1: No known agents could service the request.
[17:37:01.3405392]: [g] REQUEST 1: Searching for all unknown suitable agents.
[17:37:01.4907552]: [g] REQUEST 1: Found 3 unknown suitable agent(s).
[17:37:01.6209424]: [g] REQUEST 1: Unknown suitable agents: {a d f}.
[17:37:01.7511296]: [g] REQUEST 1: Running prejudice filter on unknown suitable agents.
[17:37:01.9514176]: [g] REQUEST 1: Unknown suitable agents (after filtering): {a d f}
[17:37:02.0816048]: [a] ACCEPT 3: Begin
[17:37:02.3419792]: [a] ACCEPT 3: Running prejudice filter on g
[17:37:02.4921952]: [a] ACCEPT 3: Request received from a trusted domain.
[17:37:02.6123680]: [a] ACCEPT 3: Servicing request from g
[17:37:02.7325408]: [a] ACCEPT 3: Start Service for g [Estimated Response Duration: 100 ms]
[17:37:03.0329728]: [a] ACCEPT 3: End Service for g [Actual Response Duration: 300.432 ms]
[17:37:03.3233904]: [a] ACCEPT 3: Waiting for confirmation from g
[17:37:03.5236784]: [g] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 100 ms]

[17:37:03.7540096]: [g] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 100.144 ms]
[17:37:03.8942112]: [a] ACCEPT 3: Confirmation received from g [Total Duration: 370.5328 ms]
[17:37:04.1145280]: [a] ACCEPT 3: Updating direct experience
[17:37:04.2447152]: [a] ACCEPT 3: Experience with g was VeryGood
[17:37:04.4049456]: [a] ACCEPT 3: Previous experience: a g Booking 1 0 0 0 VeryGood
[17:37:04.8756224]: [a] ACCEPT 3: Updated experience: a g Booking 2 0 0 0 VeryGood
[17:37:05.0759104]: [a] ACCEPT 3: End (Successful) [Total Duration: 2854.104 ms]
[17:37:05.2361408]: [g] REQUEST 1: Accepted and serviced by a [Service Duration: 3154.536 ms]
[17:37:05.4063856]: [g] REQUEST 1: Experience with a was VeryBad
[17:37:05.6968032]: [g] REQUEST 1: Previous experience: g a Booking 1 1 0 0 MostlyGood
[17:37:06.1474512]: [g] REQUEST 1: Updated experience: g a Booking 1 1 0 1 MostlyGood
[17:37:06.4478832]: [g] REQUEST 1: Completed
[17:37:06.5680560]: [g] REQUEST 1: End (Successful) [Total Duration: 5868.4384 ms]
[17:37:21.1890800]: [c] REQUEST QUEUE: Begin
[17:37:21.6697712]: [c] REQUEST QUEUE: End
[17:37:21.7999584]: [c] REQUEST 1: Begin
[17:37:21.9201312]: [c] REQUEST 1: Searching for all known suitable agents.
[17:37:22.0603328]: [c] REQUEST 1: Found 2 known suitable agent(s).
[17:37:22.2005344]: [c] REQUEST 1: Known suitable agents: {a d}.
[17:37:22.3407360]: [a] ACCEPT 4: Begin
[17:37:22.5810816]: [a] ACCEPT 4: Running prejudice filter on c
[17:37:22.7112688]: [a] ACCEPT 4: Request received from an untrusted domain.
[17:37:22.9115568]: [a] ACCEPT 4: Denying request from c
[17:37:23.1919600]: [a] ACCEPT 4: End (Denied) [Total Duration: 721.0368 ms]
[17:37:23.3722192]: [c] REQUEST 1: Denied by a [Interaction Duration: 1031.4832 ms]
[17:37:23.6826656]: [d] ACCEPT 1: Begin
[17:37:24.2334576]: [d] ACCEPT 1: Running prejudice filter on c
[17:37:24.3936880]: [d] ACCEPT 1: Request received from a trusted domain.
[17:37:24.5639328]: [d] ACCEPT 1: Servicing request from c
[17:37:24.7742352]: [d] ACCEPT 1: Start Service for c [Estimated Response Duration: 1000 ms]
[17:37:25.9359056]: [d] ACCEPT 1: End Service for c [Actual Response Duration: 1161.6704 ms]
[17:37:26.2964240]: [d] ACCEPT 1: Waiting for confirmation from c
[17:37:26.5467840]: [c] REQUEST 1: Start Confirmation for d [Estimated Confirmation Duration: 967 ms]
[17:37:27.6683968]: [c] REQUEST 1: End Confirmation for d [Actual Confirmation Duration: 971.3968 ms]
[17:37:27.8085984]: [d] ACCEPT 1: Confirmation received from c [Total Duration: 1261.8144 ms]
[17:37:27.9788432]: [d] ACCEPT 1: Updating direct experience
[17:37:28.1490880]: [d] ACCEPT 1: Experience with c was VeryBad
[17:37:28.3193328]: [d] ACCEPT 1: Previous experience:
[17:37:29.1605424]: [d] ACCEPT 1: Updated experience: d c Booking 0 0 0 1 VeryBad
[17:37:29.3007440]: [d] ACCEPT 1: End (Successful) [Total Duration: 5457.848 ms]

[17:37:29.4609744]: [c] REQUEST 1: Accepted and serviced by d [Service Duration: 5778.3088 ms]
[17:37:29.6011760]: [c] REQUEST 1: Experience with d was VeryBad
[17:37:29.7313632]: [c] REQUEST 1: Previous experience: c d Booking 1 0 0 0 VeryGood
[17:37:30.1619824]: [c] REQUEST 1: Updated experience: c d Booking 1 0 0 1 EquallyGoodAndBad
[17:37:30.4323712]: [c] REQUEST 1: Completed
[17:37:30.5725728]: [c] REQUEST 1: End (Successful) [Total Duration: 8772.6144 ms]
[17:37:32.1848912]: [e] REQUEST QUEUE: Begin
[17:37:32.7156544]: [e] REQUEST QUEUE: End
[17:37:32.9159424]: [e] REQUEST 1: Begin
[17:37:33.1162304]: [e] REQUEST 1: Searching for all known suitable agents.
[17:37:33.2764608]: [e] REQUEST 1: Found 1 known suitable agent(s).
[17:37:33.4266768]: [e] REQUEST 1: Known suitable agents: {a}.
[17:37:33.5869072]: [a] ACCEPT 5: Begin
[17:37:33.9374112]: [a] ACCEPT 5: Running prejudice filter on e
[17:37:34.1276848]: [a] ACCEPT 5: Request received from a trusted domain.
[17:37:34.2879152]: [a] ACCEPT 5: Servicing request from e
[17:37:34.4381312]: [a] ACCEPT 5: Start Service for e [Estimated Response Duration: 100 ms]
[17:37:34.7686064]: [a] ACCEPT 5: End Service for e [Actual Response Duration: 330.4752 ms]
[17:37:35.1291248]: [a] ACCEPT 5: Waiting for confirmation from e
[17:37:35.3193984]: [e] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 100 ms]
[17:37:35.5697584]: [e] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 100.144 ms]
[17:37:35.7299888]: [a] ACCEPT 5: Confirmation received from e [Total Duration: 410.5904 ms]
[17:37:35.8802048]: [a] ACCEPT 5: Updating direct experience
[17:37:36.0404352]: [a] ACCEPT 5: Experience with e was VeryGood
[17:37:36.1806368]: [a] ACCEPT 5: Previous experience:
[17:37:37.1520336]: [a] ACCEPT 5: Updated experience: a e Booking 1 0 0 0 VeryGood
[17:37:37.2922352]: [a] ACCEPT 5: End (Successful) [Total Duration: 3495.0256 ms]
[17:37:37.4925232]: [e] REQUEST 1: Accepted and serviced by a [Service Duration: 3905.616 ms]
[17:37:37.6327248]: [e] REQUEST 1: Experience with a was VeryBad
[17:37:37.8229984]: [e] REQUEST 1: Previous experience: e a Booking 1 0 0 0 VeryGood
[17:37:38.2636320]: [e] REQUEST 1: Updated experience: e a Booking 1 0 0 1 EquallyGoodAndBad
[17:37:38.6341648]: [e] REQUEST 1: Completed
[17:37:38.7943952]: [e] REQUEST 1: End (Successful) [Total Duration: 5878.4528 ms]
[17:37:44.1921568]: [d] REQUEST QUEUE: Begin
[17:37:44.5727040]: [d] REQUEST QUEUE: End
[17:37:44.7529632]: [d] REQUEST 1: Begin
[17:37:44.8931648]: [d] REQUEST 1: Searching for all known suitable agents.
[17:37:45.0734240]: [d] REQUEST 1: Found no known suitable agents.
[17:37:45.2236400]: [d] REQUEST 1: No known agents could service the request.
[17:37:45.4239280]: [d] REQUEST 1: Searching for all unknown suitable agents.

[17:37:45.6642736]: [d] REQUEST 1: Found 4 unknown suitable agent(s).
[17:37:45.8245040]: [d] REQUEST 1: Unknown suitable agents: {a g f e}.
[17:37:45.9647056]: [d] REQUEST 1: Running prejudice filter on unknown suitable agents.
[17:37:46.1449648]: [d] REQUEST 1: Unknown suitable agents (after filtering): {a g f e}
[17:37:46.2951808]: [a] ACCEPT 6: Begin
[17:37:46.6857424]: [a] ACCEPT 6: Running prejudice filter on d
[17:37:46.8559872]: [a] ACCEPT 6: Request received from an untrusted domain.
[17:37:47.0362464]: [a] ACCEPT 6: Denying request from d
[17:37:47.3767360]: [a] ACCEPT 6: End (Denied) [Total Duration: 891.2816 ms]
[17:37:47.5369664]: [d] REQUEST 1: Denied by a [Interaction Duration: 1241.7856 ms]
[17:37:47.8373984]: [g] ACCEPT 3: Begin
[17:37:48.0977728]: [g] ACCEPT 3: Running prejudice filter on d
[17:37:48.2379744]: [g] ACCEPT 3: Request received from a trusted domain.
[17:37:48.3781760]: [g] ACCEPT 3: Servicing request from d
[17:37:48.5083632]: [g] ACCEPT 3: Start Service for d [Estimated Response Duration: 100 ms]
[17:37:48.7487088]: [g] ACCEPT 3: End Service for d [Actual Response Duration: 240.3456 ms]
[17:37:49.0391264]: [g] ACCEPT 3: Waiting for confirmation from d
[17:37:49.2193856]: [d] REQUEST 1: Start Confirmation for g [Estimated Confirmation Duration: 752 ms]
[17:37:50.1006528]: [d] REQUEST 1: End Confirmation for g [Actual Confirmation Duration: 761.0944 ms]
[17:37:50.3109552]: [g] ACCEPT 3: Confirmation received from d [Total Duration: 1091.5696 ms]
[17:37:50.4812000]: [g] ACCEPT 3: Updating direct experience
[17:37:50.6514448]: [g] ACCEPT 3: Experience with d was VeryBad
[17:37:50.8317040]: [g] ACCEPT 3: Previous experience:
[17:37:51.8331440]: [g] ACCEPT 3: Updated experience: g d Booking 0 0 0 1 VeryBad
[17:37:52.0334320]: [g] ACCEPT 3: End (Successful) [Total Duration: 4055.832 ms]
[17:37:52.2337200]: [d] REQUEST 1: Accepted and serviced by g [Service Duration: 4396.3216 ms]
[17:37:52.3839360]: [d] REQUEST 1: Experience with g was VeryBad
[17:37:52.5742096]: [d] REQUEST 1: Previous experience:
[17:37:53.4254336]: [d] REQUEST 1: Updated experience: d g Booking 0 0 0 1 VeryBad
[17:37:53.7959664]: [d] REQUEST 1: Completed
[17:37:53.9461824]: [d] REQUEST 1: End (Successful) [Total Duration: 9193.2192 ms]
Completed simulation at: [17:37:53.9461824]

SimulationTrace.txt with prejudice filters

Trace of Agent 'a':

[17:36:30.2458272]: [a] ACCEPT 1: Begin
[17:36:30.7165040]: [a] ACCEPT 1: Running prejudice filter on b
[17:36:30.8466912]: [a] ACCEPT 1: Request received from an untrusted domain.
[17:36:30.9468352]: [a] ACCEPT 1: Denying request from b
[17:36:31.2873248]: [a] ACCEPT 1: End (Denied) [Total Duration: 881.2672 ms]
[17:36:38.1872464]: [a] REQUEST QUEUE: Begin
[17:36:38.4776640]: [a] REQUEST QUEUE: End
[17:36:38.6679376]: [a] REQUEST 1: Begin
[17:36:38.9082832]: [a] REQUEST 1: Searching for all known suitable agents.
[17:36:39.0484848]: [a] REQUEST 1: Found no known suitable agents.
[17:36:39.1987008]: [a] REQUEST 1: No known agents could service the request.
[17:36:39.4190176]: [a] REQUEST 1: Searching for all unknown suitable agents.
[17:36:39.5492048]: [a] REQUEST 1: Found 6 unknown suitable agent(s).
[17:36:39.7194496]: [a] REQUEST 1: Unknown suitable agents: {d b c g e f}.
[17:36:39.8696656]: [a] REQUEST 1: Running prejudice filter on unknown suitable agents.
[17:36:40.0499248]: [a] REQUEST 1: Unknown suitable agents (after filtering): {g e f}
[17:36:41.7924304]: [a] REQUEST 1: Start Confirmation for g [Estimated Confirmation Duration: 100 ms]
[17:36:42.0327760]: [a] REQUEST 1: End Confirmation for g [Actual Confirmation Duration: 100.144 ms]
[17:36:43.3446624]: [a] REQUEST 1: Accepted and serviced by g [Service Duration: 3094.4496 ms]
[17:36:43.5449504]: [a] REQUEST 1: Experience with g was VeryGood
[17:36:43.6851520]: [a] REQUEST 1: Previous experience:
[17:36:44.6164912]: [a] REQUEST 1: Updated experience: a g Booking 1 0 0 0 VeryGood
[17:36:44.9269376]: [a] REQUEST 1: Completed
[17:36:45.0771536]: [a] REQUEST 1: End (Successful) [Total Duration: 6409.216 ms]
[17:36:50.4248432]: [a] ACCEPT 2: Begin
[17:36:50.7753472]: [a] ACCEPT 2: Running prejudice filter on f
[17:36:50.9255632]: [a] ACCEPT 2: Request received from a trusted domain.
[17:36:51.0958080]: [a] ACCEPT 2: Servicing request from f
[17:36:51.2059664]: [a] ACCEPT 2: Start Service for f [Estimated Response Duration: 100 ms]
[17:36:51.4663408]: [a] ACCEPT 2: End Service for f [Actual Response Duration: 260.3744 ms]
[17:36:51.7367296]: [a] ACCEPT 2: Waiting for confirmation from f
[17:36:52.2374496]: [a] ACCEPT 2: Confirmation received from f [Total Duration: 350.504 ms]
[17:36:52.3676368]: [a] ACCEPT 2: Updating direct experience
[17:36:52.4978240]: [a] ACCEPT 2: Experience with f was VeryGood
[17:36:52.6480400]: [a] ACCEPT 2: Previous experience:
[17:36:53.4892496]: [a] ACCEPT 2: Updated experience: a f Booking 1 0 0 0 VeryGood
[17:36:53.6294512]: [a] ACCEPT 2: End (Successful) [Total Duration: 3034.3632 ms]
[17:37:02.0816048]: [a] ACCEPT 3: Begin
[17:37:02.3419792]: [a] ACCEPT 3: Running prejudice filter on g

[17:37:02.4921952]: [a] ACCEPT 3: Request received from a trusted domain.
[17:37:02.6123680]: [a] ACCEPT 3: Servicing request from g
[17:37:02.7325408]: [a] ACCEPT 3: Start Service for g [Estimated Response Duration: 100 ms]
[17:37:03.0329728]: [a] ACCEPT 3: End Service for g [Actual Response Duration: 300.432 ms]
[17:37:03.3233904]: [a] ACCEPT 3: Waiting for confirmation from g
[17:37:03.8942112]: [a] ACCEPT 3: Confirmation received from g [Total Duration: 370.5328 ms]
[17:37:04.1145280]: [a] ACCEPT 3: Updating direct experience
[17:37:04.2447152]: [a] ACCEPT 3: Experience with g was VeryGood
[17:37:04.4049456]: [a] ACCEPT 3: Previous experience: a g Booking 1 0 0 0 VeryGood
[17:37:04.8756224]: [a] ACCEPT 3: Updated experience: a g Booking 2 0 0 0 VeryGood
[17:37:05.0759104]: [a] ACCEPT 3: End (Successful) [Total Duration: 2854.104 ms]
[17:37:22.3407360]: [a] ACCEPT 4: Begin
[17:37:22.5810816]: [a] ACCEPT 4: Running prejudice filter on c
[17:37:22.7112688]: [a] ACCEPT 4: Request received from an untrusted domain.
[17:37:22.9115568]: [a] ACCEPT 4: Denying request from c
[17:37:23.1919600]: [a] ACCEPT 4: End (Denied) [Total Duration: 721.0368 ms]
[17:37:33.5869072]: [a] ACCEPT 5: Begin
[17:37:33.9374112]: [a] ACCEPT 5: Running prejudice filter on e
[17:37:34.1276848]: [a] ACCEPT 5: Request received from a trusted domain.
[17:37:34.2879152]: [a] ACCEPT 5: Servicing request from e
[17:37:34.4381312]: [a] ACCEPT 5: Start Service for e [Estimated Response Duration: 100 ms]
[17:37:34.7686064]: [a] ACCEPT 5: End Service for e [Actual Response Duration: 330.4752 ms]
[17:37:35.1291248]: [a] ACCEPT 5: Waiting for confirmation from e
[17:37:35.7299888]: [a] ACCEPT 5: Confirmation received from e [Total Duration: 410.5904 ms]
[17:37:35.8802048]: [a] ACCEPT 5: Updating direct experience
[17:37:36.0404352]: [a] ACCEPT 5: Experience with e was VeryGood
[17:37:36.1806368]: [a] ACCEPT 5: Previous experience:
[17:37:37.1520336]: [a] ACCEPT 5: Updated experience: a e Booking 1 0 0 0 VeryGood
[17:37:37.2922352]: [a] ACCEPT 5: End (Successful) [Total Duration: 3495.0256 ms]
[17:37:46.2951808]: [a] ACCEPT 6: Begin
[17:37:46.6857424]: [a] ACCEPT 6: Running prejudice filter on d
[17:37:46.8559872]: [a] ACCEPT 6: Request received from an untrusted domain.
[17:37:47.0362464]: [a] ACCEPT 6: Denying request from d
[17:37:47.3767360]: [a] ACCEPT 6: End (Denied) [Total Duration: 891.2816 ms]

Trace of Agent 'f':

[17:36:49.1830576]: [f] REQUEST QUEUE: Begin
[17:36:49.6937920]: [f] REQUEST QUEUE: End
[17:36:49.8239792]: [f] REQUEST 1: Begin
[17:36:49.9541664]: [f] REQUEST 1: Searching for all known suitable agents.
[17:36:50.0843536]: [f] REQUEST 1: Found 1 known suitable agent(s).
[17:36:50.2846416]: [f] REQUEST 1: Known suitable agents: {a}.
[17:36:51.8869456]: [f] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 100 ms]

[17:36:52.1172768]: [f] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 100.144 ms]
[17:36:53.8097104]: [f] REQUEST 1: Accepted and serviced by a [Service Duration: 3384.8672 ms]
[17:36:53.9599264]: [f] REQUEST 1: Experience with a was VeryBad
[17:36:54.1101424]: [f] REQUEST 1: Previous experience: f a Booking 1 0 0 0 VeryGood
[17:36:54.5808192]: [f] REQUEST 1: Updated experience: f a Booking 1 0 0 1 EquallyGoodAndBad
[17:36:54.8612224]: [f] REQUEST 1: Completed
[17:36:55.0014240]: [f] REQUEST 1: End (Successful) [Total Duration: 5177.4448 ms]

Trace of Agent 'g':

[17:36:31.7479872]: [g] ACCEPT 1: Begin
[17:36:32.2787504]: [g] ACCEPT 1: Running prejudice filter on b
[17:36:32.4289664]: [g] ACCEPT 1: Request received from a trusted domain.
[17:36:32.5591536]: [g] ACCEPT 1: Servicing request from b
[17:36:32.7293984]: [g] ACCEPT 1: Start Service for b [Estimated Response Duration: 100 ms]
[17:36:32.9597296]: [g] ACCEPT 1: End Service for b [Actual Response Duration: 230.3312 ms]
[17:36:33.2401328]: [g] ACCEPT 1: Waiting for confirmation from b
[17:36:34.5119616]: [g] ACCEPT 1: Confirmation received from b [Total Duration: 1051.512 ms]
[17:36:34.6621776]: [g] ACCEPT 1: Updating direct experience
[17:36:34.8925088]: [g] ACCEPT 1: Experience with b was VeryBad
[17:36:35.0427248]: [g] ACCEPT 1: Previous experience:
[17:36:35.8839344]: [g] ACCEPT 1: Updated experience: g b Booking 0 0 0 1 VeryBad
[17:36:36.0942368]: [g] ACCEPT 1: End (Successful) [Total Duration: 4186.0192 ms]
[17:36:40.2502128]: [g] ACCEPT 2: Begin
[17:36:40.6107312]: [g] ACCEPT 2: Running prejudice filter on a
[17:36:40.7509328]: [g] ACCEPT 2: Request received from a trusted domain.
[17:36:40.9111632]: [g] ACCEPT 2: Servicing request from a
[17:36:41.0613792]: [g] ACCEPT 2: Start Service for a [Estimated Response Duration: 100 ms]
[17:36:41.3217536]: [g] ACCEPT 2: End Service for a [Actual Response Duration: 260.3744 ms]
[17:36:41.6522288]: [g] ACCEPT 2: Waiting for confirmation from a
[17:36:42.1729776]: [g] ACCEPT 2: Confirmation received from a [Total Duration: 380.5472 ms]
[17:36:42.3031648]: [g] ACCEPT 2: Updating direct experience
[17:36:42.5034528]: [g] ACCEPT 2: Experience with a was VeryGood
[17:36:42.6937264]: [g] ACCEPT 2: Previous experience: g a Booking 0 1 0 0 Good
[17:36:43.0943024]: [g] ACCEPT 2: Updated experience: g a Booking 1 1 0 0 MostlyGood
[17:36:43.2144752]: [g] ACCEPT 2: End (Successful) [Total Duration: 2784.0032 ms]
[17:37:00.1888832]: [g] REQUEST QUEUE: Begin
[17:37:00.4692864]: [g] REQUEST QUEUE: End
[17:37:00.6996176]: [g] REQUEST 1: Begin
[17:37:00.8698624]: [g] REQUEST 1: Searching for all known suitable agents.
[17:37:01.0200784]: [g] REQUEST 1: Found no known suitable agents.

