Introduction

1.1 Introduction

The concept of Mobile Computing System is ingrained in the concepts of distributed systems. Distributed systems [7] may be viewed as a collection of independent computers over a network that appears to be a single coherent system to its users.

With time, technological progresses have made distributed systems go mobile giving birth to mobile computing system. Machine hardware became transportable and then truly portable. Communication methods proliferated and became ubiquitous. Mobile computing systems can be regarded as an extreme case of traditional distributed system. As is stated in [8], “In a sense, mobile computing is the worst case of distributed computing since fundamental assumptions about connectivity, immobility and scale are no longer valid”.

The mobile environment brings challenges like

- Resources - Mobile elements are not rich in resources, unlike their static counterparts, though some difficulties are already masked with the recent advances in mobile telephony like GPRS, UMTS etc. As is predicted in [9], “For a given cost and level of technology, considerations of weight, power, size and ergonomics will exact a penalty in computational resources ... While mobile elements will improve in absolute ability, they will always be resource-poor relative to static elements.”

- Vulnerability - Portable computers are more vulnerable to intrusion, loss or damage [9].

- Connectivity - The requirement of wireless connectivity is one of the characteristics of mobile computing systems. A mobile client may have to rely on a low-bandwidth network with gaps in coverage while outdoor, though indoors may offer reliable, high-bandwidth wireless connectivity. Thus mobile connectivity is inherently unreliable.

- Energy - The importance that energy plays in the context of mobile computing systems with resource constrained mobile devices is evident from [9], “While battery technology will undoubtedly improve over time, the need to be sensitive to
power consumption will not diminish. Concern for power consumption must span many levels of hardware and software to be fully effective."

These constraints call for the following design issues [10].

- **Wireless Communication** - Mobile devices connected with sufficient bandwidth can operate as traditional distributed systems. But when communication medium provides high latency and low bandwidth, “synchronous” interactions in client/server models [9] get highly affected [11]. Networks that generally follow client-server models may be referred to as infrastructured network while the ones not having the server infrastructure may be called ad hoc network. An ad hoc network typically refers to any set of devices forming a network where all devices have equal role with respect to communication and computation. Such networks can be viewed as a special case where connectivity horizons are restricted.

- **Mobility** - Mobility of nodes allows nodes to come in contact with heterogeneous networks. Changing network access point calls for address migration as this address reflects the location of the node [12]. Thus mobility crosses administrative and security domains. Mobility if utilized carefully can be used to reduce uncertainty [13].

- **Portability** - Power conservation is a basic concern in the design and operation of hardware for supporting mobile computing. Moreover, making computers and devices portable, heightens their risk of unauthorized access whereas small storage and memory often lead to tradeoffs among memory, computation and energy consumption (as lack of caching increases computation time keeping the processor busy for longer periods).

Thus distributed systems built on top of mobile wireless networks should address the issues of the underlying environment.

### 1.2 Traditional Message Passing System vs Mobile Agents

Most of the implementation of mobile computing system is based on message passing. Since it is a distributed system no global observer or common clock can be present. Thus communication and synchronization happen by passing messages. A mobile computing system can be built on top of mobile infrastructured network or mobile ad hoc network. If infrastructured network is used, the system consists of both Mobile Hosts (MH) and static Mobile Support Stations (MSS). A base station (as in telephony) [11] or an access
1. Introduction

Figure 1.1: Infrastructured wireless network with node mobility

point (as in wireless LAN according to IEEE 802.11 standard [14]) can serve as MSS (Figure 1.1). A set of dynamic and wireless communication links can be established between an MH and an MSS, and a set of high-speed communication link is assumed between the MSSs. An MSS may communicate with a number of MHs but an MH at a time communicates with only one MSS. An MH communicates with the rest of the system via the MSS it is connected to.

If built on top of ad hoc network then such a system would consist of Mobile Hosts/Nodes only. The mobile nodes (MN) are connected to each other by wireless links. The wireless medium is a shared broadcast medium where transmissions made by a node are received by its neighbors, that is, the nodes within direct transmission range of the sender. Consequently, when a node receives data, no other neighboring node should transmit (except the sender). For communication with nodes beyond direct transmission range, each node needs to rely on its neighbors for forwarding its traffic. It is to be mentioned here that traffic forwarding can be expensive in terms of power consumption and/or memory usage. Due to mobility, network topology may change rapidly and unpredictably over time. The network being decentralized, all network activity including discovering the topology and delivering messages must be executed by the nodes themselves, that is routing functionality must be incorporated into MNs themselves. In contrast with other network types that depend on fixed infrastructure assets, the Mobile Ad hoc NETwork (MANET) gains its advantage from two main aspects:

- its absence of infrastructure and
- its self-forming nature
In other words, MANET can be deployed without the infrastructure that is the foundation of traditional networks; routers, switches, wiring, base-stations, and transceivers [15]. Thus, in a MANET, it is the nodes themselves and the links they form that define the network’s configuration. The creation and termination of links depend on the position of a node relative to the other nodes in the network resulting in a dynamically changing network configuration. The volatility (frequency and significance of change) of the network configuration is a result of the mobility of its nodes.

Message passing paradigm is the most fundamental paradigm in distributed system and provides satisfactory performance for static distributed applications. A process sends a request in the form of a message that gets processed at the receiver and often results in another message containing the response. The response may also trigger a further request. Thus it directly follows client-server model. But due to low bandwidth, limited power consumption, low memory along with frequent disconnection, message passing paradigm may not be that much suitable for mobile computing system particularly when a distributed computation involves a number of mobile nodes.

In MANET, nodes access and exchange information regardless of their geographic position (location independence) or proximity to infrastructure. This enables development of ubiquitous applications to facilitate the delivery of services that are sensitive and responsive to people [16]. Due to the inherent distributed nature, the agent paradigm offers an attractive metaphor for dealing with the complexity of such applications.

A mobile agent is an autonomous program that can migrate under its own control from one node to another in a heterogenous network [17]. It can be viewed as a distributed abstraction layer that provides the concepts and mechanisms for mobility and communication [18]. An agent consists of three components:

• the program which implements it,
• the execution state of the program and
• the data.

A node can act as a host for a mobile agent (Figure 1.2) if it provides a host platform (shown as MA in Figure 1.2) for execution of the agent’s code. A mobile agent may migrate from one host to another in two ways, namely, weak migration and strong migration. Weak migration occurs when only the code of the agent migrates to its destination, a strong migration occurs when the mobile agent carries out its migration between different hosts while conserving its data, state and code. The platform is the environment of execution. The platform makes it possible to create mobile agents. The
platform offers the necessary elements required by the agents to perform their tasks such as execution, migration towards other platforms and so on. The node where a mobile agent gets created is known as its owner (Figure 1.2). Typical benefits of using mobile agents include

- Asynchronous task execution [19]: While the agent acts on behalf of its owner on a remote host (node), the owner may perform other tasks.

- Bandwidth conservation: If a complex database query needs large data (for example weather data) to be processed and if only a few relevant pieces of information is needed to be extracted then transferring the computation (that is, the agent) to the data would be more beneficial than to ship the data to the computation [19].

- Reduced latency: By migrating to the location of a needed resource an agent can interact with the resource locally (Figure 1.2). Similarly by migrating to the location of a user an agent can react to user actions rapidly [17]. This enables faster response to events remote to the agent owner. Moreover the agent can continue its interaction with the user or resource even if the owner becomes disconnected from the current host. This feature makes the agents particularly attractive for mobile computing applications [17].
• Support heterogeneity: Distributed computing often incorporates heterogeneous devices from both hardware and software perspectives. Since mobile agents are generally independent of physical, data link, network and transport layer (dependent only on their execution environments), they provide optimal conditions for seamless system integration [20].