[17:37:01.1402512]: [g] REQUEST 1: No known agents could service the request.
[17:37:01.3405392]: [g] REQUEST 1: Searching for all unknown suitable agents.
[17:37:01.4907552]: [g] REQUEST 1: Found 3 unknown suitable agent(s).
[17:37:01.6209424]: [g] REQUEST 1: Unknown suitable agents: {a d f}.
[17:37:01.7511296]: [g] REQUEST 1: Running prejudice filter on unknown suitable agents.
[17:37:01.9514176]: [g] REQUEST 1: Unknown suitable agents (after filtering): {a d f}
[17:37:03.5236784]: [g] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 100 ms]
[17:37:03.7540096]: [g] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 100.144 ms]
[17:37:05.2361408]: [g] REQUEST 1: Accepted and serviced by a [Service Duration: 3154.536 ms]
[17:37:05.4063856]: [g] REQUEST 1: Experience with a was VeryBad
[17:37:05.6968032]: [g] REQUEST 1: Previous experience: g a Booking 1 1 0 0 MostlyGood
[17:37:06.1474512]: [g] REQUEST 1: Updated experience: g a Booking 1 1 0 1 MostlyGood
[17:37:06.4478832]: [g] REQUEST 1: Completed
[17:37:06.5680560]: [g] REQUEST 1: End (Successful) [Total Duration: 5868.4384 ms]
[17:37:47.8373984]: [g] ACCEPT 3: Begin
[17:37:48.0977728]: [g] ACCEPT 3: Running prejudice filter on d
[17:37:48.2379744]: [g] ACCEPT 3: Request received from a trusted domain.
[17:37:48.3781760]: [g] ACCEPT 3: Servicing request from d
[17:37:48.5083632]: [g] ACCEPT 3: Start Service for d [Estimated Response Duration: 100 ms]
[17:37:48.7487088]: [g] ACCEPT 3: End Service for d [Actual Response Duration: 240.3456 ms]
[17:37:49.0391264]: [g] ACCEPT 3: Waiting for confirmation from d
[17:37:50.3109552]: [g] ACCEPT 3: Confirmation received from d [Total Duration: 1091.5696 ms]
[17:37:50.4812000]: [g] ACCEPT 3: Updating direct experience
[17:37:50.6514448]: [g] ACCEPT 3: Experience with d was VeryBad
[17:37:50.8317040]: [g] ACCEPT 3: Previous experience:
[17:37:51.8331440]: [g] ACCEPT 3: Updated experience: g d Booking 0 0 0 1 VeryBad
[17:37:52.0334320]: [g] ACCEPT 3: End (Successful) [Total Duration: 4055.832 ms]

Trace of Agent 'd':

[17:37:23.6826656]: [d] ACCEPT 1: Begin
[17:37:24.2334576]: [d] ACCEPT 1: Running prejudice filter on c
[17:37:24.3936880]: [d] ACCEPT 1: Request received from a trusted domain.
[17:37:24.5639328]: [d] ACCEPT 1: Servicing request from c
[17:37:24.7742352]: [d] ACCEPT 1: Start Service for c [Estimated Response Duration: 1000 ms]
[17:37:25.9359056]: [d] ACCEPT 1: End Service for c [Actual Response Duration: 1161.6704 ms]
[17:37:26.2964240]: [d] ACCEPT 1: Waiting for confirmation from c
[17:37:27.8085984]: [d] ACCEPT 1: Confirmation received from c [Total Duration: 1261.8144 ms]
[17:37:27.9788432]: [d] ACCEPT 1: Updating direct experience
[17:37:28.1490880]: [d] ACCEPT 1: Experience with c was VeryBad

[17:37:28.3193328]: [d] ACCEPT 1: Previous experience:
[17:37:29.1605424]: [d] ACCEPT 1: Updated experience: d c Booking 0 0 0 1 VeryBad
[17:37:29.3007440]: [d] ACCEPT 1: End (Successful) [Total Duration: 5457.848 ms]
[17:37:44.1921568]: [d] REQUEST QUEUE: Begin
[17:37:44.5727040]: [d] REQUEST QUEUE: End
[17:37:44.7529632]: [d] REQUEST 1: Begin
[17:37:44.8931648]: [d] REQUEST 1: Searching for all known suitable agents.
[17:37:45.0734240]: [d] REQUEST 1: Found no known suitable agents.
[17:37:45.2236400]: [d] REQUEST 1: No known agents could service the request.
[17:37:45.4239280]: [d] REQUEST 1: Searching for all unknown suitable agents.
[17:37:45.6642736]: [d] REQUEST 1: Found 4 unknown suitable agent(s).
[17:37:45.8245040]: [d] REQUEST 1: Unknown suitable agents: {a g f e}.
[17:37:45.9647056]: [d] REQUEST 1: Running prejudice filter on unknown suitable agents.
[17:37:46.1449648]: [d] REQUEST 1: Unknown suitable agents (after filtering): {a g f e}
[17:37:47.5369664]: [d] REQUEST 1: Denied by a [Interaction Duration: 1241.7856 ms]
[17:37:49.2193856]: [d] REQUEST 1: Start Confirmation for g [Estimated Confirmation Duration: 752 ms]
[17:37:50.1006528]: [d] REQUEST 1: End Confirmation for g [Actual Confirmation Duration: 761.0944 ms]
[17:37:52.2337200]: [d] REQUEST 1: Accepted and serviced by g [Service Duration: 4396.3216 ms]
[17:37:52.3839360]: [d] REQUEST 1: Experience with g was VeryBad
[17:37:52.5742096]: [d] REQUEST 1: Previous experience:
[17:37:53.4254336]: [d] REQUEST 1: Updated experience: d g Booking 0 0 0 1 VeryBad
[17:37:53.7959664]: [d] REQUEST 1: Completed
[17:37:53.9461824]: [d] REQUEST 1: End (Successful) [Total Duration: 9193.2192 ms]

Trace of Agent 'e':

[17:37:32.1848912]: [e] REQUEST QUEUE: Begin
[17:37:32.7156544]: [e] REQUEST QUEUE: End
[17:37:32.9159424]: [e] REQUEST 1: Begin
[17:37:33.1162304]: [e] REQUEST 1: Searching for all known suitable agents.
[17:37:33.2764608]: [e] REQUEST 1: Found 1 known suitable agent(s).
[17:37:33.4266768]: [e] REQUEST 1: Known suitable agents: {a}.
[17:37:35.3193984]: [e] REQUEST 1: Start Confirmation for a [Estimated Confirmation Duration: 100 ms]
[17:37:35.5697584]: [e] REQUEST 1: End Confirmation for a [Actual Confirmation Duration: 100.144 ms]
[17:37:37.4925232]: [e] REQUEST 1: Accepted and serviced by a [Service Duration: 3905.616 ms]
[17:37:37.6327248]: [e] REQUEST 1: Experience with a was VeryBad
[17:37:37.8229984]: [e] REQUEST 1: Previous experience: e a Booking 1 0 0 0 VeryGood
[17:37:38.2636320]: [e] REQUEST 1: Updated experience: e a Booking 1 0 0 1 EquallyGoodAndBad
[17:37:38.6341648]: [e] REQUEST 1: Completed
[17:37:38.7943952]: [e] REQUEST 1: End (Successful) [Total Duration: 5878.4528 ms]

Trace of Agent 'b':

[17:36:28.2830048]: [b] REQUEST QUEUE: Begin
[17:36:28.7837248]: [b] REQUEST QUEUE: End
[17:36:28.9239264]: [b] REQUEST 1: Begin
[17:36:29.0541136]: [b] REQUEST 1: Searching for all known suitable agents.
[17:36:29.1943152]: [b] REQUEST 1: Found no known suitable agents.
[17:36:29.3745744]: [b] REQUEST 1: No known agents could service the request.
[17:36:29.5448192]: [b] REQUEST 1: Searching for all unknown suitable agents.
[17:36:29.7350928]: [b] REQUEST 1: Found 6 unknown suitable agent(s).
[17:36:29.8652800]: [b] REQUEST 1: Unknown suitable agents: {a d c g e f}.
[17:36:29.9854528]: [b] REQUEST 1: Running prejudice filter on unknown suitable agents.
[17:36:30.1056256]: [b] REQUEST 1: Unknown suitable agents (after filtering): {a g e f}
[17:36:31.4475552]: [b] REQUEST 1: Denied by a [Interaction Duration: 1211.7424 ms]
[17:36:33.4604496]: [b] REQUEST 1: Start Confirmation for g [Estimated Confirmation Duration: 710 ms]
[17:36:34.3517312]: [b] REQUEST 1: End Confirmation for g [Actual Confirmation Duration: 711.0224 ms]
[17:36:36.2144096]: [b] REQUEST 1: Accepted and serviced by g [Service Duration: 4466.4224 ms]
[17:36:36.3345824]: [b] REQUEST 1: Experience with g was VeryBad
[17:36:36.4747840]: [b] REQUEST 1: Previous experience:
[17:36:37.2859504]: [b] REQUEST 1: Updated experience: b g Booking 0 0 0 1 VeryBad
[17:36:37.6064112]: [b] REQUEST 1: Completed
[17:36:37.7566272]: [b] REQUEST 1: End (Successful) [Total Duration: 8832.7008 ms]

Trace of Agent 'c':

[17:37:21.1890800]: [c] REQUEST QUEUE: Begin
[17:37:21.6697712]: [c] REQUEST QUEUE: End
[17:37:21.7999584]: [c] REQUEST 1: Begin
[17:37:21.9201312]: [c] REQUEST 1: Searching for all known suitable agents.
[17:37:22.0603328]: [c] REQUEST 1: Found 2 known suitable agent(s).
[17:37:22.2005344]: [c] REQUEST 1: Known suitable agents: {a d}.
[17:37:23.3722192]: [c] REQUEST 1: Denied by a [Interaction Duration: 1031.4832 ms]
[17:37:26.5467840]: [c] REQUEST 1: Start Confirmation for d [Estimated Confirmation Duration: 967 ms]
[17:37:27.6683968]: [c] REQUEST 1: End Confirmation for d [Actual Confirmation Duration: 971.3968 ms]
[17:37:29.4609744]: [c] REQUEST 1: Accepted and serviced by d [Service Duration: 5778.3088 ms]
[17:37:29.6011760]: [c] REQUEST 1: Experience with d was VeryBad
[17:37:29.7313632]: [c] REQUEST 1: Previous experience: c d Booking 1 0 0 0 VeryGood
[17:37:30.1619824]: [c] REQUEST 1: Updated experience: c d Booking 1 0 0 1 EquallyGoodAndBad
[17:37:30.4323712]: [c] REQUEST 1: Completed

[17:37:30.5725728]: [c] REQUEST 1: End (Successful) [Total Duration: 8772.6144 ms]

Appendix C

Incorporating Prejudice into Trust Models to Reduce Network Overload (Wojcik, Venter, Eloff & Olivier 2005)

In: *Proceedings of SATNAC: South African Telecommunication Networks & Applications Conference*, Champagne Sports Resort, Drakensberg, South Africa, 1-6.

Incorporating Prejudice into Trust Models to Reduce Network Overload (May 2005)

M. Wojcik[1], H.S. Venter[2], J.H.P. Eloff[3], M.S. Olivier[4]

hibiki[1]@tuks.co.za
{hventer[2],eloff[3],molivier[4]}@cs.up.ac.za

Information and Computer Security Architectures Research Group
(ICSA)
Department of Computer Science
University of Pretoria

Abstract— Trust and trust models have invoked a wide interest in the field of computer science. Trust models are seen as the solution to interactions between agents (computer systems) that may not have previously interacted with one another; as is often the case in the uncertain world of e-commerce. These models are seen as facilitators to the definition and development of interaction between two such agents. Trust models rely heavily on a knowledge-building process to evaluate the value, or in some instances to become aware of the risk of communicating with another agent. The observation of other agents or the sharing of knowledge between agents accomplishes this. Thus, trust models rely heavily on the flow of information between machines.

The problem this paper addresses is: How do we lessen the number of communications a single agent has to deal with in order to allow the agent to have sufficient time and resources to accurately analyse these interactions? The suggested solution involves adding a prejudice filter to current trust models. This paper investigates the value of reducing network overload by limiting communication through prejudice and suggests possible filtering factors that can be used in such a scenario. These factors are based on existing security and trust implementations in order to simplify the incorporation of prejudice into current trust models and trust model architecture.

Index Terms— category, certificate, domain, intermediary, learning, organization, prejudice, policy, recommendation, trust, trust models.

This material is based upon work supported by the National Research Foundation under Grant number 2054024. Any opinion, findings and conclusions or recommendations expressed in this material are those of the author(s) and therefore the NRF does not accept any liability thereto.

I. INTRODUCTION

The world of e-commerce is a vast dynamic domain often requiring 'virtual' businesses to communicate and establish contracts in an environment where changes in business customs can be made almost in an instant [1], thus, bringing in the need to define whom one can trust and more specifically how one trusts [2], [3].

Determining and defining whom an agent trusts and to what degree is the core of trust models. In this paper the term agent refers to a computer within an e-commerce environment using a trust model to determine trust. Trust in the world of Computer Science can be defined as an agent's belief in the dependability and capability of another agent; the value more often than not is a result of experiences, observations and/or recommendations [5].

Trust models rely on the collection and analysis of information to form trust opinions. This leads to the interesting problem of how is one to filter out unwanted flooding of communications all vying for analysis. This paper attempts to solve this problem by introducing a prejudice filter to lighten the load of information any agent needs to deal with in order to establish a trust relationship. The paper further investigates how this prejudice filter can be incorporated into trust principles already in existence.

The remainder of the paper is structured as follows. Section II is the background to the paper and serves to define trust, prejudice and trust models. Thereafter, section III investigates where and how prejudice can be implemented within trust models. Finally section IV concludes the paper.

II. BACKGROUND

Concepts dealt with by this paper include trust, trust models, and prejudice. A clear understanding of these concepts is required in order to understand the aims of this paper. This section gives a broad overview of each of these concepts.

A. *Trust*

Trust is a subjective concept unique to each individual and each individual's worldview. It is often dynamic in nature and influenced by environment, state and situation. Nooteboom [6] defines trust as a four-place predicate stating that: "Someone has trust in something, in some respect and under some conditions." The four predicates mentioned here are: the entity trusting (someone), the entity being trusted (something), the reason and goals that define the need for trust (respect) and the conditions under which the trust is given (conditions). Thus trust involves risk.

Trust can be directed to individuals, institutions, organizations as well as socio-economic systems. Trust in systems can result in individual trust where the trust an individual has in another individual is a direct result of the trust the individual has in the organization to which the other individual belongs [6].

B. *Trust Models*

Trust and trust models are an area of interest in the field of Computer Science resulting in the formation of varied machine trust models. Numerous trust models have been studied in an attempt to define a common set of features in order to give a guideline as to what aspects are required in such a trust model [2], [7], [8], [9], [10], [11], [12], [13], [14]. Such an initial set of features common to most trust models as determined from these sources includes the following:

Recommendation of trust: Trust is built by an agent based on the recommendations it receives from other agents.

Dynamic trust: Due to the dynamic ever-changing nature of the interactions an agent has to deal with, trust requires to be continually updated.

Evaluation and ranking of trust: This involves the codification of trust whereby trust levels are given explicit values that can be translated into machine code.

Trust in policies: The policies an agent adheres to define the community an agent belongs to, defining how an agent chooses to build and evaluate trust relationships.

Delegation of trust: A form of recommendation where agents are able to delegate certain rights to other agents.

Webs of trust: Trust is propagated throughout the network in order to create webs of trust to be used by agents in order to accomplish given tasks.

Situation and trust: Trust is highly dependent on the context in which an interaction takes place.

Examples of work done in this field include trust evolution and trust update functions defined by Maarten Marx and Jan Treur [15] and a 'soft security' distributed model defined by Alfarez Abdul-Rahman and Stephen [8]. Maarten defines a trust model that defines and updates trust based on past interactions it had with an agent. While Alfarez Abdul-Rahman and Stephen [8] define a distributed trust model based on the assumption that trust is transitive, and can thus be propagated throughout the system via interaction between agents and define a Recommender Protocol to build social webs of trust.

Due to the dynamic nature of agents, trust requires the continual re-evaluation of the above-defined features. A filtering mechanism can contribute significantly to minimise the amount of workload required to do trust formation. This paper proposes such a mechanism based on prejudice discussed in section III.

Current trust models rely on flooding as an information-gathering phase [16], propagating trust through intricate communication, thus opening the doors to network overload [7]. This is further complicated by allowing agents to take proactive actions towards certain goals as a result of changes in the environment [9]. Prejudice allows trust models to filter through and simplify numerous and complex interactions, lessening the communicative load [15].

C. *Prejudice*

Prejudice can be defined as a negative attitude towards an entity based on stereotype, placing all entities of a certain stereotyped group into the same category [17] and is used to simplify initially a complex interactions.

Prejudice makes use of categorization. Categorization assumes that a group possesses either assumed or imagined characteristics that place them in a particular category. This allows the individual to respond to the group members based on their membership to a specific category rather than on their individual uniqueness.

Prejudice is influenced by culture. A form of cultural prejudice is that of institutional prejudice whereby assumptions are institutionalised. Forms of institutionalising assumptions include policies and practices [17].

III. THE INCORPORATION OF PREJUDICE INTO MACHINE TRUST MODELS

An agent's primary goal is to minimise the risk inherent during communications with other agents. To accomplish this, agents need some method of determining the risk involved during communication with the other agents. Trust models have been proposed as a solution to this dilemma. However, as discussed in section II.B above, trust models rely on information gathering as a primary means of trust formation, that require several messages, carrying the required information, to travel through the network. This leads to network overload in an environment where the number of potentially new, previously unevaluated communications that an agent has to deal with, are vast.

The proposed solution investigates how prejudice filtering can be used to minimise the number of messages that need to travel across the network in order for an agent to successfully formulate trust. Prejudice filters need to limit the number of communications an agent needs to deal with to allow an sufficient time and resources to properly analyse the incoming communications and evaluate trust.

The goal of this paper is to extend the initial set of features described in section II.B by adding prejudice filters to the features. Nine features were explicitly chosen for investigation. These nine features were chosen due to the fact that these are principles already in use by trust models thus simplifying the incorporation of the prejudice filters. Each feature can thus be incorporated into a trust models based on any of these principles. Each feature will be best incorporated into a trust model if it extends the core principle a trust model relies on. For instance a recommendation based trust model could be extended to

include recommendation based prejudice filters. The rest of this paper investigates how this may be achieved.

1. *Prejudice and intermediaries*

Prejudice can be incorporated into trust models by allowing a trusted intermediary to keep a list of agents that are to be trusted. This extends the concept of trusted intermediaries such as Certificate Authorities [1]. However, the intermediaries in this case, are used solely for the collection, grouping, definition and categorization of trust-related data [9]. Prejudice filters can be implemented by allowing an agent to trust 'only' other agents that are trusted by particular intermediaries. Figure 1 illustrates this concept. The intermediary that agent A trusts, trusts only Agents F, D and B, therefore, A will also only trust F, D and B. Searching for trusted agents to communicate with is done through the intermediary leaving the agent with a lighter communication load. This also has a positive affect on network overload due to the fact that all information is gathered by a single intermediary instead of having to flow to all the nodes existing within a network.

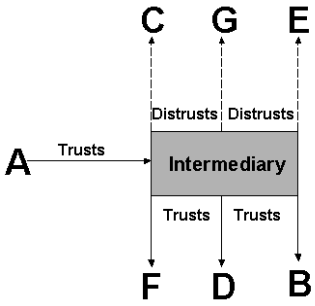

Fig1: Implementing Prejudice by means of an intermediary

2. *Prejudice and categorization*

Different levels of trust can be supplied by segregating agents into categories and levels. Here the agent makes assumptions about the agents it is to interact with and what privileges it will grant. It then evaluates the interaction based on those assumptions [2].

Assumptions lead to categorization. Allowing an agent to make assumptions based on categories reduces the amount of information that needs to be shared between two agents in order to participate in a transaction. This alternately reduces the number of messages and communications travelling across the network.

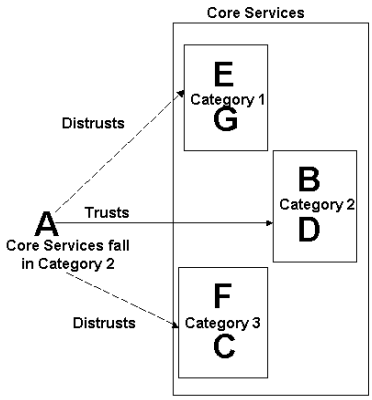

Fig2: Prejudice and Categories

Categorization is a simple way to implement prejudice and its concepts. Agents can be categorized according to their core services, products and policies. Each of these defines the fundamentals of businesses on the web. An agent can filter out and refuse to communicate with other agents that do not adhere to the same core principles [1]. For example, agents can be grouped into categories depending on their core services. An agent will then only trust agents that fall in the same category as it does as shown in figure 2.

Categorization can be further extended, allowing access rights to be divided into 'trust categories'. This allows an agent to define levels of trust by defining the category of an interaction. Thus the rights delegated to an interaction are limited by the category and assumption to which the category implies [8].

3. *Prejudice and policy*

Trust models tend to approach the issue of trust from a generic point and often neglect the fact that agents interact with one another according to the policies and rules defined by the environment or community to which they belong. It is important to define the conventions and prejudices that develop in such communities and use it to evaluate possible interactions with other agents [11].

Fundamental principles followed by communities refer to three basic sources of expectations. These include general rules shared by all agents; social rules that the agent's belonging to a specific group share and the institutional norms, which are defined and enforced by the institution within which the negotiating agents interact [11]. These rules define agent's expectations and include concepts such as privacy policies, what information is used for, encryption policies, transaction contexts as well as other norms used by agents during interactions.

To form a common community agents share their local policies through a process of mutual recursion. Mutual recursion relies on agents sharing one another's local policies and merging them to form a common community policy [12]. The policies are propagated throughout the network in a recursive manner ensuring all agents within the community receive the integration of these local policies.

There is an initial network communication load inherent in this approach during the establishment of a common local policy. However, once this shared community local policy has been established, less communication is required on the network simply due to the fact that any agent belonging to the community can report on and abide by the rules the community has defined. The need to analyse each individual agent from a specific community is removed and any agent wishing to communicate with multiple agents coming from a specific community, can safely assume that the policy held by all agents in the community is the same.

Prejudice filters are a good way to avoid policy misunderstandings simply by allowing the agent to disregard other agents with policies that differ vastly from its own. Possible implementations include dividing agents into world zones. This is possible due to the fact that world zones define varying cultural beliefs. These cultural beliefs tend to dominate the business world in which an agent resides. Examples of this include the varying cultural values between Asian and Western culture. Asian cultures emphasize the community while Western cultures tend to put more emphasis on individualism and public reputation [8].

It is important to define whom the agent you wish to

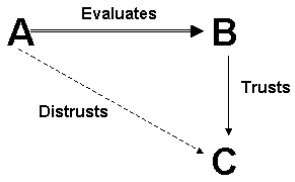

Fig3: Prejudice using Policy

communicate with, trusts. For example, as shown in figure 3, Agent A is in the process of evaluating Agent B to determine whether it trusts B as well as to analyse whom B trusts. A requests B to send a list of the agents that B trusts and analyses this list comparing it to its own. A, knows that it distrusts Agent C. A sees from the list B has supplied that B trusts C. A will give B less trust based on the fact that B trusts C. The reason A will give less trust is because A may not wish to disclose information to B that B may potentially share with C. This interaction results in A being prejudiced against all agents that trust C whether the agents that trust C would disclose information gained from Agent A to C or not [16].

4. *Prejudice and organizational certificates*

There are various mechanisms for identifying other agents one communicates with. Digital signatures vouch for people; computer addresses identify computers; and organizations represent themselves with signed certificates binding together groups of people and computer addresses [2].

These digital signatures and signed certificates can be used to implement forms of 'institution' prejudice [5]. An agent can filter out addresses and digital signatures that do not belong to an organization that it trusts. This is a form of categorization where agents use organizations to determine which agents they communicate with, communicating with only agents that belong to approved organizations [2]. This extends distrust to agents that do not belong to any organization. Using this logic it is clear that Agent A in figure 4 will trust E and G, due to the fact that it trusts organization 1, but distrust B, D, F and C.

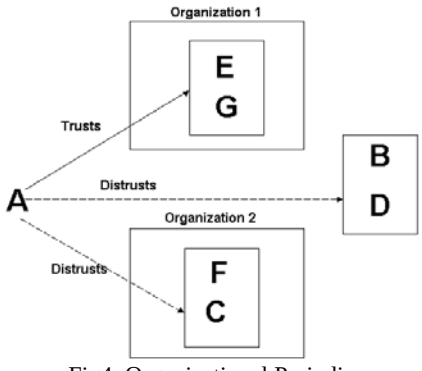

Fig4: Organizational Prejudice

Network traffic is minimised as agents only communicate with and analyse communications coming from agents that belong to trusted agents, ignoring and filtering out all other incoming traffic.

5. *Prejudice and roles*

Roles can be considered as an extension of categorization and are investigated here to see how they can be used to implement prejudice filters.

Using roles is a form of categorization that groups agents into roles and assigns to them the privileges associated with specific roles. Categorizing agents into roles makes prejudiced assumptions about the roles they play, allowing for an agent to differentiate between levels of trust given to other agents. For example, an agent wishing to act as a customer will not be given access to administrative information.

The grouping of agents into roles allows a business agent to define standard actions and privileges to a particular role instead of worrying about defining unique access to each individual agent it comes into interaction with. This simplifies the grey areas that emerge when a trust model evaluates a trust value and attempts to define trust levels. Agents that have met the criteria of a role get assigned the role for the duration of the transaction and get all the privileges and access rights associated with that role only.

Roles can be defined as shown in figure 5. Agent B requests a transaction. This instigates Agent A's analysis process. A first analysis the role into which a specific transaction would requires B to play order to succeed. A then checks if it trusts B to take on the required role. If A's level of trust in B is equal to or higher than that required by the role, A assigns the required role and all privileges to B. The communication between A and B is then limited by the constraints of the role. The limiting of reduces the number of messages that need to pass between agents lowering network traffic.

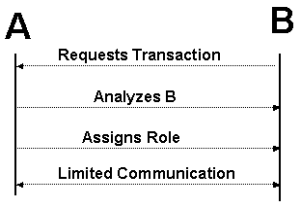

Fig5: Process of establishing role based prejudice

6. *Prejudice and domain*

One way of reducing the number of interactions an agent needs to deal, with is to allow it to only communicate with agents that fall into a limited address range, delegating communication with agents from a different range to another agents within its domain. Prejudice in a domain can be seen as an extension of organisational prejudice. However, the key difference here would be that domain prejudice can be implemented within an organization itself as a feature to limit network traffic.

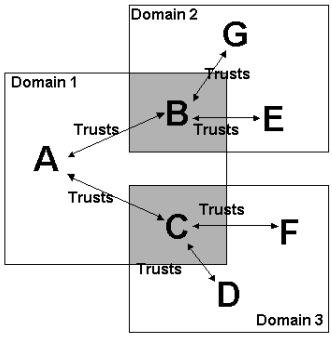

Fig6: Domain based prejudice

This allows each agent to have a limited communication circle even within its own domain, but can still communicate indirectly with agents outside it's domain by

using one of the agents within its domain as a communication intermediary. The limited communication circle reduces the number of messages that flow within the network. Figure 6 illustrates this scenario. A communicates only with B and C. If G wishes to communicate with A, it does so through B, using B as an intermediary. The principle here is that G and E need to move through B to communicate with A while F and D need to move through C.

Here roles are used by an agent to indicate whether it is acting on its own accord or on behalf of another thus preventing another agent from losing trust in a particular agent simply because interactions have failed while the agent was acting as an intermediary [10].

7. Prejudice and path

Path length is already a form of prejudice filter used by agents in a network. This is due to the fact that extremely long paths are seen as untrustworthy. This is not necessarily the case, but this form of assumption allows an agent to cut down on interactions it has to evaluate and at the same time allow it to lower its risk and the network load [16].

Trust in another agent is highly influenced by the path of communication between two agents. The more secure the path, the safer the transaction and, therefore, the higher the possible level of trust. Thus, trust between two agents is also influenced by the trust these two agents have in agents that connect them along the path of communication. An agent has the right to refuse communication with another based on the fact that the path of communication passes through another agent that the agent in question does not trust [13]. This path can be traced and discovered using the tracert protocol to discover the path a message takes [19]. An agent thus refuses to communicate along paths that contain untrustworthy agents in a prejudiced manner.

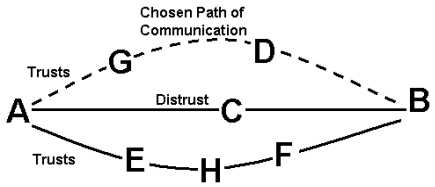

Fig7: Prejudice and path

Prejudice and path length are illustrated by figure 7. Under normal network conditions without the inclusion of trust, communication between A and B would pass through C due to the fact that, that is the path that contains the least nodes between them. However, due to the fact that either A or B distrust C, this path cannot be used. Therefore, the path with the next lowest number of nodes between the two desired points of interaction is chosen, provided that both A and B trust all the agents along that path. The path chosen in figure 7 passes through G and D since there is no distrust indicated along that path.

8. Prejudice and recommendation

Trust models rely heavily on 'soft security' where the key to managing and defining security is that of 'social control'. A way to manage trust often relies on a 'community of agents'. An agent trusts another agent as well as having trust values for the trust it has on the other agent's recommendations. If A trusts B and propagates that trust to C, many trust models allow C to also trust B provided C also trusts A as shown in figure 8 below. If A discovers that it no longer trusts B, it is A's responsibility to propagate the change in trust onto C. C then uses this change in information as well its own information on its experiences in interactions with both A and B to recalculate its trust values [8].

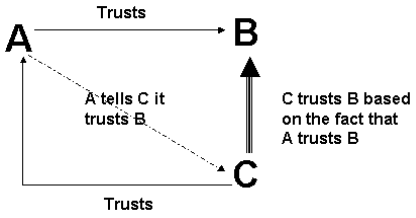

Fig8: Trust and recommendation

Intermediaries discussed in section III.1 as well as other trust models rely on this principle of recommendation to pass on trust values. Trust models that do not use intermediaries use a more social approach where an agent relies on its peers to recommend possible interactions.

Communication and levels thereof between agents can be delegated. If an agent requests a service, the agent the service was requested from, can delegate the responsibility of the service to another trusted agent. This is a form of a recommendation trust model where one agent recommends another. Prejudice can be implemented in a manner that allows an agent to only trust recommendations from a fixed set of delegating agents [16].

Network load is minimised due to the fact that an agent simply does not communicate with any agents that are outside of the community's recommendation circle.

9. Prejudice and learning

Prejudice doesn't necessarily need to be explicitly defined. An agent can learn it. To accomplish this, an agent relies on 'first impressions'. The agent defines a category for the new agent it comes into contact with via an information-gathering process. The agent then proceeds to attempt a transaction with the other agent in question.

If this transaction fails, the agent tags certain general information given by the agent the interaction failed with, for instance, which organization the agent represents, and refuses further transactions from agents from that category.

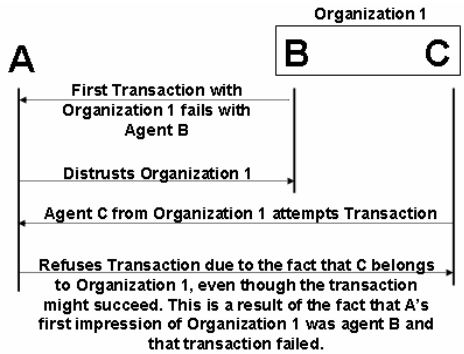

Fig9: Establishing prejudice through first impressions

Figure 9 demonstrates this by means of a sequence diagram. As shown A refuses any transactions from Organization 1 simply based on the fact that its first impression of Organization 1 came from B with whom A's transaction failed. A does not care that interactions with other agents from Organization 1 might succeed. If the transaction succeeds, a trust relationship is formulated. It is

important to note that the first impression is the key one. If the failure in transaction occurs at a later stage it does not terminate the agreement or the relationship between agents but simply lowers the trust value thereof [8].
Network load is thus gradually reduced as an agent learns prejudice.

IV. Conclusion

This paper has introduced the concept of prejudice and investigated how it may be incorporated into current trust models to minimise the interaction load among agents. Means of incorporating prejudice into current infrastructure that have been investigated include intermediaries, categories, policies, certificates, roles, domains, path definitions, recommendations and learning. Each of these areas explore how various measures that are already in use, can be used as means of filtering out the overload of communication on a network.

This paper provided a conceptual discussion, which requires further investigation regarding implementation issues. The initial set of features for machine trust models defined for the purpose of this paper needs to be explored in more detail and should be made more concrete to be used as a guideline for future trust models, as well as for the evaluation of current trust models defined.

More in-depth work needs to be done on protocols for the definition and interpretation of the features discussed, which are required by trust models. These concepts are required for the evaluation of trust. It is hoped that we can move towards some kind of data representation standardization that would allow various agents to easily understand each other and find the information they are looking for when working towards trust formation.

Future work

Future work will include experimental implementation and evaluation of the concepts discussed. Example applications will be designed, allowing the resulting data to be analysed and quantified. Each feature's impact on network performance will be looked at. Finally, a comparison of each feature's impact, under various situational constraints, will be performed quantifying situational influence on performance.

References

[1] Siyal, M.Y. & Barkat, B. 2002. A novel trust service provider for the internet based commerce applications. Internet research: electronic networking applications and policy, 12(1):55-65.

[2] Khare, R., and Rifkin, A., Weaving a Web of Trust. In: World Wide Journal, Volume 2, Number 3, 1997, pp. 77-112.

[3] Britner, M.J., Servicescapes: the impact of physical surroundings on customers and employees, Journal of marketing, Volume 54, 1992, pp. 69-82.

[4] Yang, Y., Brown, l., Newmarch, J., and Lewis, E., eCommerce Trust via the Proposed W3 Trust Model, the PACCS01Conference Proceedings, July 2001, Australia, pp. 9-14 .

[5] Nooteboom, B., (2002) Trust: Froms, Foundations, Functions, Failures, and Figures, Edward Elgar Publishing, Ltd. Cheltenham UK, Edward Elgar Publishing, Inc. Massachusettes, USA, ISBN: 1 84064 545 8.

[6] Guha, R., Kumar R., Raghaven, P., and Tomkins, A., Propagation of Trust and Distrust. International World Wide Web Conference, Proceedings of the 13th international conference on World Wide Web New York, NY, USA SESSION: Reputation networks table of contents, pp. 403 – 412.

[7] Abdul-Rahman, A., and Hailes, S., A Distributed Trust Model. New Security Paradigms Workshop, Proceedings of the 1997 workshop on New security paradigms, Langdale, Cumbria, United Kingdom, 1998, pp. 48 – 60.

[8] Papadopou, P., Andreou, A., Kanellis, P., and Martakos, D., Trust and relationship building in electronic commerce. In: Internet Research: Electronic Networking Applications and Policy, Volume 11, Number 4, 2001, pp. 322-332.

[9] Lamsel, P., Understanding Trust and Security. Available: http://wiki.uni.Lu/secan-lab/ UnderstandingTrustAndSecurity.pdf. Accessed 25 April 2005.

[10] Ramchurn, S. D., Sierra, C., Godo, L. and Jennings, N. R. A computational trust model for multi-agent interactions based on confidence and reputation. In Proceedings of 6th International Workshop of Deception, Fraud and Trust in Agent Societies, , Melbourne, Australia, 2003, pp. 69-75.

[11] Carbone, M., Nielsen, M., and Sassone, V., A Formal Model for Trust in Dynamic Networks. In: Software Engineering and Formal Methods, 2003. Proceedings. First International Conference on 25-26 Sept. 2003, pp. 54- 61.

[12] Datta A., Hauswirth M., and Aberer K., Beyond "web of trust": Enabling P2P E-commerce.In: E-Commerce, 2003. CEC 2003. IEEE International Conference on Publication Date: 24-27 June 2003, pp. 303- 312.

[13] Patton, M.A., and Josang, A., Technologies for Trust in Electronic Commerce. In: Electronic Commerce Research, Volume 4, 2004, pp9-21.

[14] Marx, M., and Treur, J., Trust Dynamics Formalised in Temporal Logic. In: L. Chen, Y. Zhuo(eds.), Proc. of the Third International Conference on Cognitive Science, ICCS 2001, pp. 359-363. Available: http://www.cs.vu.nl/~treur/cve/publ.html. Accesed on 26 April 2005.

[15] Langheinrich, M., When Trust Does Not Compute - The Role of Trust in Ubiquitous Computing. Workshop on Privacy at Ubicomp 2003, Seattle, Washington, October 2003. Available: http://www.inf.ethz.ch/personal/langhein/articles/archive.html #2003. Accessed on 26 April 2005.

[16] Bagley, C., Verma, G.K., Mallick, K., and Young, L., (1979), Personality, Self-esteem and Prejudice, Saxon House, Teakfield Ltd, Westmead, Farnborough, Hants., England, ISBN: 0 566 00265 5.

[17] Held, G., Focus on PATHPING. In: International Journal of Network Management, Volume 11, Issue 4, 2001, pp 259 – 261.

[18] English, C., Nixin, P., Terzis, S., McGettrick, A., and Lowe, H., Dynamic Trust Models for Ubiquitous Computing Environments. Workshop on Security in Ubiquitous Computing, UBICOMP 2002, October 2002. Available: http://www.teco.edu~philip/ubicomp2002ws/organize/paddy. pdf. Accessed: 26 April 2005.

Marika Wojcik was born in Carletonville, South Africa in 1982. Obtained a B.Sc. IT Information and Knowledge Systems degree at the University of Pretoria in 2003, and a B.Sc. (Hons.) Computer Science degree from the University of Pretoria in 2004. Is currently studying her M.Sc. Computer Science majoring in Computer Science at the University of Pretoria.

Dr. H.S. Venter's research interests are in computer and Internet security, including network security, Intrusion detection, information privacy, and digital forensics. He has published in a

number of accredited international subject journals and attended a number of acclaimed international and national, Computer and Information Security conferences, to present his research papers. He is a member of the organizing committee for ISSA and SAICSIT.

Appendix D

Trust Based Forensics: Trust-based Forensics: Applying machine trust models to forensic investigation (Wojcik, Venter, Eloff & Olivier 2006).

In: *Fifth Annual Digital Forensics Conference*, Orlando, Florida, USA, 1-20.

Trust Based Forensics: Applying Machine Trust Models to Forensic Investigation

M. Wojcik[1], H.S. Venter[2], J.H.P. Eloff[3], M.S. Olivier[4]

hibiki[1]@tuks.co.za
{hventer[2], eloff[3], molivier4}@cs.up.ac.za

Information and Computer Security Architectures Research Group
(ICSA)
Department of Computer Science

University of Pretoria

Abstract

Digital forensics is a field of criminal investigation that wishes to find evidence of computer-related crime. This field relies on investigating evidence contained in digital format in order to trace criminal activity. However, as with any investigation, contradictory evidence can severely impede an investigation. Forensic investigators thus need a means to determine which evidence can be trusted. This is particularly true in a trust model environment where computerized agents are allowed to make trust-based decisions thus influencing interactions in the system.

This paper seeks to find a means to analyze evidence contained in such environments as well as to assist the investigator in determining which evidence can be trusted. The means in which this is accomplished is a proposed initial model based on trust-model architecture that can be implemented in a forensic tool in order to do a trust-based forensic investigation.

The proposed model takes into consideration the trust environment and parameters that influence interactions in the suspect network and attempts to recreate the crime in order to test whether the evidence found is in actual fact that of the suspected crime. This newly

created evidence can thus be used to support a given theory should the results of the newly created evidence correlate with the original evidence.

Key words- trust model, digital forensics, evidence, transaction

1. INTRODUCTION

The advancement of technology has opened up new opportunities for criminal activity. The advent of computers has resulted in businesses shifting their internal working to take advantage of these new technologies. This has, at the same time, caused a shift in criminal activities to also take advantage of these opportunities.

Forensics seeks to capture the perpetrators of criminal activity. Digital forensics has emerged in order to keep up with changing technologies and capture perpetrators of computer-based crime. Digital forensics can, thus, be defined, as *"the process of identifying, preserving, analysing and presenting digital evidence in a manner that is legally acceptable"* [McKemmish 1999].

This process is a four-step process that consists of identifying the evidence, preserving the original state of the evidence, analyzing the evidence and presenting the results and conclusions thereof [Ryder 2002]. However, what does an investigator do with a collection of evidence that contradicts itself? Evidence may contradict itself because of an attacker's activity [Stallard & Levitt 2003]. Such evidence is valuable to an investigator when trying to track down such an attacker but can cause a problem when the investigator does not know which evidence can be trusted.

This problem is especially critical when the network containing the evidence in question is running some form of trust model architecture. This is due to the fact that such a network

allows computerized agents to participate in transactions on behalf of a user in order to find the most efficient way to conduct these interactions. Thus, the trust model in place will influence the evidence of suspect activity.

This paper proposes a trust-based model consisting of three phases, discussed later in this paper, to solve this dilemma. An investigator runs a tool, based on the proposed model, in order to determine which nodes in the network are trustworthy and which are dubious. The investigator is also able to determine how the trust model influences the creation of evidence in the specific environment thus leading to more accurate analysis of evidence.

This model can also be used to support the evidence of a suspected crime by recreating the crime and the evidence thereof. This ensures that evidence is complete, reliable and believable when admitted in court. These factors are vital if evidence is to be used in a trial [Ryder 2002].

The paper is structured as follows. Section 2 defines and investigates the background of both digital forensics and trust models. The proposed model is discussed and broken down in section 3. Thereafter, the advantages and shortcomings of this model are investigated by section 4 with section 5 concluding this paper.

2. BACKGROUND

Trust model architectures and digital forensics are both relatively new areas of interest in the field of information technology. Each has several issues that still require to be addressed. Trust models have been proposed in order to change the means in which networks process information. However, this also opens new opportunities for crime. Thus, this paper, merges the two concepts in order to improve the forensic investigative process. Both trust models, and digital forensics are introduced in the sections that follow.

2.1 Digital forensics

Technological advances have resulted in new means of storing, accessing and securing information in order to assist in business functionality. This new means of supporting business functionality has given more value to the intangible asset of information. Thus, this has created new forms of criminal activities designed to take advantage of this new asset. As business and business activities become more sophisticated so the criminal moves towards committing more sophisticated criminal acts.

Thus, in order to keep up with the evolving world of crime, forensic tools have been required to evolve along with technology. This has resulted in a school of forensics known as digital forensics. Digital forensics refers to the gathering of evidence that exists in digital format. This evidence can be gathered from various devices such as computers, cell phones and other digital devices [Baryamureeba & Tushabe 2004].

When gathering evidence, the investigators need to be careful not to compromise the original state of the evidence. It is important to record various configurations and settings of the technology that the evidence is being extracted from as this influences the nature of the data retrieved. A means of ensuring that evidence is not altered or damaged is by using forensic tools to create an image of the suspected media [Svensson 2005]. All forensic investigations are then,

ideally, performed on the image to prevent the booting of the original machine, which could lead to data being altered [Bui, Enyeart & Luong 2003].

Evidence is extracted from a digital device in two different ways. Physical extraction of information relies on the extraction of evidence from the binary data located on the physical disk. This process is not dependent on file systems. Logical extraction is dependent on the file system of a computer and can recover fragmented files that physical extraction is unable to recover from [Svensson 2005].

The evidence searched for includes inculpatory evidence, exculpatory evidence and evidence of tampering. Inculpatory and exculpatory evidence refer to supporting and contradictory evidence, respectively.

The data, from which this evidence can be extracted, can be found in several locations. These locations include slack space, unallocated space on the hard disk, user-created files and computer-created files. User-created files have high forensic value due to the fact that they were personally created. Computer-created files are files created during a computer's normal operation and contain evidence of what interactions a computer has been engaged in.

Forensic evidence can be obtained from either live or dead systems. Live systems contain a lot of information. However, evidence that remains on a live system runs the risk of being tampered with if the system is left in a live state. Switching a live system off runs the risk of losing volatile information, network connections and mounted file systems. It is up to the investigator to decide which risk is greater and to make a decision whether to turn a live system off or not [Svensson 2005].

2.2 Trust models

The issue of establishing trust in order to do business is ever under investigation. New emerging technologies have changed the business world to such an extent that even the means of establishing trust during transactions has had to be revised to keep up with the means in which transactions are being performed. This has led to the formulation of trust models.

In order to fully understand how trust models work, one needs to understand the concept of human trust. Trust is an abstract concept, the exact definition of which is unique with every individual. Trust relies on the formulation of templates in order to group similar situational experiences together. This allows an individual to group various experiences and their associated trust representations.

In order to give trust a more standard definition, Nooteboom [2002] defines trust as a four-place predicate. Nooteboom states [2002]: "Someone has trust in something, in some respect and under some conditions." The individuals participating in a trust relationship in the context of trust models are agents. Agents, for the purposes of this research, refer to non-human, coded entities. These coded entities are defined by a programmer and consist of logical rules [Josang 1997] and restrictions against which interactions are analysed and processed in order to obtain a trust value. A trust value is a value that is calculated by a trust model that indicates the level of trust an agent has in another. The exact values that indicate trust, distrust, and all the areas of partial trust in between depend on the model implemented. Someone and something in the above-mentioned predicate refer to two agents participating in an interaction. Each agent has some form of trust in the other agent. The *respect* under which the trust is given refers to the situational factors that instigated the need for the transaction and the *conditions* refer to the limitations under which the transaction occurs.

Trust models are used to analyze the trustworthiness of other agents. This includes the trustworthiness of information shared between the agents, since this information is often used to make important decisions. Several experts have proposed and formulated various trust models for implementation [Carbone, Nielson & Sassone 2003], [Patton & Josang 2004], [Marx & Treur 2001], [Coetzee & Eloff 2004].

The trust values are obtained and assigned in various ways. Dynamic means of evaluating an agent and calculating a trust value include observation, experience and negotiation. Observation allows an agent to observe the interaction other agents participate in before attempting an interaction. Direct experience allows an agent to participate in a direct interaction and analyzing the outcome thereof [Esfandiari & Chandrasekharan 2001]. Negotiation, on the other hand, requires that two agents share trust-related information contained in their security policies before commencing an interaction [Kagal, Finin, & Joshi 2001].

The result of the trust analysis process is a trust value that is used to restrict the interaction that occurs. This restricts the information that is shared and defines the behaviour of the interaction. Higher trust values result in more free interactions and higher trust in the information shared during these interactions.

Since, the trust model in place influences the means in which interactions are conducted; it also influences the means in which information about these interactions is stored. This has a direct influence on forensic investigations as it influences potential evidence of criminal activity. Trust models also define a level of trust for a particular node. Thus, analysing the levels of trust within a given system can assist in pinpointing suspect nodes that may either contain evidence of suspect activities or contain untrustworthy evidence.

3. DEFINING TRUST IN FORENSICS

In order to be able to make sound judgments based on evidence gathered, an investigator needs to know which evidence can be trusted. This is a matter, often under debate, seeing as criminals, who wish to remain uncaught, tamper with evidence in a manner as to affect the trustworthiness of the evidence. The criminals, hence, impede not only the process of gathering this evidence but also the process of correctly analysing the evidence.

In order to gather evidence, an investigator looks for anomalies, failures and specific results when running various tests on a system. If the information has been tampered with to the degree that the expected anomalies, failures and specific results do not arise, an investigation can be lead away from the source of criminal activity [Peron & Legary 2003?]. As mentioned previously, evidence can be found in several places. Of the various locations, the easiest evidence to tamper with would be that contained in user-created files. Typically computer created files are in obscure locations and protected by the operating system. The criminal requires a higher level of access and a more technical knowledge base in order to access and change this information. User-created files are easier to find, have more access rights assigned to them and are easier to understand.

Computer-created files can also contain a wealth of information and although they may also be tampered with, require more skill and foresight to do so. In a network, relying on trust architecture, these computer-created files are created according to the trust rules that govern the processing of interactions. Trust models define the level of trust given to agents participating in an interaction. This level of trust influences the restrictions that are placed on the interaction itself and the information shared. The restrictions placed by a trust model on interactions influence the manner in which the transactions are conducted and, thus, the computer-created files.

The results of these interactions update the state of trust in the system, influencing the future processing of other similar interactions [Jonker & Treur. 1999], [Xiong & Lui 2003]. Keeping this in mind, it is possible to test for criminal activities that have occurred over the network, by testing the state of the trust relationships within the network as well as the reactions of various nodes to similar activities. A proposed model for using trust models to determine the presence and trustworthiness of forensic evidence is given by figure 1 below.

In figure 1, numbered and labelled directional arrows indicate a process that occurs. These processes occur on four logical components. The first is the original network that is suspected of containing evidence of criminal activity. The second logical component is the copy of the original network, which is made in order to preserve the original state of the original network. Creating a copy of the network is an intricate process. This requires copying over sections of hard disks from the original network. The sections copied are defined by the data of interest to the investigator and should include this data. The relationships between various nodes are copied by copying the network settings of a particular node. Included in these network settings are the files and programs that define the trust model in place. It is important to include these trust defining settings when copying over network settings. Detailed exploration of this process is, however, beyond the scope of this paper.

The third and fourth logical components make up one physical component, the trust analysis unit, but are logically divided into two. The two logical components are the trust analyzer that participates in the various analysis processes and creates a logical trust analyzer node. The logical trust analyzer node is the node that is introduced into the copy of the network and acts as an additional node in the network.

There are three phases to this model, the establishment phase, the evidence-gathering phase and the analysis phase. The first three processes in the diagram are part of the establishment phase. Processes 4 and 5 are part of the evidence-gathering phase and process 6 is a phase on its own, referring to the analysis phase.

The establishment phase sets up the necessary criteria and environmental variables in order to conduct the investigation. The evidence-gathering phase actively gathers evidence for analysis, while the final, the analysis phase, does the final analysis on the evidence in order to reach a conclusion.

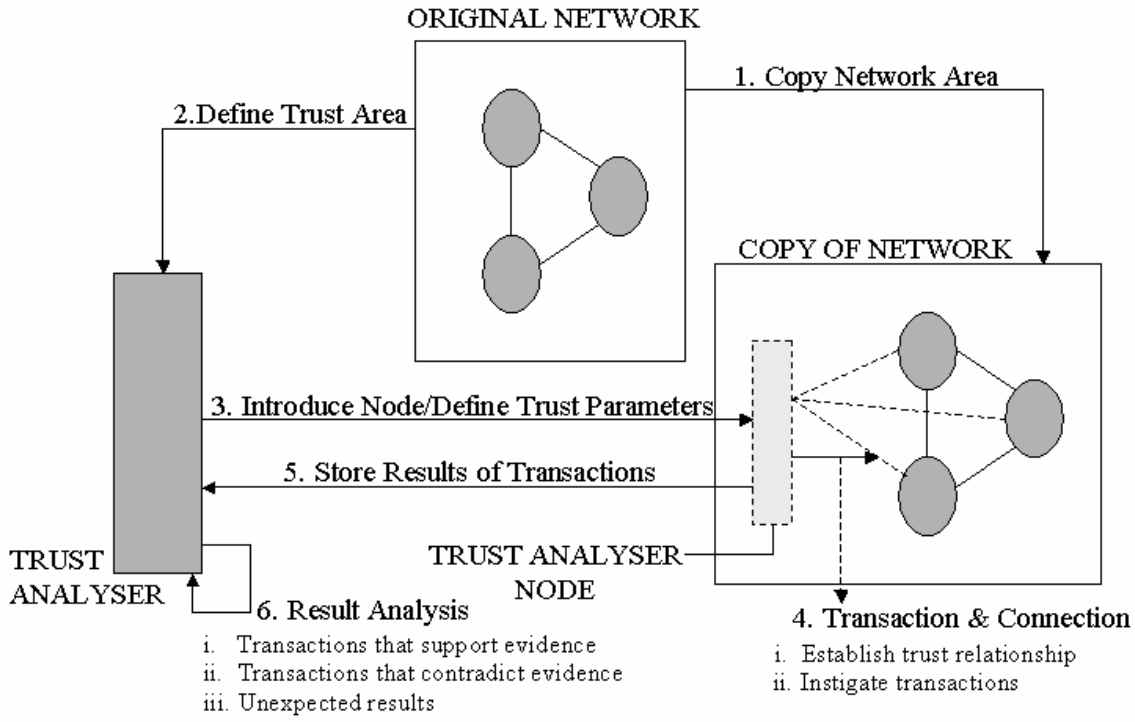

Figure1: Using trust models to gather forensic evidence

The above-mentioned phases are discussed in the sections that follow. The establishment phase and all its processes is discussed in section 3.1. Section 3.2 discusses the evidence-gathering phase. Finally, section 3.3 explores the analysis phase and the various results that can be obtained.

3.1. Establishment phase

The establishment phase of this model begins along with the identification phase of a forensic investigation. This phase is paramount as it influences the progress of all the phases that follow and thus needs to be considered carefully [Ryder 2002].

Once an investigator has identified what evidence is present and how it is stored, the investigator is required to isolate and collect the evidence in such a manner as to prevent the damage and loss of the original evidence while at the same time allowing for this evidence to be analyzed in order to acquire the relevant information required by the investigation [Svensson 2005].

The first step required by this model is to duplicate the network domain the investigator is interested in. This is done in order to preserve the original state of the evidence. Both the links and data of interest are copied to a dead version of the live network.

The model that this paper proposes, makes use of the concept of distributed digital forensics as initially proposed by Roussev and Richard [?]. Distributed digital forensics proposes a tool that relies on a pool of distributed resources in order to successfully analyze evidence contained in a distributed environment. The means in which this concept is implemented by this particular model, is allowing the network in question to be duplicated on several machines in order to simulate the ‘live’ state of the network on a so called ‘dead’ copy. However, it is not necessary to have a physical machine for every physical machine in the target network. Several sub-sections of the network can be duplicated on a single machine. The core nodes that are suspected to have taken an active part in the criminal activity are each placed on their own computer while the nodes that simply support the nodes that are suspect are grouped with the nodes they support in order to create a sub-domain. The use of multiple machines allows for the

duplication of some of the more vital physical links and for a more accurate representation of the live state of the original network when under analysis.

Once the network area of interest has been duplicated, an analysis, seeking to define the appropriate trust attributes, is conducted. This information can be obtained in a variety of manners. One possible manner of determining the trust setup of the network is to query the relevant people involved with the setup of the network in question. Should this not be possible, these attributes can be queried from the network itself by searching for global policies that have been defined. The investigator should also be able to directly access the list of rules governing trust, from any of the physical nodes in the network. Whether trust is built by reputation, observation or direct interaction with new nodes is an instance of rules that influence the trust environment in place. It is important to note that this process must be done with as much, if not more, care as taken in the process of doing the copy of the system in order to simply extract information and not trigger a change in the information state. These rules and are input into the trust analyzer.

The nature of the suspected crime needs to be defined as a set of attributes. For instance, a crime involving the leaking of information could have the means in which the information was leaked and nature of the confidentiality level of the information leaked as attributes. These attributes are also input into the analyser along with any data that may be required to recreate the crime. This information is then also input into the analyser as part of the definition of the trust area process. These attributes are used to determine the transactions that will be attempted in order to recreate the criminal activity.

The trust analyzer then processes these rules and uses the rules and attributes to define a virtual node for the network that runs according to the same rules and attributes defined by the

network. This virtual node is then introduced into the copy of the network where it is required to run and gather trust related information.

3.2 Evidence-gathering phase

The core goal of the evidence-gathering phase is to gather information in such a manner as to assist the forensic investigation. Allowing the evidence to be gathered within the restrictions placed upon the process by the establishment phase does this. The virtual node that has been introduced into the copy of the original network controls these processes.

The driving process of this phase is the transaction and connection process, which is made up of two key sub-processes. The two sub-processes are the trust establishment process and the transactions process. The trust establishment process takes into consideration the trust area and trust parameters it receives from the establishment phase and uses this information to establish communication links with the other nodes in the network. This establishes the trust levels between nodes and ensures that the new node is governed by the same context as the original nodes in the suspect network. The new node thus, instigates transactions in the same manner as nodes in the original network.

In order to successfully recreate the evidence, two key factors are needed. These factors include a clear definition of the nature of the suspected crime and the context in which this crime has occurred. Both of these factors are derived from the establishment phase. Once the trust context has been established, the virtual node does a detailed analysis of the attributes it has received, which are related to the nature of the suspected crime. These attributes are used to deduce interactions that should have occurred in order for the suspected criminal activity to successfully take place.

The transactions process makes use of the trust connections that have been established to try and recreate the forensic evidence that is being questioned. This process relies on the process of recreating the events that created the suspect evidence in order to test whether the results correlate with the suspected crime. Once the nature of the transactions has been defined, the virtual node begins to participate in transactions of the defined nature. The virtual node runs a set of interactions that recreate the suspected crime in order to evaluate the manner in which the system responds to such interactions.

The response of the system to the various transactions is recorded and sent back to the trust analyzer node. Once all the transactions have been finalized, data created by the various nodes, including that created by the virtual node, is collected and returned to the trust analyzer for detailed analysis, which occurs in the third and final stage. This data is representative of the system's final state after the transactions have occurred and is used as a comparison against the original evidence.

3.3 Analysis phase

In this, the final phase, results are gathered and investigated in order to reach a conclusion. The trust analyzer is supplied with the various machine-created data that was created by the various nodes in the network as a result of the transactions that the virtual trust analyzer node instigated. This machine-created data is compared to the machine created data that is part of the suspected evidence of a specific crime.

The results are analyzed along with the trust rules of the system in order to determine how the trust environment in place, influences the representation of the evidence collected. This influence is then taken into consideration during the more detailed analysis phase. The analysis may produce any or a combination of three different sets of results: results that support the

evidence, results that contradict the evidence and unexpected results. Various conclusions may be drawn from obtaining the various sets.

The results that support evidence and contradict evidence are dependent on the fact that the investigator is expecting certain evidence to correlate and other evidence to contradict. If the results of the analysis are what the investigator expected, the investigator has a means of proving that their suspicions are in fact true. Results that correlate are indicative of a successful recreation of a crime and can be used in conjunction with the original evidence in order to prove that the crime of the suspected nature did indeed occur. Results that were expected to be contrary, perhaps due to a suspicion that data had been tampered with, also support a given theory.

Results that are unexpected can be scrutinized in two possible ways, depending on an investigator's initial outlook. These results include results that were expected to correlate and do not as well as those that were expected to be contrary but do in actual fact correlate. If the investigator feels that their theory is sound and they are certain as to what the expected results should be in order to support their theory, unexpected results could mean that the investigator's entire theory and suspicions are incorrect. This would then require the investigator to re-evaluate the evidence they are in possession of and consider alternative possibilities.

If the initial outlook was uncertainty as to which evidence is to be trusted and which is to be disregarded, the model is only run until the trust relationships have been established. The trust relationships are established according to the trust parameters in place. For instance, if the trust model in place is a recommendation-based model the establishment of trust relationships relies on recommendations from trusted nodes. In order to recreate the environment as closely as possible, nodes that trust the suspect node will be modified as to trust the new virtual node to a

similar degree. The investigator needs to keep in mind that sometimes this trust value will need to be rolled back to a different that has since changed due to the effect of the criminal related transactions on the trust environment itself. The degree to which such an environments state can be rolled back will depend on the trust model in place and requires further investigation.

Thereafter, the trust relationship values between the virtual trust analyzer node and the other nodes in the network are analyzed. This is a fairly simple concept as the trust relationship between various nodes is often represented as a single value in its final form. Nodes that obtained a high trust value from the virtual trust analyzer node are seen as more trustworthy than those with lower values. Thus, the evidence contained on these nodes is given a higher probability of being trustworthy.

The evidence of the suspected crime can then be gathered by focusing on these trusted nodes. Evidence gathered from nodes trusted by the network is more likely to be applicable to the crime due to the nature of trust models. Trust models only allow a transaction to take place if the nodes participating in the transaction are trusted. If not, the transaction would not have taken place.

Once this evidence has been gathered, the investigator can continue the evaluation process by entering the parameters of the now-suspected crime into the trust analyzer and allowing the analyzer to continue from the process that established the trust relationships and attempt to recreate the suspected crime. If the results of the analysis are unexpected, the investigator needs to re-evaluate the levels of trust given to the data contained on various nodes searching for discrepancies, which could be indicative of evidence tampering.

4. DISCUSSION

It is important to note that the proposed model is only an initial model and requires extensive further research in order to more firmly establish the exact workings thereof. The proposed model discussed in this paper can be used in two ways. An investigator can use the model to determine which evidence can be trusted and which evidence is suspect as well as to recreate the crime and create supporting evidence to prove the existence of a suspected crime.

This model assumes the network being tested for evidence has some form of trust mechanisms in place that control the interactions that occur in the context of this network. However, this model will also work in networks that do not have an explicit trust model architecture in place. Should such a context be found, the process of defining the trust parameters and trust area will change. Instead of defining a trust model in the same manner as that in the network, a default trust context will have to be fallen back upon. More research needs to be conducted to define the default context.

The re-enacted transactions need to be of a similar nature as those of the suspected crime. However, these transactions must be conducted carefully so that they do not change any data left by the original crime but only add to it. The original data should not be lost as the tests are run on a copy of the system. Should the investigator find that the original data was altered by the attempts to re-enact the crime, the transactions used to recreate the crime need to be analyzed and controlled more carefully. More research in how this control is to be achieved needs also to be conducted.

The resources needed to conduct such an investigation on a large network could be too large to make such an investigation feasible. Should this occur, an investigator has the option of simply copying a critical part of the network and running the tool on that part.

5. CONCLUSION

This paper proposed an initial forensic model using the concept of trust models to evaluate forensic evidence in order to test which evidence can be trusted and which evidence has been tampered with. This model also allows for the crime to be recreated in order to create more substantial proof of a crime that has been committed.

The various phases in order to achieve this result have been investigated from a conceptual view and more research into explicit definition of what happens in each phase and how this is accomplished is required. Area's that warrant attention are how the network is to be copied in order to preserve the trust environment in place, how the protocols will work on the network copy and how to explicitly define the nature of the crime the tool is attempting to re-enact.

An interesting dilemma that arises is the effect of allowing a computerized agent to actively instigate transactions on behalf of a user. In such an environment, an agent is given rights to participate in transactions without the users direct knowledge. Forensics needs to take into consideration the fact that the crime committed could have occurred either from a flaw in the code, or from malicious intents from the programmer of the code of such an agent and not directly from the user. Methods of proving and testing code need to be defined and incorporated into the proposed model.

6. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Research Foundation under Grant number 2054024. Any opinion, findings and conclusions or recommendations expressed in this material are those of the author(s) and therefore the NRF does not accept any liability thereto.

7.REFERENCES

- Baryamureeba, V. & Tushabe, F. 2004. The Enhanced Digital Investigation Process Model. In *Digital Forensics Research Workshop (DFRWS)*, August 2004.
- Bui, S., Enyeart, M. & Luong, J. 2003. Issues in Computer Forensics. http://www.cse.scu.edu/~jholliday/COEN150sp03/projects/Forensic%2520Investigation.pdf. May 22, 2003. Accessed: 12 November 2005.
- Carbone, M., Nielsen, M. & Sassone, V. 2003. A formal model for trust in dynamic networks. In *Software Engineering and Formal Methods, 2003.* 25-26 Sept. 2003, pp. 54- 61.
- Coetzee, M, Eloff, JHP. 2004. Towards Web Services Access Control, In *Computers & Security*, Vol. 23 (7), pp 559-570
- Esfandiari, B. & Chandrasekharan, S. 2001. On How Agents Make Friends: Mechanisms for Trust Acquisition, *In 4th Workshop on Deception, Fraud and Trust in Agent Societies*.
- Jonker C.M. & Treur J. 1999. Formal Analysis of Models for the Dynamics of Trust based on Experiences. In: F.J. Garijo, M. Boman (eds.), *Multi-Agent System Engineering, Proceedings of the 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW'99.* Lecture Notes in AI, vol. 1647, Springer Verlag, Berlin, 1999, pp. 221-232.
- Josang, A. 1997. Prospectives for Modelling Trust in Information Security. In *Proceedings of the Second Australasian Conference on Information Security and Privacy*, pp. 2-13.
- Kagal, L., Finin, T. & Joshi, A. 2001. Trust-Based Security in Pervasive Computing Environments. In *Computer*, Volume 34, Issue 12. Dec 2001. pp 154-157.
- Marx, M. & Treur, J. 2001. Trust dynamics formalised in temporal logic. In *Proceedings of the Third International Conference on Cognitive Science*, ICCS 2001, pp. 359-363.
- McKemmish, R. What is Forensic Computing? June 1999 Australian Institute of Criminology trends and issues No. 118: http://www.aic.gov.au/publications/tandi/ti118.pdf Accessed: 12 November 2005
- Nooteboom, B. 2002. Trust: forms, foundations, functions, failures, and figures. Edward Elgar Publishing, Ltd. Cheltenham UK. Edward Elgar Publishing, Inc. Massachusettes, USA. ISBN: 1 84064 545 8.

- Patton, M.A. & Josang, A. 2004. Technologies for trust in electronic commerce. In *Electronic Commerce Research*, Volume 4, pp. 9-21.
- Peron, S.J., Legary, M. & Labs, S. 2002? Digital Anti-Forensics: Emerging trends in data transformation techniques.
- Roussev, V. & Richard, G. ? Breaking the Performance Wall: The Case for Distributed Digital Forensics. In *Proceedings of the ? Annual Digital Forensic Research Workshop.* http://www.dfrws.org/bios/day2/Golden-Perfromance.pdf. Accessed: 12 November 2005.
- Ryder, K. 2002. Computer Forensics – We've had an incident, who do we get to investigate? http://www.sans.org/rr/incident/investigate.php Accessed: 12 November 2005.
- Stallard, T. & Levitt, K. 2003. Automated Analysis for Digital Forensic Science: Semantic Integrity Checking. In *Proceedings of the 19th Annual Computer Security Applications Conference,* pp. 160.
- Svensson, A. 2005. Computer Forensics Applied to Windows NTFS Computers. Masters Thesis, Stockholm's University / Royal Institute of Technology. Kista, Stockholm, Sweden. April 2005.
- Xiong L. & Lui L. 2003. A Reputation-Based Trust Model for Peer-to-Peer eCommerce Communities. In *E-Commerce,* 2003. CEC 2003. IEEE International Conference on 24-27 June 2003. pg 275- 284

Appendix E

Trust Model Evaluation Criteria: A Detailed Analysis of Trust Evaluation (Wojcik, Venter & Eloff 2006a).

WIP. In: *Proceedings of ISSA: Information Security*, Sandton, South Africa, 1-10.

TRUST MODEL EVALUATION CRITERIA: A DETAILED ANALYSIS OF TRUST EVALUATION

Author and co-authors

M. Wojcik[1], H.S. Venter[2], J.H.P. Eloff[3]

Author's affiliation

Information and Computer Security Architectures (ICSA) Research Group

Author's contact details

Department of Computer Science, University of Pretoria, Pretoria, South Africa.

hibiki[1]@tuks.co.za

{hventer[2],eloff[3]}@cs.up.ac.za

ABSTRACT

The advent of the internet has resulted in the exponential growth in availability of information as well as the sources from which this information can be gathered. Information has become an asset as well as a vital tool for decision making processes. In order to gain new information, a business is often required to give some information, it is already in possession of, up. This information could be of a sensitive nature and a business needs to know if the source it is exchanging information with can be trusted. Trust models have been proposed to solve this dilemma.

Trust models use logical rules to analyze the nature of interactions. An agent, which is a computer running a trust model, analyzes other agents it comes into contact with and determines a trust level. The trust level is a single value that controls all interactions occurring between participating agents. Values above a certain threshold are seen as trusted and values below are seen as distrusted.

The problem is that these trust models were found to be wide and varied, with no common set of features between them, making it difficult to determine which models address which particular issues of the concept of trust. This paper proposes a set of criteria that is to be used to evaluate various trust models in order to identify the issues addressed by specific models. Four main categories into which these criteria fall have been identified. Due to space constraints one of these is discussed in detail followed by an example analysis of a trust model in order to illustrate how these criteria are used during trust model evaluation.

KEY WORDS

Trust, criteria, trust model, trust evaluation

TRUST MODEL EVALUATION CRITERIA: A DETAILED ANALYSIS OF TRUST EVALUATION

1 INTRODUCTION

Trust and the intricacies thereof has been a topic of interest in disciplines such as sociology, psychology, economics, philosophy and even history [1]. The advent of the Internet and e-commerce has triggered a similar interest in the concept of trust within the discipline of Computer Science. The Internet and emerging technologies has caused a shift in the way business is conducted. Businesses strive to create a virtual presence in order to attempt to take advantage of the inter-networked, world-wide scope the Internet provides. This new environment provides a wealth of new opportunities for gathering information, providing new services and participating in business interactions. However, in the same way that this environment provides new opportunities, it also exposes the participants to new levels of risk. With the existence of risk comes the need for trust.

Several experts in the field have defined trust models in order to allow agents within a computerized environment to establish trust [2], [3], [4], [5]. [6]. An agent in this context refers to a computerized agent that has some form of trust mechanism in place. However, the models that have been proposed are wide and varied, each focusing on different aspects of the trust building process.

The problem with all these wide and varied models is that there is no consistent set of criteria that is upheld throughout making it difficult for an interested party to decide upon a particular trust model to implement. This paper attempts to solve this problem by introducing a set of criteria that can be used to analyse a given trust model. The defined criteria are driven at assisting analysis of current trust models. The criteria, as defined by the author, consist of four main categories which contain various influencing factors within. These four categories are trust establishment, initial trust, updating trust and trust evaluation. Due to space constrains, only trust evaluation is discussed in detail.

The remainder of this paper is structured as follows. The background, section 2, explores the concept of trust and how it is currently formalized by experts in the research field of trust models. Section 3 defines the set of criteria that is to be used to analyse a given trust model. A basic analysis is given in section 4 to illustrate the means in which this criteria is used. Finally, the discussion and conclusion of this paper occurs in sections 5 and 6 respectively.

2 BACKGROUND

Trust models rely heavily on the concept of trust, often modelling human psychological ways of forming, establishing and maintaining the concept. In order to fully understand trust model architecture it is important to first explore and define the concept of trust. This section explores the concept of trust and introduces some work already done in the research field of trust models.

2.1 Trust

Trust is such an intricate part of our daily lives that we use it constantly without even thinking about it. It is because it is such an intricate part of our daily activities that it has stoked such interest. However, a clear definition has been hard to come by mainly because trust is a unique concept to each individual influenced by one’s own beliefs, morals and experiences. It is thus accepted that trust is subjective.

Nooteboom [7] defines trust as a four-place predicate stating that: "Someone has trust in something, in some respect and under some conditions." These predicates refer to: the agent trusting

(someone), the agent being trusted (something), the reason and goals that define the need for trust (respect) and the conditions under which the trust is given (conditions).

This definition can be further expanded by including Golembiewski and McConkie's [8] views on uncertainty and hopefulness. With uncertainty comes risk. Without the existence of uncertainty, there would be no need for the concept of trust, as the outcome will be pre-determined. Trust does neither guarantee success nor does it give a failsafe method of analyzing interactions. Without risk, trust does not exist [7]. Trust models rely on these cognitive definitions of trust in order to implement the concept.

2.2 Trust models

Various concepts of human trust have been quantified in order to formulate system trust. These concepts include those of how humans establish trust and use trust in order to formulate templates of expected behaviour and outcome. Several trust models, focusing on various aspects of human trust have been proposed by various experts in the field [2], [3], [4], [9], [10]. Concepts of human trust that experts in the field of Computer Science have chosen to use include are the means in which human trust is established, the various forms of trust as well as the concept of making assumptions based on specific trust templates. However, system trust is allowed to be more dynamic in nature than that of human trust due to the nature of the environment that the need for this trust exists.

Mui, Mohtashemi and Halberstadt [9] focus on building trust through a process of reputation analysis and explore how a process of reciprocation can update trust values in a positive manner. Allowing other agents in a distributed environment to influence the reputation of a particular agent, requires some form of reciprocation mechanism to be in place. A good reputation is built by the reciprocation of agents that are satisfied with a particular agent's performance. The satisfied agent reciprocates by increasing the agent that met its requirements' reputation.

Montaner, López and Lluís de la Rosa [10] look at the specific concept of defining trust through recommendation and thus also explore how to define trust in recommender agents. Recommender agents are the agents that give the recommendations in question. They focus on an opinion-based information filtering method. An agent that is unsure of a recommendation it receives, asks for the opinions of a set of agents it has defined as reliable. The key concept here is that the agents consulted are approached for an opinion on a recommendation that an agent is already in possession of and not a recommendation itself.

These are but a few of the trust models that have been proposed. It is clear that these models are varied, concentrating on different aspects of trust. Finding a commonality and structure among various trust models is often difficult if not impossible. This makes it difficult for an interested party to choose a trust model that best suits specific needs. This paper addresses this problem by defining a set of criteria anyone interested in trust model implementation can use in order to determine a trust model's efficiency. This set of criteria facilitate in identification of the core features of a trust model and the environment it works in.

3 CRITERIA FOR EVALUATION OF TRUST MODELS

Basic trust model architecture can be defined by four broad criteria, as defined by the author. These criteria include trust representation, initial trust, updating trust and trust evaluation. Trust representation looks at how a trust model represents trust-related data. Initial trust defines how a trust model obtains an initial trust value for a node it has previously not interacted with while trust update allows a trust model to dynamically update a trust value over time. Initial trust and updating trust are closely linked as usually the means in which initial trust is obtained, is usually carried through to the means in which it is updated. However, not all trust models follow this scheme. For instance, some trust models have no mechanism in place for establishing an initial trust value, others have none in place to update the trust values. Finally, trust evaluation looks at various

influencing factors on the trust evaluation process. Due to space constraints, trust evaluation is the only one of the four that is discussed in detail in this paper.

3.1 Trust representation

When looking at trust representation, it is important to note that we are looking at the way trust is represented from a holistic point of view. We are not interested in specific variables, their values and storage mechanisms. Rather we are looking at broader concepts that later influence specific trust representation, development and working thereof. Specific concepts include a trust outlook, passionate versus rational trust, centralized versus decentralized trust, trust versus distrust and trust scalability.

An agent's trust outlook defines and agent's predisposition to trust and the means in which it handles trust related data and can be seen as either optimistic (expecting the best outcome), pessimistic (expecting the worst outcome) or pragmatic (balancing both optimistic and pessimistic). ***{jonker and Treus}.*** Passionate trust allows agents to be more flexible develop trust according to intangible principles that more closely emulate the formation of trust between humans. Rational agents are more prone to follow a defined set of rules that are expected to remain static and have the same result every time a trust analysis is done.

In centralised trust environments all agents hold the same beliefs about all other agents. This belief is governed by a central authority. Decentralised trust is for environments where agents are allowed to leave and enter dynamically making it difficult for a central authority to store all trust based information. In this type of environments agents have a more personal trust, each agent having their own unique trust definitions for other agents. Including means in which to explicitly indicate distrust measures removes the uncertainty of not knowing whether distrust is assigned by a model due to lack of information or because of bad behaviour. Finally, scalability considers various characteristics of a model in order to determine whether a model is scalable between small and larger environments.

3.2 Initial trust

Initial trust refers to the strategy a trust model adopts in order to obtain an initial trust value for another agent in the environment. There are several strategies that can be chosen and used. These include but are not limited to recommendation, reputation, observation, institution, collaboration, negotiation and experience [1], [2], [3], [4], [9], [10]. All of these require that the model gather some form of information in order to establish a trust value before participating in an interaction with another agent. Initial trust is a means of establishing a trust value in order to interact with an agent that has previously not been encountered, before experiences that can be judged have been obtained. The defined methods of building this form of trust are based on those found in human psychology. For instance, reputation based trust is based on the human tendency to trust based simply on hearsay and reputation.

3.3 Updating trust

Due to dynamic trust environments influencing agents and trust contexts, a means of updating trust is vital. This allows an agent to change a trust value over time in order to keep up with changing environmental and situational factors rapid changes in technology and development are prone to cause. Agents are, thus, able to protect themselves from other agents that used to be trustworthy but became malicious over time. Often trust is updated using the same mechanisms that were used to define initial trust though there are a few additional factors that need to be taken into consideration. These are the influences of direct experience, feedback and transaction analysis. Direct experiences and feedback are merged along with the methods of establishing initial trust in order to obtain a more personal evaluation of trust.

3.4 Trust evaluation

In order to successfully update trust, a trust model needs a means of evaluating the information it has gathered. Many trust models rely on categorisation in order to limit the trust given to an interaction. This trust categorisation is used to evaluate an interaction and occurs in many various ways. Various trust models rely on the means in which they establish initial trust and update trust in order to evaluate trust and form various categories. However, there are factors other than the ones that are used to form trust that influence the success of interactions. Good trust models take these factors into consideration when evaluating the result of an interaction. These factors include trust context and risk. When taking these factors into consideration, a trust model often is required to do a more detailed analysis of trust. Trust models take two general approaches towards the detail to which trust is evaluated: that of approximate or dynamic evaluation. All of these concepts are important to consider when evaluating a trust model and are discussed in the sections that follow.

3.4.1 Trust categorisation

Trust categories allow for an agent to specify the specific context in which a certain relationship is given trust. For instance, certain agents may be trusted to forward data but may not be trusted to analyze the data [11]. The means in which categorisation occurs, influences the trust assigned to another agent. Two main means of categorisation have been identified by Li, Valacich and Hess namely Reputation (second hand knowledge) and stereotyping [12].

Reputation allows an agent to categorize other agents as a direct result of reputation information it receives about other agents. Various reputation-based models have various ways of implementing this form of categorisation and various actual categories into which these agents can fall. For instance, reputation can be gathered through recommendations, or through observation of an environment. Section 4 gives an example of one such model and the categories into which the model places other agents.

Stereotyping allows for agents to make assumptions about other agents depending on which category an agent falls into [13]. For instance, agents can be categorized according to which organization they belong to, the rights they possess, the roles they play and even the policies they are required to adhere to. Each category is in possession of rather specific privileges and rights associated with it [14]. For instance an agent assigned an administrative role will be given access to far more information than an agent simply assigned a client role.

The means in which this categorisation occurs influences a model’s efficiency on various environments. Stereotyping works very well in environments where specific categories can be explicitly defined and tend to remain static over time. This means of categorisation allows for faster analysis than that of simple reputation because an agent can simply ask for category information and make assumptions based on thus. Forming categories from reputation information requires a higher processing overhead but is more effective when no static explicitly defined categories exist, allowing agents to create their own categories as they are required. It is also possible to leave specific categories for reputation information undefined and allow the reputation value for an agent o be a category on its own.

3.4.2 Context

Trust is entirely based on situation. An agent needs to take into consideration situational constraints before it chooses to participate in a trust-based interaction. Such constraints include capability, need for an interaction and state of the environment [3]. Often the environment changes due to extenuating circumstances, such as a crash of a critical machine or hacker activity. Agents need to be able to detect such changes in the system and take alternative action, especially if such unexpected changes have influenced a critical part of the system as a whole. Shrobe, Doyle and Szolovitz[15] recommend that systems have a means of determining to what degree various agents

can be trusted so that agents may know which contingencies can be taken and which other agents can be trusted with sensitive information should a critical node in the environment be compromised.

A contextual factor that may influence the formation of trust is intention. The reason an agent seeks to establish a trust relationship is a clear indicator of a need that has been expressed. It is also the determining factor as to which information will be shared and which will be withheld. The context a transaction occurs in is also critical when evaluating the feedback that is received. For instance, the size of a transaction is indicative of the effort that was required in order to successfully complete the transaction. Small transactions carry lower risk than large transactions and so success in a small transaction should have a smaller influence on trust than one on a large interaction.

Contexts include both transactional context as well as environmental context. Transactional context include size, category and time of a transaction. Environmental contexts are more concerned with the state of the environment at the time of a transaction. Such contexts include which agents were running and which were not, suspicious increase in activity, network overload and addition of new agents [4]. Context is, hence, a means of controlling various interactions and it is important to determine whether a trust model takes this into consideration. Trust model implementations that neglect to take context into consideration result in poor trust evaluations due to the depth to which context influences interactions. When choosing a trust model implementation it is also important to note what type of contexts are taken into consideration. In order to find a trust model that works optimally for a certain business environment it is important that the type of context that is taken into consideration be the same as the contexts that most influence the business itself. For instance, a business with high risks needs to find a trust model that considers the risk of interactions before assigning trust values.

3.4.3 Risk

All trust relationships carry some form of risk factor with them. This applies not only to business context but any contexts where an exchange of information or service is required. In order to successfully handle this factor, risk needs to be assimilated into the decision-making process and accepted as inevitability. Helpful ways of dealing with risk is by having fallback mechanisms to do some form of damage control should the worst case scenario actually come about. Knowledge of risk allows for making plans that take this risk factor into account and so this factor can by no means be ignored [1]. A means of controlling the risk encountered in an interaction is by placing constraints on both the truster and the trustee while requiring the interaction to take place within these predefined constraints.

The very presence of risk in trust relationships has both the potential to positively and negatively influence trust levels. Successful completion of an interaction that is laced with risk boosts the trust level. Unsuccessful completion can seriously cause trust value to drop, especially if the risk was large and serious harm has resulted from the interaction.

It is important to note that risk is not the same as uncertainty and ignorance. It situations of risk, the various states of the environment and their probable influences on the interaction are known. In uncertainty the probable influences of the states are not known while in ignorance an agent may be unaware of even the existence of the states that could influence an interaction [1]. This brings to fore the fact that the very existence of uncertainty and ignorance is a risk to the system. An agent can work under any of these three states depending on the knowledge it had access to prior to encountering an interaction with another agent whose trustworthiness is in question.

A means of determining and handling the risk inherent in forming trust relationships is a vital component in a trust model. There is a trade off between the risk a trust model takes during interactions and the processing required analyzing another agent. The less detailed the analysis the larger the risk will be which leads us to the concept of dynamic versus approximate evaluation.

3.4.4 Dynamic versus Approximate evaluation

When implementing trust architecture it is important to decide on the means in which one wishes one's trust evaluation to occur. When participating in trust evaluation there is a clear trade off between the accuracy of trust evaluation and the processing power required to do a trust evaluation. In order to have an accurate trust evaluation a more dynamic approach is taken that continually incorporates changes in the environment and agent's interactions into trust evaluation. This is clearly a time-costly procedure. In order to save on processing and time, an agent can choose to do a more approximate evaluation of trust. This alternately carries the risk of a trust evaluation being inaccurate [4].

These four broad criteria categories have been proposed in order to ease analysis of currently proposed trust models. The next section demonstrates how this analysis process works for trust representation. An existing trust model is chosen and analysed using various points within the trust representation criteria category.

4 EXAMPLE ANALYSIS OF TRUST EVALUATION

The trust model chosen in order to demonstrate a sample analysis using trust representation is Abdul-Rahman and Hailes' Trust-Reputation model [16]. This model was chosen due to its relative simplicity. This model determines the trustworthiness of other agents based on collected statistics that include direct experiences and recommendations.

A subset of these direct experiences and recommendations is summarized in order to obtain a trust value. Experiences are recorded into two separate sets in order to be able to differentiate those that are a result of direct trust and those that are linked to recommendation. This way an agent is able to keep track of the direct trust it has in others as well as the trust level it has in recommendations from other agents.

In order to obtain a direct trust value for a particular agent in a particular context, this model relies on a basic system of counters. For every agent within a specific context, an agent possesses four counters. These counters are for varying trust degrees. The four counters this model makes use of are counters or very good, good, bad and very bad. Hence, these counters represent various trust levels namely very trustworthy, trustworthy, untrustworthy and very untrustworthy respectively. These counters are incremented with direct experiences an agent has. The trust model runs a max function on these four counters that return the counter that has the largest value for the specific agent in a specific context. The trust counter with the largest value indicates the trust level that is to be assigned.

Trust values that agents receive as recommendations rely on a semantic closeness value for analysis. This semantic closeness value determines how closely a recommender agent's perception of trust with another agent resembles that of the agent seeking a recommendation. Because trust is a subjective concept, it is logical to assume that perception of trust will differ among different agents. What one agent may define as trustworthy could be seen as untrustworthy by another agent based simply on differing perceptions. Thus, agents seeking recommendations are allowed to adjust recommendations it receives according to their own perceptions.

In order to accomplish this, an agent needs to obtain an adjustment value. In order to obtain this adjustment value, an agent compares its own trust evaluation result for a particular context with the trust evaluation result of the recommender agent in the same context. If the trust evaluation result differs, the agent creates an adjustment value for recommendations coming from that particular recommender in that particular context.

For example, agent A is looking for an adjustment value for recommender B. Agent A knows that the trust value it has for agent D is 't' (trustworthy) in context C. A receives information from B that B's trust value for D in context C is 'vt' (very trustworthy). In other words, A's value for agent D in the same context as B's value for the same agent is one level lower than B's. The

adjustment value that A obtains will be -1. This value is then used to lower the trust value of all recommendations coming from agent B by 1 in order to more closely represent A's own perceptions.

This trust model is analysed using the concepts discussed under the trust representation section as follows.

Categorisation: The type of categorisation this model uses is the reputation based categorisation. This trust model creates its own rather unique categorisation system as a result. Agents are not only identified, but interactions with them are categorized by the context in which they occur. The contexts are left open to definition and any required context can be chosen. Every context with a particular agent has four counters associated with it. The highest of these four counters indicates the trust level associated with the particular context and agent. Once trust evaluation of the counters occurs, the resulting agent's trust level can fall into any of five main categories that influence the trust given the interaction. These are 'vt' very trustworthy, 't' trustworthy, 'u' untrustworthy, 'vu' very untrustworthy and uncertainty which is a mixture of a trustworthy and untrustworthy value. Uncertainty is achieved when there is more than one maximum value for the four types of possible experience counters. The uncertainty assigned can lean towards positive trust, negative trust or remain neutral depending on where the uncertainty lies. If there is uncertainty between very good and good trust level then it is a positive uncertainty. If there is uncertainty between very bad and bad it is negative uncertainty. All other combinations result in neutral uncertainty.

Context: This model specifically takes context into consideration when dealing with other agents. In fact, context is seen as so important that the same agent will have varying trust values in different contexts. This model requires an agent to save trust levels in agent-context pairs so that when trust is determined it is not only the particular agent, but the context as well that is looked up. The context is left undefined so that various forms of contexts can be chosen by anyone wishing to implement the particular model.

Risk: This model has no explicit consideration of risk but contains several implicit risk management mechanisms. This is explicitly defined as a reputation-based model. By definition, such models usually rely on second hand information in order to determine a trust value for another agent. This method inherently carries the risk of receiving false information from another. This risk is addressed by allowing agents to incorporate both direct as well as indirect information together. Agents are also required to analyse the worth of another agents recommendations by comparing a decision the recommender has made about another agent in a given context with the decision the agent itself has made about the same other agent in the same context. The result of this analysis is an adjustment value that an agent uses to adjust all recommendations coming from that specific recommender. Another means of handling risk is by allowing an agent to assign an uncertainty trust level if there is not enough information or if information the agent has gathered is contradictory. Experiences increment the counters associated with a particular agent in a particular context with the result of the experience. For instance, good experiences increment the good counter mentioned above. This continual process of updating the value allows for changes in the system to be echoed by changes in trust thus lowering the risk of having a trusted agent suddenly become malicious.

Dynamic versus Approximate Evaluation: This is a dynamic approach allowing an agent to record a set of experiences that it later uses, in order to determine a trust value. Experiences are updated and the updates influence further trust evaluation for agents in specific contexts. Agents gather recommendations from other agents and merge them in order to obtain a global, more specific, analysis. As more experiences are stored, the evaluation process becomes longer and the danger of running out of space when dealing with large number of agents and interactions exists.

5 DISCUSSION

This paper addresses the problem of analysing various trust models in order to determine the value of a trust model and pinpoint issues that have not been addressed. Four main categories of criteria have been identified by the author. These categories define the main features any good trust model is required to have. The main categories identified are trust representation, initial trust, updating trust and trust evaluation.

Due to space constraints, only one of the four criteria categories has been discussed in detail. The chosen category is that of trust evaluation. Trust representation has already been discussed in another paper submitted for review [17]. Initial trust and updating trust were not discussed due to their length and space constraints. The first three categories, therefore, have not been addressed in this paper in detail. Factors influencing the means in which trust is evaluated that have been identified and discussed in this paper include trust categorisation, trust context, risk and whether the trust evaluation process is dynamic or approximate.

6 CONCLUSION

This paper introduced and discussed the concept of a set of criteria that is to be used for the evaluation of trust models. These criteria are intended to be a guideline to trust model evaluation in order to identify the worth of a particular trust model as well as the areas in which a trust model lacks attention. In the same way that they can be used to evaluate a currently implemented trust model, they can also be used as a guide for factors and issues to take into consideration for future trust model implementations.

Using these criteria, one is able to identify how a trust model addresses certain issues and also which issues have not been addressed. This is important knowledge to have when considering choosing a particular trust model for implementation and can also be used to guide future improvements on trust model architectures.

Abdul-Rahman and Hailes' Trust-Reputation model was taken as an example and analysed using the factors identified within the trust evaluation criteria category. It is important to realize that this criteria are not necessarily all the possible defined criteria that can be taken into consideration. They are based on 'known' issues and known implementations and it is possible to expand them in future work. Future work also includes the development of a measurement mechanism in order to score the degree to which a trust model addresses the issues identified.

7 ACKNOWLEDGEMENTS

This research is funded through the Centre of Excellence in Teletraffic Engineering for the Information Society (CeTEIS), Telkom SA Limited. Any opinion, findings and conclusions or recommendations expressed in this material are those of the author(s) and therefore Telkom does not accept any liability thereto.

8 REFERENCES

[1] Marsh, S.P. Formalising Trust as a Computational Concept. In *Dissertation for the Department of Computing Science and Mathematics*, University of Stirling. 1994.

[2] Li, X. Lyu, M.R. & Liu, J.. A Trust Model Based Routing Protocol for Secure Ad Hoc Networks. In *IEEE Aerospace Conference Proceedings.*. 2004. ISSN: 0-7803-8155-6/041

[3] Wang, Y. & Vassileva, J. Bayesian Network-Based Trust Model. In *Proceedings of Web Intelligence, 2004*. WI 2004. IEEE/WIC?ACM International Conference on 20-24 Sept 2004. pp 341-348. ISSN: 0-7695-2100-2

[4] Li. X. & Liu, L. PeerTrust: Supporting Reputation-Based Trust for peer-to-Peer Electronic Communities. *In IEEE Transactions on Knowledge and Data Engineering*, Vol. 16 No. 7

July 2004.

[5] Khambatte, M., Dasgupta, P. & Dong Ryuu, K. A Role-Based Trust Model for Peer-to-Peer Communities and Dynamic Coalitions. *In ACM Computing Surveys (CSUR).* Vol. 36. Issue 4. 2004. pp 335-371. ISSN: 0360-0300

[6] Wen, W. & Mizoguchi, F. An Authorization-based Trust Model for Multi-agent Systems. 1999? http://www.istc.cnr.it/T3/download/aamas1999/Wen-Mizoguchi.pdf. Last Accessed: 23 April 2003.

[7] Nooteboom, B., (2002) Trust: Froms, Foundations, Functions, Failures, and Figures, Edward Elgar Publishing, Ltd. Cheltenham UK, Edward Elgar Publishing, Inc. Massachusettes, USA

[8] Golembiewski, Robert T., & McConkie, Mark. 1975. The Centrality of Interpersonal Trust in Group Processes. Chap. 7, pages 131–185 of: Cooper, Cary L. (ed), Theories of Group Processes. Wiley.

[9] Mui, L., Mohtashemi, M. & Halberstadt, A. A Computational Model of Trust and Reputation. In *Proceedings of the 35th Hawaii International Conference on System Sciences.* 2002

[10] Montaner, M., Lopez, B. & Lluis de la Rosa, J. Developing Trust in Recommender Agents. In *Proceedings of the First International Joint Conference on Autonomous and Multi Agent Systems* (AAMAS'02). Bologna (Italy) 2002.

[11] Pirzada, A.A. & McDonald, C. Establishing Trust in Pure Ad-hoc Networks. In *Research and Practice in Information Technology*. Vol. 26 V. Estivill-Castro, Ed. 2004.

[12] Li, X., Valacich, J.S. & Hess, T.J. Predicting User Trust in Information Systems: A Comparison of Competing Trust Models. *In Proceedings of the 37th Hawaii International Conference on Systems Sciences.* 2004.

[13]Khare, R., & Rifkin, A., Weaving a Web of Trust. In: *World Wide Journal*, Volume 2, Number 3, 1997, pp. 77-112.

[14] Shand, B., Dimmock, N. & Bacon, J. Trust for Ubiquitous, Transparent Collaboration. In *First IEEE Conference on Pervasive Computing and Communications.* Ft. Worth, Texas, USA. 2003.

[15]Shrobe, H., Doyle, J. & Szolovitz, P. Active Trust Management for Autonomous Adaptive Survivable Systems. In *Proposal to the Defense Advanced Research Projects Agency in response to BAA #00-15 "Information Assurance and Survivability (IA&S) of the Next Generation Information Ifrastructure (NGII)".* 1999.

[16] Abdul-Rahman, A. & Hailes, S. Relying on Trust To Find Reliable Information. In *Proceedings of DWACOS'99, Baden-Baden, Germany.*, 1999.

[17] Wojcik, M., Venter, H.S. & Eloff, J.H.P Trust Model Evaluation Criteria: A Detailed Analysis of Trust Representation. Submitted for review for SATNAC 2006

Appendix E

A detailed analysis of trust representation as a trust model evaluation criterion (Wojcik, Venter & Eloff 2006b).

In: *Proceedings of SATNAC: South African Telecommunication Networks & Applications Conference*, Spier Conference Centre, Stellenbosch, South Africa, 1-6.

A detailed analysis of trust representation as a trust model evaluation criterion (May 2006)

M. Wojcik[1], H.S. Venter[2], J.H.P. Eloff[3]

hibiki[1]@tuks.co.za
{hventer[2],eloff[3]}@cs.up.ac.za

Information and Computer Security Architectures Research Group
(ICSA)
Department of Computer Science
University of Pretoria

Abstract— Technology and the advent of the Internet have resulted in an environment that is both dynamic and changing. This dynamic environment has resulted in the need for trust establishment between agents that have not previously interacted before. In order to address this need, trust models have been proposed in order to handle the establishment and evaluation of trust. Trust models rely on logical rules to analyze the nature of interactions between two agents. An agent in this context is usually a computer running a trust model, analyzes other agents it comes into contact with and determines a trust level.

However, due to the novelty of the subject matter, the trust models currently proposed are wide and varied. Currently, there is no common set of features that have been standardized throughout trust model implementation and no means of evaluating various trust model implementations exists. This paper proposes a set of criteria that is to be used to evaluate various trust models in order to identify the specific trust-based issues addressed by a particular trust model. Four main categories into which these criteria fall, have been identified. Due to space constraints only one of these categories is discussed in detail followed by an example analysis of a trust model in order to illustrate how these criteria are used for trust model evaluation.

Index Terms— Trust, criteria, trust model, trust representation

This research is funded through the Centre of Excellence in Teletraffic Engineering for the Information Society (CeTEIS), Telkom SA Limited. Any opinion, findings and conclusions or recommendations expressed in this material are those of the author(s) and therefore Telkom does not accept any liability thereto.

I. INTRODUCTION

Trust and the intricacies thereof has been a topic of interest in disciplines such as sociology, psychology, economics, philosophy and even history [1]. The advent of the Internet and e-commerce has triggered a similar interest in the concept of trust within the discipline of Computer Science. This new environment provides a wealth of new opportunities for gathering information, providing new services and participating in business interactions. However, in the same way that this environment provides new opportunities, it also exposes the participants to new levels of risk. With the existence of risk comes the need for trust.

The dilemma that emerges in the field of Computer Science is that of defining trust among agents that may not have interacted before. Interacting within a dynamic environment such as the Internet allows agents that may not have interacted before to interact with one another. In order to somehow manage the risk that comes from interacting with unknown nodes, a means of defining trust is required.

Several experts in the field have defined trust models in order to allow agents within a computerized environment to establish trust [2], [3], [4], [5]. An agent in this context refers to a computerized agent that has some form of trust mechanism in place. However, the models that have been proposed are wide and varied, each focusing on different aspects of the trust building process.

The problem with all these wide and varied models is that there is no consistent set of criteria that is upheld throughout making it difficult for an interested party to decide upon a particular trust model to implement. This paper attempts to solve this problem by introducing a set of criteria that can be used to analyse a given trust model. Another goal of this set of criteria is to allow for easy identification of possible lack in current trust model architecture and to assist in guiding the development of future trust models by defining what is required. The criteria, as defined by the author, consist of four main categories which contain various

influencing factors contained within. These four categories are trust establishment, initial trust, updating trust and trust evaluation. Due to space constrains only trust representation is discussed in detail.

The remainder of this paper is structured as follows. The background, section II, explores the concept of trust and how it is currently formalized by experts in the research field of trust models. Section III defines the set of criteria that is to be used to analyse a given trust model. A basic analysis is given in section IV to illustrate the means in which this criteria is used. Finally, the discussion and conclusion of this paper occurs in sections V and VI respectively.

II. Background

Trust models rely heavily on the concept of trust, often modelling human psychological ways of forming, establishing and maintaining the concept. In order to fully understand trust model architecture it is important to first explore and define the concept of trust. This section explores the concept of trust and introduces some work already done in the research field of trust models.

A. Trust

Trust is such an intricate part of our daily lives that we use it constantly without even thinking about it. However, a clear definition has been hard to come by mainly because trust is a unique concept to each individual influenced by one's own beliefs, morals and experiences. It is thus accepted that trust is subjective.

Nooteboom [6] defines trust as a four-place predicate stating that: "Someone has trust in something, in some respect and under some conditions." These predicates refer to: the agent trusting (someone), the agent being trusted (something), the reason and goals that define the need for trust (respect) and the conditions under which the trust is given (conditions).

This definition can be further expanded by including Golembiewski and McConkie's [7] views on uncertainty and hopefulness. With uncertainty comes risk. Without the existence of uncertainty there would be no need for the concept of trust as the outcome will be pre-determined. In essence, in order to trust, one has to be willing to submit to the risk that the trust may fail, but expecting that it will not. Trust does neither guarantee success nor does it give a failsafe method of analyzing interactions. Without risk, trust does not exist [6].

Trust is a dynamic concept that changes over time depending on the experiences and situational factors that influence the formation thereof. Many situational and cognitive factors influence the formation of trust among humans. These factors include concepts such as reputation, recommendation, observation and experience. Since humans build on the concept of trust, these factors are also found among computerised trust model definitions.

B. Trust Models

Several trust models have been proposed by various experts in the field. These models focus on various aspects of the concept of trust. Various computing environments present various challenges when considering trust formation. Establishing trust in the environment of ad-hoc networks is explored by Pirzada and McDonald [8]. Pirzada and McDonald approach the formation of trust in such an environment from a more passive perspective by allowing agents to observe one another in passive mode before attempting to participate in any interactions. Information is gathered through analysis of forwarded, received and overheard packets. Certain events are recorded, categorized and analysed in order to obtain a trust evaluation.

Pirzada and McDonald's trust model illustrates the formation of indirect trust requiring agents to form trust before interacting with another agent. Trust can also be established directly by allowing direct interactions with an agent in question. A means of establishing a direct trust is formalized by Jonker and Treur.[9]. Jonker and Treur define a trust evolution function by using a mathematical function to formalize the influence of past experiences on the level of trust; whereby sequences of experiences are converted into trust representations.

These are but a few of the trust models that have been proposed. It is clear that these models are varied, concentrating on different aspects of trust. Finding a commonality and structure among various trust models is often difficult if not impossible. This makes it difficult for an interested party to choose a trust model that best suits specific needs. This paper addresses this problem by defining a set of criteria anyone interested in trust model implementation can use in order to determine a trust model's efficiency.

III. Criteria for Evaluation of Trust Models

Basic trust model architecture can be defined by four broad criteria, as defined by the author. These criteria include trust representation, initial trust, updating trust and trust evaluation. Trust representation looks at how a trust model represents trust-related data. Initial trust defines how a trust model obtains an initial trust value for a node it has previously not interacted with while trust update allows a trust model to dynamically update a trust value over time. Initial trust and updating trust are closely linked as usually the means in which initial trust is obtained, is usually carried through to the means in which it is updated. Finally, trust evaluation looks at various influencing factors on the trust evaluation process. Due to space constraints, trust representation is the only one of the four that is discussed in detail in this paper.

A. Trust representation

When looking at trust representation, it is important to note that we are looking at the way trust is represented from a holistic point of view. We are not interested in specific variables, their values and storage mechanisms. Rather we are looking at broader concepts that later influence specific trust representation, development and working thereof. Specific concepts include a trust outlook, passionate versus rational trust, centralized versus decentralized trust, trust versus distrust and trust scalability.

1) Trust Outlook: An agent's trust outlook refers to an agent's disposition to trust. This further refers to the extent to which an agent is willing to depend on another given specific context [10]. An agent's trust outlook influences the means in which a trust value is developed and maintained.

Jonker and Treur [9] have identified six means which define an agents trust outlook. These are defined as follows:

Blindly positive: A blindly positive agent starts from a low trust value and looks for positive experiences with other agents. Once it reaches unconditional trust it remains there indefinitely. Unconditional trust is a state in which an agent trusts another unconditionally and does not limit the interactions participated in. Only positive experiences are evaluated and used to increase the trust value. Negative experiences are ignored.

Blindly negative: A blindly negative agent is the mirror case of the blindly positive one, starting from a high trust value for other agents and dropping the trust value as a result of negative experiences. Once it reaches unconditional distrust it remains there indefinitely. Unconditional distrust is a state in which an agent completely distrusts another and under no circumstances will participate in any interactions with that agent. Only negative experiences are evaluated and used to decrease the trust value. Positive experiences are ignored.

Slow positive, fast negative: An agent uses several positive experiences to build trust but can lose trust very easily with only a few negative trust experiences. For instance positive experiences will cause trust to increase by only one degree and negative experiences will cause a decrease in trust by only two degree.

Slow negative, fast positive: Trust is built based on only a few positive experiences but several negative experiences are required in order to lose trust. For instance positive experiences will cause trust to increase by two degrees and negative experiences will cause a decrease in trust by only one degree.

Balanced slow: Allowing for slow update of trust in both the positive and negative domains. Trust will be increased or decreased in small degrees after thorough evaluation.

Balanced fast: Allowing for fast update of trust in both the positive and negative domains. Any change in trust will drastically influence a trust level in both a negative and positive manner.

These six approaches can be grouped under three main trust outlook approaches: optimism, pessimism and pragmatism. An optimistic agent expects good outcomes while a pessimistic agent expects the worst. A pragmatic agent exists somewhere in the spectrum between a pessimistic agent and an optimistic agent balancing positive and negative experiences [1]. The blindly positive as well as the slow negative, fast positive approaches are optimistic while the blindly negative and slow positive, fast negative are pessimistic. The balanced approaches are pragmatic.

Optimistic agents may present a system with more opportunities for interactions but also open a system up to greater risk. Pessimistic agents may lower risk exposure, but it may risk closing the system off from opportunities when the reason for failure is only temporary. Pragmatic agents lie somewhere in-between. The particular trust outlook that is considered best depends on the system the model is implemented in. A system that does not need to look for new opportunities and where the repercussions of a mistake are rather large and critical, such as in Internet banking, may prefer to take a pessimistic approach to trust.

2) Passionate versus rational trust: Passionate agents are considered to have a free will and thus act in a very human-like manner. A passionate agent is expected to either be benevolent (an agent that behaves in an honest manner and follows the expected rules) or malicious (an agent that is deliberately dishonest and sets out to cause harm, hiding its intent to do so). Passionate agents develop trust according to intangible principles, or rather principles that have no specific binding mathematical rules attached to them. These principles include trust through collaboration, experience, reputation and recommendation and have varying mathematical definitions among various trust model implementations.

Rational agents are system-like agents. These agents are governed by a specific set of rules that tend to remain static. A rational agent lacks free will and is not expected to be benevolent or malicious. When trusting a rational agent it is expected that it will resist attempts of malicious manipulation from other agents [11]. The most common rational agent, is that of a firewall which grants and denies access according to fixed mathematical parameters that have been defined. Rational agents provide more control over interactions while passionate agents are given more freedom. Passionate agents are more adaptable than rational ones but also tend to expose a system to higher risk with higher potential gain. A passionate agent has a dynamic trust level for other agents. This level is increased and decreased over time and as a result of experiences. Unlike a firewall, a passionate agent will have varying trust levels over time. Depending on experiences and information gathered it grants access rights according to this dynamic trust level.

3) Centralized versus decentralized trust: Trust structures can either be centralized or decentralized [3]. Centralized structures allow a single centralized node to gather information from all other agents involved in an interaction. This node manages the interaction and ensures that all parties concerned abide by the same trust definitions. For instance, when defining trust using reputation, information is made globally visible, and defined by the entire system. Agents are allowed to influence one another's reputations by providing satisfaction feedback. A satisfaction feedback is feedback in which an agent indicates their level of satisfaction with a particular interaction. The feedback impacts other agents' reputation, which is stored in a single place and defines a new trust level for a particular agent.

Decentralized systems are a bit more involved in the trust establishment process than centralized nodes. Trust is subjective and established by each and every agent for other agents. Thus, agents' trust levels in others will vary from agent to agent dependent on each individual subjective context. Reputation is not global and the process of acquiring a reputation requires an agent to query other agents and combine the results received in order to obtain a global estimate. This form of establishing trust requires a high communication overhead.

In a centralized model, such as eBay, every agent has the same opinion as the central agent. EBay has a central agent that gathers all reputation based information. All agents looking for trust related information gather this information from this central source. Thus trust is global and every agent has the same trust value for another. In a decentralized approach, every agent has its own unique trust value based on the trust model in place [12]. Agents in a decentralized environment, such as ad-hoc environments [2] often do not have a stable central source of information. Thus a

decentralized approach is taken. Every agent determines a trust value for another according to their own rules and perceptions.

4) Trust versus distrust: Many trust models use a single value over a specific range in order to represent trust. This value is known as a trust value. For instance, values in the range between -1 and 1 are used to represent trust, 1 representing trust in another and -1 representing distrust.

This representation is simple and can be rather effective but has a very interesting problem. Does the -1 value represent distrust as a result of negative experiences or simply distrust based on lack of prior experience and knowledge [13]? This problem is further demonstrated by the uncertainty problem in ad-hoc environments. In environments with high mobility, such as ad-hoc networks, being uncertain about another node's trustworthiness is a common phenomenon [2], making it vital to distinguish between distrust and simple uncertainty. Modeling distrust as a separate parameter has been recommended to solve this problem.

A means of dealing with the uncertainty versus distrust problem has been proposed by Li, Lyu and Liu [2]. They recommend that trust representation is a 3-dimentional metric that includes values for trust, distrust as well as uncertainty. In order to successfully weigh trust against distrust, both positive and negative experiences are recorded. Positive experiences influence the trust parameter and negative values influence the distrust parameter. A trust value is ascertained as a combination of the two.

When working with values of distrust, it is important to keep in mind the fact that generally negative behaviour has more incentive to impact trust than positive behaviour. An interesting quote by Gambetta [14] summarizes this phenomenon rather nicely:

"While it is never that difficult to find evidence of untrustworthy behaviour, it is virtually impossible to prove its mirror image."

Once distrust has been established it is often difficult to regain trust [1]. Thus, when working with parameters that indicate distrust, a trust model needs a means of addressing this issue in order for the evaluation to be accurate.

5) Trust scalability: As in any implementation, an important factor to take into consideration is the scale on which a trust model is expected to work [15]. Most authors do not address the scalability of their models [4], [5], [12], [16], thus requiring a little deductive reasoning when analysing their models for scalability. There are a few basic factors that can be taken into consideration when considering the scalability of a trust model.

Scalability deals with processing, network and space constraints that a particular system has. Experience and historical information are a good means of keeping a dynamic accurate trust value, but require space in which to store this information. This need for space can be curbed by allowing a trust model to only keep historical and experience-based information for only a short limited time period before overwriting old stale information with new information. Another means of saving space is to require an agent to only store such information about particular agents it may be interested in instead of all agents it may have encountered.

Another consideration is the networking capabilities of a particular environment. Some trust models require more messages to be exchanged than others incurring a higher network load. The more complex a trust representation is, the more likely it is that a higher message load will be required in order to successfully establish a trust relationship.

Some trust model implementations require agents to keep a list of friends, trusted nodes, trusted recommenders and even information about other nodes. The degree to which storage of this information is required, as well as possible overhead among agents, needs to be taken into consideration. Though not likely to pose a problem in small systems, space constraints can become an issue in large systems. Model's such as Abdul-Rahman and Hailes' Trust-Reputation model [16], that are required to store vast quantities of data are not very scalable. This model requires the storage of experience as well as recommendation information by each agent about all other agents it has encountered in an environment as well as the contests in which this information was gathered.

Trust representation influences all the other categories of trust criteria defined by the author as it defines how information looks and flows. Only a brief description of the other three categories follows due to limited space constraints.

B. Initial trust

Initial trust refers to the strategy a trust model adopts in order to obtain an initial trust value for another agent in the environment. There are several strategies that can be chosen and used. These include but are not limited to recommendation, reputation, observation, institution, collaboration, negotiation and experience [1], [4], [5], [9], [12], [13], [14] . All of these require that the model gather some form of information in order to establish a trust value before participating in an interaction with another agent.

C. Updating trust

Due to the dynamic, ever-changing nature of trust, it is important to have a means of updating trust values. This allows an agent to change a trust value over time in order to keep up with changing environmental and situational factors. Agents are, thus, able to protect themselves from other agents that used to be trustworthy but became malicious over time. Often trust is updated using the same mechanisms that were used to define initial trust though there are a few additional factors that need to be taken into consideration. These are the influences of direct experience, feedback and transaction analysis.

D. Evaluation of trust

In order to successfully update trust, a trust model needs a means of evaluating the information it has gathered. Although several trust models use mechanisms that are closely linked to the means in which trust is initially established and updated in order to evaluate their trust values, there are additional strategies that are in use in order to evaluate trust and limit interactions. These include the use of roles and categorization. When evaluating a particular interaction, there are certain factors that need to be taken into consideration if an accurate analysis is to be obtained. These include the context in which an interaction occurred, the possible reasons for failure, the risk that was involved and whether a dynamic or approximate computation is desired.

These four broad criteria categories have been proposed in order to ease analysis of currently proposed trust models. The next section demonstrates how this analysis process works for trust representation. An existing trust model is chosen and analysed using various points within the trust representation criteria category.

IV. Example Analysis of Trust Representation

The trust model chosen in order to demonstrate a sample analysis using trust representation is Abdul-Rahman and Hailes' Trust-Reputation model [16]. This model was chosen due to its relative simplicity and inclusion of many of the concepts discussed. This model determines the trustworthiness of other agents based on collected statistics that include direct experiences and recommendations.

A set of direct experiences and recommendations is summarized in order to obtain a trust value. Experiences are recorded into two separate sets in order to be able to differentiate those that are a result of direct trust and those that are linked to recommendation. This way an agent is able to keep track of the direct trust it has in others as well as the trust level it has in recommendations from other agents.

In order to obtain a trust value for a particular agent in a particular context, this model relies on a basic system of counters. For every agent within a specific context, an agent possesses four counters. These counters are for varying trust degrees. The four counters this model makes use of are counters or very good, good, bad and very bad. Hence, these counters represent various trust levels namely very trustworthy, trustworthy, untrustworthy and very untrustworthy respectively. These counters are incremented with direct experiences an agent has as well as with recommendations an agent receives. The trust model runs a max function on these four counters that return the counter that has the largest value for the specific agent in a specific context. The trust counter with the largest value indicates the trust level that is to be assigned.

It is important to note that this model incorporates the concept of varying perception. Because trust is a subjective concept, it is logical to assume that perception of trust will differ among different agents. What one agent may define as trustworthy could be seen as untrustworthy by another agent based simply on differing perceptions.

This model allows for an agent to adjust recommendation values it receives from other agents in order to include the difference in perception. In order to accomplish this, an agent obtains an adjustment value. In order to obtain this adjustment value, an agent compares its own trust evaluation result for a particular agent in a particular context with the trust evaluation result of the recommender agent for the same agent in the same context. If the trust evaluation result differs, the agent creates an adjustment value for recommendations coming from that particular recommender.

For example, agent A is looking for an adjustment value for recommender B. Agent A knows that the trust value it has for agent D is 't' (trustworthy) in context C. A receives information from B that B's trust value for D in context C is 'vt' (very trustworthy). In other words, A's value for agent D in the same context as B's value for the same agent is one level lower than B's. The adjustment value that A obtains will be -1. This value is then used to lower the trust value of all recommendations coming from agent B by 1 in order to more closely represent A's own perceptions.

This trust model is analysed using the concepts discussed under the trust representation section as follows.

Trust Outlook: The agents in this trust model have a rather pragmatic approach keeping a record of both positive and negative experiences in order to balance them in given contexts. Agents possess four experience counters: very good, good, bad and very bad that are incremented with direct experiences and recommendation information. Good experiences increment the good counter; very good experiences increment the very good counter and so forth. When determining a trust value a max function returns the highest count. Trust is assigned according to which counter has the highest count. If the max function returns more than one of the counters, an uncertainty value is assigned. *Passionate versus Rational:* This model takes on a passionate approach, incorporating a mechanism of social control that clusters towards results that are positively reputable. The social concept this model builds on is that of reputation and recommendation, relying on the 'opinions' of other agents to make trust based decisions.

Centralized versus Decentralized: This model is decentralized. It is proposed for open distributed systems. There is no central agent that gathers all the information. Each agent has its own evaluations for others and is even allowed to adjust trust values it has received as recommendations.

Trust versus Distrust: While there are no explicit variable defined in order to handle distrust, an agent can differentiate between lack of trust due to lack of knowledge and lack of trust due to experiences and information it has gathered. An agent keeps an experience set for each agent it has interacted with and the context it has interacted in. This set has a counter value for very good, good, bad and very bad experiences. These values are incremented with experiences gained.

Scalability: The model keeps the format of the information that an agent needs to store rather simple. Although, it does not require a lot of space to store information about a single agent in a single context, this model still has scalability issues due to space constraints. The model does not limit the number of agents, or the number of contexts or even the period of time for which experience information is gathered and stored. Should these issues remain unaddressed; this model will encounter scalability issues due to the sheer number of records the agent is expected to keep seeing as agents store trust information in agent context pairs.

V. Discussion

This paper addresses the problem of analysing various trust models in order to determine the efficiency of a trust model and pinpoint issues that have not been addressed. Four main categories of criteria have been identified by the author. These categories define the main features any good trust model is required to have. The main categories identified are trust representation, initial trust, updating trust and trust evaluation.

Due to space constraints, only one of the four criteria categories is discussed in detail. The chosen category is that

of trust representation. Trust evaluation is covered in another paper submitted by the author for review [17]. This particular category was chosen because of its importance. Trust representation influences later the means in which initial trust is established, trust is updated and trust is evaluated. The latter three have not been addressed in this paper in detail. Factors influencing the means in which trust is represented that have been identified and discussed in this paper include trust outlook, passionate versus rational trust, centralized versus decentralized trust, weighing trust against distrust and the scalability of a model.

VI. CONCLUSION

This paper introduced and discussed the concept of a set of criteria that is to be used for the evaluation of trust models. These criteria are intended to be a guideline to trust model evaluation in order to identify the efficiency of a particular trust model. In the same way that they can be used to evaluate a currently implemented trust model, they can also be used as a guide for factors and issues to take into consideration for future trust model implementations.

Using these criteria, one is able to identify how a trust model addresses certain issues and also which issues have not been addressed. An example of such an issue is the scalability issue that was not successfully addressed by Abdul-Rahman and Hailes' model above. This is important knowledge to have when considering choosing a particular trust model for implementation and can also be used to guide future improvements on trust model architectures.

Abdul-Rahman and Haile's Trust-Reputation model was taken as an example and analysed using the factors identified within the trust representation criteria category. It is important to realize that this criteria are not necessarily all the possible defined criteria that can be taken into consideration. They are based on 'known' issues and known implementations and it is possible to expand them in future work. Future work also includes the development of a measurement mechanism in order to score the degree to which a trust model addresses the issues identified.

REFERENCES

[1] Marsh, S.P. Formalising Trust as a Computational Concept. In Dissertation for the Department of Computing Science and Mathematics, University of Stirling. 1994.

[2] Li, X. Lyu, M.R. & Liu, J.. A Trust Model Based Routing Protocol for Secure Ad Hoc Networks. In IEEE Aerospace Conference Proceedings.. 2004. ISSN: 0-7803-8155-6/041.

[3] Wang, Y. & Vassileva, J. Bayesian Network-Based Trust Model. In Proceedings of Web Intelligence, 2004. WI 2004. IEEE/WIC?ACM International Conference on 20-24 Sept 2004. pp 341-348. ISSN: 0-7695-2100-2.

[4] Azzedin, F. & Maheswaran, M. Trust Modeling for Peer-to-Peer Based Computing Systems. In Proceedings of the International Parallel and Distributed Processing Symposium. 2003.

[5] Huynh, T.D., Jennings, N.R. & Shadbolt, N.R. FIRE: An Integrated Trust and Reputation Model for Open Multi-Agent Systems. In Proceedings of 16th European Conference on Artificial Intelligence, pp. 18-22, Valencia, Spain. 2004.

[6] Nooteboom, B., (2002) Trust: Froms, Foundations, Functions, Failures, and Figures, Edward Elgar Publishing, Ltd. Cheltenham UK, Edward Elgar Publishing, Inc. Massachusettes, USA, ISBN: 1 84064 545 8.

[7] Golembiewski, Robert T., & McConkie, Mark. 1975. The Centrality of Interpersonal Trust in Group Processes. Chap. 7, pages 131–185 of: Cooper, Cary L. (ed), Theories of Group Processes. Wiley.

[8] Pirzada, A.A. & McDonald, C. Establishing Trust in Pure Ad-hoc Networks. In Research and Practice in Information Technology. Vol. 26 V. Estivill-Castro, Ed. 2004.

[9] Abdul Jonker C.M. & Treur J. 1999. Formal Analysis of Models for the Dynamics of Trust based on Experiences. In Modelling Autonomous Agents in a Multi-Agent World, pp 221-231. 1999.

[10] Li, X., Valacich, J.S. & Hess, T.J. Predicting User Trust in Information Systems: A Comparison of Competing Trust Models. In Proceedings of the 37th Hawaii International Conference on Systems Sciences. 2004.

[11] Josang, A. The right type of trust for distributed systems. In C. Meadows, editor, Proc. of the 1996 New Security Paradigms Workshop. ACM, 1996.

[12] Liang, Z. & Weisong, S. PET: A PErsonalized Trust Model with Reputation and Risk Evaluation for P2P Resource Sharing. In Proceedings of the 38th Hawaii International Conference on Systems Sciences. 2005.

[13] Guha, R., Kumar, R., Raghaven, P. & Tomkins, A. Propagation of Trust and Distrust. In Proceedings of the Thirteenth International World Wide Web Conference, 2004.

[14] Gambetta, Diego. 1990. Can we Trust Trust? Chap. 13, pages 213–237 of: Gambetta, Diego (ed), Trust. Blackwell.

[15] Luo, H., Zerfos, P., Kong, J., Lu, S. & Zhang, L. Self-securing Ad Hoc Wireless Networks. In: Seventh IEEE Symposium on Computers and Communications (ISCC). 20021.

[16] Abdul-Rahman, A. & Hailes, S. Relying on Trust To Find Reliable Information. In Proceedings of DWACOS'99, Baden-Baden, Germany., 1999.

[17] Wojcik, M., Venter, H.S. & Eloff, J.H.P Trust Model Evaluation Criteria: A Detailed Analysis of Trust Evaluation. Submitted for review for ISSA 2006.

Appendix G

Trust model architecture: Defining prejudice by learning (Wojcik, Eloff & Venter 2006).

In: *Proceedings of the 3rd International Conference on Trust, Privacy & Security in Digital Business (TrustBus)*, Krakow, Poland, 1-10.

In: *Lecture Notes in Computer Science: Trust, Privacy, and Security in Digital Business*, London UK: Springer-Verlag, 4083:182-191.

Trust Model Architecture: Defining Prejudice by Learning

M Wojcik[1]
JHP Eloff[2], HS Venter[2]

Information and Computer Security Architectures Research Group (ICSA)
Department of Computer Science, University of Pretoria
[1]{hibiki}@tuks.co.za
[2]{eloff, hventer}@cs.up.ac.za

Abstract. Due to technological change, businesses have become information driven, wanting to use information in order to improve business function. This perspective change has flooded the economy with information and left businesses with the problem of finding information that is accurate, relevant and trustworthy. Further risk exists when a business is required to share information in order to gain new information. Trust models allow technology to assist by allowing agents to make trust decisions about other agents without direct human intervention. Information is only shared and trusted if the other agent is trusted. To prevent a trust model from having to analyse every interaction it comes across – thereby potentially flooding the network with communications and taking up processing power – prejudice filters filter out unwanted communications before such analysis is required. This paper, through literary study, explores how this is achieved and how various prejudice filters can be implemented in conjunction with one another.

1 Introduction

Technological development has influenced the principles required to run a successful economy [1]. However, the advent of new technologies and the subsequent implementations thereof have resulted in exposure to new risks. Two risk factors exist that continually drive research towards lessening the risks encountered: effective communication and security.

In order to accomplish an organisation's desired task, effective and timely communication is required. An organisation makes use of technology to communicate and share information. This information is an asset to the organisation and is used to assist decision-making processes. It is important that this information be reliable and accurate so that it can be trusted [2].

Trust models have been proposed in order to minimise the risk of sharing and successfully analysing information [3], [4]. Trust models rely on the abstract principle of trust in order to control what information is shared and with whom. Trust models evaluate the participants of a transaction and assign a numerical value, known as a trust value, to the interaction. This numerical value is used to determine the restrictions placed on the transaction and the nature of information shared. This

process of analysis occurs with all interactions an agent running a trust model encounters and can lead to an overwhelming communication load. In order to control the number of interactions a trust model encounters, prejudice filters have been proposed.

This paper introduces and defines the concepts of prejudice, trust and trust models in Section 2 by introducing a basic trust management architecture and expanding on work already done in these areas. The concept of prejudice filters and their interdependencies is explored in Section 3, with special focus on one relationship involving the learning filter. This is followed by a discussion of concepts in Section 4 and a conclusion in Section 5.

2 Background

Since trust model architecture is based on the concept of trust, a basic understanding of trust is required. This section introduces the concept of trust in the context of human relationships and then explores how this concept is put into practice by trust model architecture. The concept of prejudice is also explored, with special attention to how this concept can lighten communication load required to make trust-based decisions.

2.1 Trust Models and Trust

Trust models rely on the concept of agents [4]. Within the context of trust models, an agent refers to a non-human-coded entity used to form and participate in machine-based trust relationships. This agent would usually be situated on a computer and implement some form of logical rules to analyse the interactions with which it comes into contact in order to determine whether another agent is to be trusted or not. These logical rules may be static or adjustable by the agent in a dynamic manner, based on results of transactions the agent has participated in.

Trust is a subjective concept – the perception of which is unique to each individual. Trust is based on experience and cognitive templates. Cognitive templates are templates formed by experiences that are later used to analyse future experiences of a similar nature. Trust is dynamic in nature and influenced by environment, state and situation. According to Nooteboom [5], "*[s]omeone* has trust in *something*, in some *respect* and under some *conditions*".

Each of the four key concepts highlighted by Nooteboom exists within trust model architecture. *Someone* and *something* define two agents participating in an interaction. The former refers to the instigator of the interaction whiles the latter refers to the agent accepting the request. The *respect* is defined by the reason for instigating an interaction. Finally, the *conditions* refer to the situational factors that influence the success of an interaction.

2.2 Trust Model Architecture

Trust models assist agents that have not previously encountered one another by forming and participating in trust-based interactions. Various experts have already proposed numerous trust models [6], [7], [8]. A survey of the literature conducted by the author has identified four components that have been used in trust model implementation: trust representation, initial trust, trust dynamics and trust evaluation.

Catholijn M. Jonker and Jan Treur [9] focus on how trust is represented by agents in order to simulate intelligence and make trust-based decisions. They propose a simple qualitative method of representing trust that defines four basic trust values. These values include unconditional distrust, conditional distrust, conditional trust and unconditional trust. Other issues of trust representation, as identified by Damiani, De Capitani di Vimercati and Samarati [10], include protocols that are required in order to communicate and discern trust related information. These protocols are required to identify and analyse trust related information in anonymous environments as well as to control what identity information is released under specific circumstances.

Jonker and Treur in further research state that trust models incorporate trust characteristics that can be divided into two states. These states refer to initial trust – the initial trust state of an agent – or trust dynamics – the mechanisms that allow for the change in and updating of trust [9]. The initial trust state of an agent determines the agent's predisposition wherein the agent can be predisposed towards trust, distrust or neutrality. Taking the dynamic nature of trust in consideration, Marx and Treur [8] concentrate on a continuous process of updating trust over time. Experiences are evaluated and used by a trust evolution function.

Changing trust values requires that some form of trust evaluation should take place. The reputation-based model of Li Xiong and Ling Liu [11], known as PeerTrust, emphasises the importance of this evaluation process by evaluating various parameters, such as nature of information shared and purpose of interaction, in order to update the trust value an agent retains.

Trust models are able to obtain trust values in several manners. Trust information and state can be pre-programmed into the agent as a list of parameters. These parameters can also be dynamically formulated, based on pre-defined and logically formed trust rules that an agent uses to evaluate trust.

2.3 Example of a Typical Trust Architecture

According to Ramchurn *et al*. [12] basic interactions among agents go through three main phases. These phases are negotiation, execution and outcome evaluation. Trust plays an essential part in all three of these phases. This is illustrated by Figure 1.

Two agents attempting to communicate with one another are first required to establish a communication link, usually initiated by one agent and accepted by another. This process initiates a negotiation process whereby two agents negotiate various parameters, such as the security level of information that is to be shared or the services for which permission will be granted, that will define boundaries of the interaction. A trust value for the interaction is defined through comprehensive analysis of logical rules. The simplest way of storing and implementing these rules is

to have them present in a list that the agent accesses and processes. In Figure 1, storage of these rules occurs in the trust definition list.

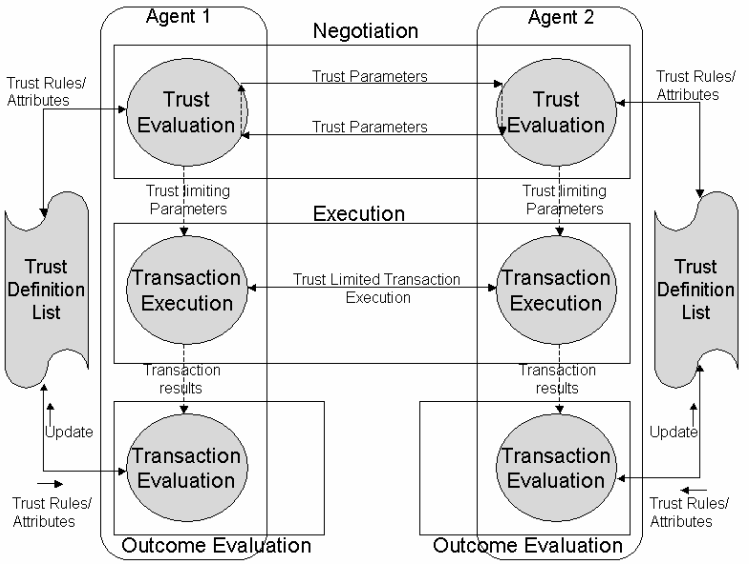

Figure 1: Operation of an agent using a trust model

The successful negotiation and establishment of a trust value triggers an analysis of the trust value. Provided the trust value is above a certain acceptable threshold, the transaction execution process is started. Trust models control the context of the interaction during the execution phase, limiting trust given and hence controlling which information or services are accessible and which are not.

Once transaction execution has terminated, the results of the interaction are sent to the transaction evaluation process. This process evaluates the results and updates the trust definition list in either a positive or negative manner. Negative updating of the logical rules occurs due to business transaction failure, while business transaction success will trigger a positive update.

The evaluation of trust among agents is a time-consuming process that requires comprehensive evaluation of the defined logical rules in order to attain an accurate trust value to be used during an interaction. Only once the trust value has been obtained, the agents will decide whether to participate in a transaction or not.

In a networking environment, the amount of possible agents that will request participation in such an interaction can be vast. To successfully assess another agent, agents pass several messages to obtain the required information that is to be analysed against the defined trust parameters. For instance, the formal model for trust in dynamic networks proposed by Carbone, Nielson and Sassone [7] passes delegation information between agents in order to create a global trust scheme. Delegation allows a particular agent to trust another agent, based on the fact that the other agent is trusted by agents that the agent in question trusts. This reliance on the passing of messages exposes the network to the possibility of network overload. Another

potential problem arising during the process of establishing trust is the level of comprehensiveness required by the analysis process. Having a large number of strict rules define a trust relationship limits the communications an agent will be able to participate in, while at the same time adding to the analysis load. Rules that are too generic open the system up to a higher level of risk by allowing an agent to participate in interactions with other agents that have not been fully analysed for trustworthiness.

Prejudice filters have been proposed to lessen the number of interactions that require comprehensive trust evaluation [13] so as to solve the problems mentioned above. Stereotyped grouping of interactions allows for characteristics to be assumed instead of evaluated in detail. It also allows trust evaluation to focus on characteristics that are not assumed, instead of evaluating the interaction against the entire list of logical rules that represent expectations.

3 Prejudice Filters

In order to understand the concept of prejudice filters, an understanding of prejudice is required. Prejudice is an extension of the concept of trust-building processes and is defined as a negative attitude towards an entity, based on stereotype. It is important to note that the negative nature of prejudice as prejudice allows negative assumptions in order to evaluate trust. Prejudice influences trust by allowing certain negative assumptions to be made about certain groups. These negative assumptions are based on prior knowledge and experience with such groups. All entities of a certain stereotyped group are placed in the same category, allowing assumptions to be made and simplifying the processing required before trust can be established [14]. This way an agent only needs to analyse attributes it does not have assumptions about in order to adjust trust value. An agent is allowed to completely distrust an agent simply because it falls into a category which it perceives as negative.

Agents see prejudice filters as simplified trust rules that rely on the concept of prejudice in order to limit the number of interactions an agent needs to analyse in detail. Prejudice filters rely on broad definitions of attributes that lead to distrusted interactions, thus denying interactions that can be defined by these attributes. For example, if an agent has interacted with another agent from a specific organisation and the interaction failed in terms of expectations, future requests from agents belonging to the same organisation will be discriminated against. Figure 2 illustrates where prejudice filters extend the trust architecture as originally depicted in Figure 1.

Prejudice filters affect two phases of the three-phase interaction cycle: the negotiation and outcome evaluation phases. In the negotiation phase, the prejudice filters are consulted first to provide a quick, simplistic evaluation of trust in order to filter unwanted communications before they are required to go through detailed trust evaluation and definition. Once an interaction has passed the prejudice evaluation, it moves onto the trust evaluation in order to acquire a trust value. When the execution phase concludes, the outcome evaluation phase includes the prejudice parameters when it evaluates the interaction. Failed transactions update the prejudice filters in order to filter out other transactions of a similar nature at an earlier stage.

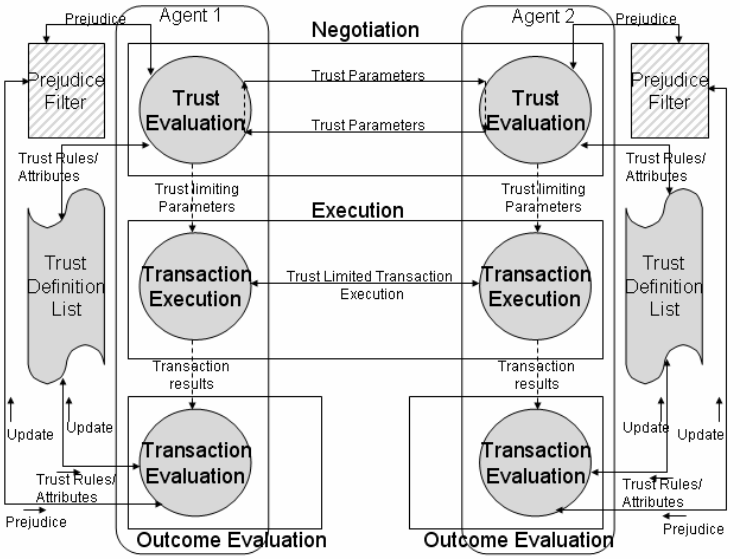

Figure 2: Operation of an agent using a trust model with prejudice filters

3.1 Extending Existing Models to Include Prejudice Filters

Existing trust models rely on various means of establishing trust, which include recommendation, reputation, third party reference, observation, propagation, collaboration, negotiation and experience [1], [2], [4], [6], [7], [8], [9], [10], [11], [12]. Based on these, five means of implementing prejudice filters have been identified by the author in order to simplify the extension of existing models to include prejudice. These five are as follows [13]:

Learning: When using the learning filter, prejudice is not defined explicitly. An agent relies on 'first impressions' to learn prejudice. If an interaction fails, the agent analyses the interaction's attributes and looks for unique attributes of other interactions that were previously encountered and found to be successful. These unique attributes are used to create a category to be used as a prejudice filter.

Categorisation: An agent creates various categories that are trusted. If an interaction request does not fall into a trusted category, the agent filters out that interaction in a prejudiced manner. This can also be implemented in a reverse manner where an agent creates categories that are distrusted and filters out communications that fall into those categories. Categories can also be created to represent various levels of trust. Any interactions falling into such categories are assigned the default trust value associated with that particular category.

Policy: Policies define the operational environment in which an agent exists and affect parameters of interactions that are regarded acceptable. Policy-based prejudice filters out interactions with agents whose policies differ from the agent doing the filtering. One way of doing this is to request data on the country an agent resides in. Such data defines the laws and culture that bind business interactions for that agent, as well as controls the means in which data and confidentiality are is handled.

Path: Path-related prejudice allows an agent to refuse an interaction, simply because of the fact that the path of communication between two agents passes through a distrusted agent.

Recommendation: Agents that are trusted to make recommendations are known as recommender agents. Implementing prejudice by using recommendation allows a particular agent to only trust other agents that are trusted by the particular agent's recommender agents.

The above five filters can be incorporated into current trust models to extend their capability, while at the same time allowing for these filters to merge with a particular trust model's main philosophy. Just as some models use a combination of concepts to implement the concept of trust, interrelated filters can be implemented in different combinations in order to optimise their effectiveness.

3.2 Defining Interrelationships between Filters

The five prejudice filters discussed above can be organised into a structure of relationships as shown in Figure 3. This structure depicts relationships that exist between these filters. The root node of a relationship between two prejudice filters indicates the dominant filter. The second filter can be incorporated into the workings of the dominant filter when the two are implemented together. The directional arrows in Figure 3 illustrate this. The dominant filter is situated at the tail of the directional arrows. Two prejudice filters emerge as more dominant than the others: learning and policy. These prejudice filters are always situated at the tail end of the arrows in Figure 3 and can be implemented in conjunction with all the other lesser filters.

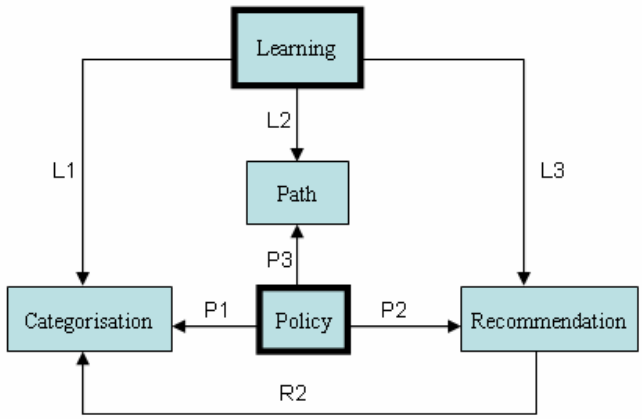

Figure 3: Overview of the inter-relationships between prejudice filters

Due to space constraints, only one of the illustrated relationships is explored, leaving the rest for further discussion in future work. The relationship discussed has the learning filter as its root node and is labelled L1 – linking the learning and categorisation prejudice filters.

Learning-Dominated Relationships. The nature and success of learning is governed by the nature and variety of information and experience that an agent is exposed to [15]. Experiences and information are filtered to form templates unique to each agent. Templates are default rules that have been formed by experiences and that are subsequently used to evaluate other similar experiences.

When using learning, prejudice is not defined explicitly, and an agent relies on 'first impressions' to learn prejudice. Possible implementation of this concept allows an agent to begin with a basic set of rules that it uses to evaluate the success of an interaction. Initially, the agent will interact with any agent with which it comes into contact, under restricted conditions of trust. Each interaction instigates an analysis process by means of which the agent will identify parameters such as location of an agent, security of information required, and even factors such as an agent's reputation. These parameters become the characteristics of the particular interaction and should the interaction fail, they will be analysed in order to identify a means of filtering out future interactions of a similar nature.

Due to the fact that learning creates various forms of templates [16], learning various forms of prejudice can be accomplished. One of these is discussed below.

Learning by Categorisation (L1). Categorisation is an umbrella term that allows for objects or concepts with similar attributes to be grouped together. This allows for certain assumptions to be made in order to simplify analysis of such objects. The attributes that can be assumed are those that define a certain object or concept as belonging to a specific category. For instance, agents that belong to the same policy category are assumed to hold similar policy values, such as information privacy constraints. Only agents from acceptable categories will be sent for trust evaluation by an agent wishing to interact with another. Agents that are defined as unacceptable at the onset of the interaction are discarded before entering the comprehensive trust evaluation phase. This eases the processing load by filtering out undesirable categories before sending the interaction to the trust evaluation process which determines a trust value.

The process of learning prejudice relies heavily on categorisation. Learning analyses a transaction to determine its unique features. If the transaction fails, the agent uses this analysis process to create a category of failure to be used in future category-based prejudice decisions. Implementation of this concept relies on allowing an agent to form categories defined by the trust rules in place. For instance, if the trust rules in place require transactions to be analysed in order to determine the policies used by the agents in question, these agents can be categorised by their policies and characteristics. Agents can be categorised by their core services, products and policies [17].

An agent is required to either keep a list of categories that are trusted or categories that are not trusted. Whenever a new interaction is encountered, the interaction is analysed against the characteristics of the various categories in order to define the category the interaction belongs to. Once the category has been defined, the agent checks its list of trusted or distrusted categories in order to determine whether interactions of that nature are trusted. If the interaction type exists in the distrusted categories list or alternately does not exist in the trusted list, the interaction is seen as distrusted and is then discarded. Unknown or undefined categories are by default considered to be distrusted.

Categorisation can also be used to define different levels of trust. This is accomplished by assigning a default trust value associated with a category to agents that fall into that category. The rights delegated to an interaction are consequently limited by the category to which it belongs [6]. An example of such a category is role. Various roles are given differing rights. An administrative role is given more access rights than a client role.

4 Discussion

The concept of implementing prejudice as discussed in this paper is a very new concept that still requires further experimentation and analysis. One of the shortcomings of these filters is related to the fact that they allow machines to deny access due to the values of prejudice that were obtained.

This can lead to a situation in which agents that are in actual fact trustworthy are seen as untrustworthy, simply because of the prejudice filter in place. A situation like this, however, can be controlled by allowing agents to interact with several agents with similar defined characteristics before deciding prejudice against them. Increasing the number of interactions in which an agent participates increases the risk an agent is exposed to. Thus, there is a trade-off between accuracy of prejudice prediction, and the risk an agent is willing to take.

5 Conclusion

This paper has introduced the concept of trust models and prejudice. Different means of incorporating prejudice include categories, policies, path, recommendation and learning. Several of these filters are related in such a manner that they may be implemented in conjunction with one another. One of these relationships, namely that between learning and categorisation, has been explored and defined by this paper.

The authors have explored this topic from a conceptual standing that requires implementation and testing. Since only one relationship was scrutinised in this paper, further work requires more detailed investigation of the other defined existing relationships. More in-depth work needs to be done on means to standardise the representation of trust-related data, thus allowing agents from various platforms and using various models to efficiently interact with one another.

References

1. Hultkrantz, O., Lumsden, K., E-commerce and consequences for the logistics industry. In: Proceedings for Seminar on “The Impact of E-Commerce on Transport.” Paris (2001)
2. Patton, M.A., Josang, A., Technologies for trust in electronic commerce. In Electronic Commerce Research, Vol 4. (2004) 9-21

3. Abdul-Rahman, A., Hailes, S., A distributed trust model: new security paradigms workshop. In Proceedings of the 1997 workshop on new security paradigms, Langdale, Cumbria, United Kingdom, (1998) 48-60
4. Ramchurn S.R., Sierra, C., Jennings, N.R., Godo, L., A Computational Trust Model for Multi-Agent Interactions based on Confidence and Reputation. In: Proceedings of 6th International Workshop of Deception, Fraud and Trust in Agent Societies, Melbourne, Australia, (2003) 69-75
5. Nooteboom, B., Trust: forms, foundations, functions, failures, and figures. Edward Elgar Publishing, Ltd., Cheltenham UK. Edward Elgar Publishing, Inc. Massachusettes, USA. ISBN: 1 84064 545 8 (2002)
6. Papadopou, P., Andreou, A., Kanellis, P., Martakos, D., Trust and relationship building in electronic commerce. In: Internet Research: Electronic Networking Applications and Policy, Vol 11. No. 4 (2001) 322-332
7. Carbone, M., Nielsen, M., & Sassone, V., A formal model for trust in dynamic networks. In: Software Engineering and Formal Methods. In: Proceedings of the First International Conference on 25-26 Sept. (2003) 54-61
8. Marx, M., Treur, J., Trust dynamics formalised in temporal logic. In: Proceedings of the Third International Conference on Cognitive Science, ICCS (2001) 359-362
9. Jonker, C.M., Treur, J., Formal Analysis of Models for the Dynamics of Trust based on Experiences. In: Proceedings of MAAMAW'99. LNAI (1999).
10. Damiani, E., De Capitani di Vimercati, S., Samarati, P., Managing Multiple and Dependable Identities. International World Wide Web Conference. In: Proceedings of the 13th international conference on World Wide Web . New York, NY, USA. (2003) 403-412
11. Xiong L., Lui L., A Reputation-Based Trust Model for Peer-to-Peer eCommerce Communities. E-Commerce, IEEE International Conference on 24-27 June (2003) 275-284
12. S.R., Sierra, C., Jennings, N.R., Godo, L., Devising a trust model for multi-agent interactions using confidence and reputation. In: Applied Artificial Intelligence. , Vol. 18., (2004) 833-852
13. Wojcik, M., Venter, H.S., Eloff, J.H.P., Olivier, M.S., Incorporating prejudice into trust models to reduce network overload. In: Proceedings of South African Telecommunications and Networking Application Conference (SATNAC 2005). SATNAC, Telkom, CD ROM Publication. (2005)
14. Bagley, C., Verma, G., Mallick, K., Young, L., Personality, self-esteem and prejudice. Saxon House. , Teakfield Ltd, Westmead. Farnborough, Hants. England. ISBN: 0 566 00265 5 (1979)
15. Bowling, M., Manuela, V., Multiagent learning using variable rate. In: Artificial Intelligence. Vol. 136 (2002) 215-250
16. Dasgupta, D., Artificial neural networks and artificial immune systems: similarities and differences. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC '97), Orlando, October 12-15 (1997)
17. Siyal, M.Y., Barkat, B., A novel Trust Service Provider for the Internet based commerce applications. In Internet research: electronic networking applications and policy, Vol. 12(1) (2002) 55-65