• More dynamics: Mobile agents can sense their host environment and react autonomously to changes. Multiple mobile agents have the unique ability of distributing themselves among the hosts in the network to maintain the optimal configuration for solving a particular problem [20].

Most distributed applications can fit naturally into mobile agent model since mobile agents can migrate sequentially through a set of hosts (nodes), send out a wave of child agents that can visit multiple hosts in parallel, remain stationary and interact with resources remotely or any combination of these three extremes [17]. Thus the owner of an agent can decide the sequence of hosts to visit or a mobile agent can decide its next destination host on the fly.

A very large distributed computing system consisting of a large number of computers and communication links, almost always functions with some non functional part. Over time, only the identity and number of failed components change [21]. Failures arise from various sources including software bugs, performance overload, severe congestion, electronic component failures, or malicious subversion [22]. Additionally, environmental disasters such as fires, floods and earthquakes etc., shut down portions of distributed systems.

Thus the ability of a distributed system (based on message passing or mobile agents) to perform in presence of above mentioned conditions should be thoroughly analyzed before the system is commercially implemented. The analysis will be able to tell whether users can depend on that system or not.

1.3 Dependability of System

Dependability of any computing system may be defined as the trustworthiness of the system which allows reliance to be justifiably placed on the service it delivers [1]. Service delivered by a system is its behaviour as perceived by its users. User may be considered to be another system that communicates with the former at the service interface. When a delivered service deviates from the system function we consider a system failure to have occurred. Error is a part of the system state that may cause a system failure. Fault is the cause of error. When a fault is active, it causes an error and when that error
reaches the system interface, it is likely to cause a system failure. So fault, error and failure are the threats to dependability.

Dependability includes the following attributes of a computing system:

- **Availability**: implies readiness for correct service - it may be defined as “the probability that the system is operating at a specified time t” [23]. It represents the survivability of services.

- **Reliability**: implies continuity of correct service which may be defined as the probability of failure-free software operation for a specified period of time in a specified environment. Reliability can be estimated by applying statistical inference techniques to failure data. Reliability of software system can be achieved by fault prevention or fault removal or fault tolerance or fault/failure forecasting.

- **Maintainability**: implies the ability to undergo repairs and modifications. It is defined as the ease with which a software system or component can be modified to correct faults, improve performance, or other attributes, or adapt to a changed environment.

- **Safety**: absence of catastrophic consequences on the user(s) and the environment.

- **Security**: concurrent existence of availability for authorized users only. It ensures the following
  
  - **Confidentiality**: Absence of unauthorized disclosure of information.
  - **Integrity**: Absence of improper system state alterations.

The attributes of dependability, threats to it and means of achieving it are summarized in Figure 1.3.

A dependable system can be developed by a combined utilization of the four techniques:

- **Fault prevention**: means to prevent the occurrence or introduction of faults. This consists of techniques, such as inspection, whose intent is to eliminate the circumstances by which faults arise. Fault prevention for software can be achieved by structured programming, information hiding, modularization etc. Malicious faults can be prevented by firewalls and similar measures.

- **Fault tolerance**: It is the property that enables a system to continue operating properly in the event of the failure of some of its components, thus ensuring
1. Introduction

Fault tolerance is required since systems are susceptible to failure and therefore ability to tolerate failures becomes a desirable property of such systems. Recovery from errors in fault-tolerant systems can be characterized as:

- roll-forward (takes the system state at that time and corrects it, to be able to move forward.)
- roll-back (reverts the system state back to some earlier, correct version. The technique of checkpointing may be utilized for roll-back recovery.)

Depending on the type/characteristic of application and error, some systems make use of both roll-forward and roll-back recovery for different errors or different parts of one error (that is, the various failures caused by the error). Fault tolerance can be realized by anticipating exceptional conditions and building the system to cope with them, and, in general, aiming for self-stabilization so that the system converges towards an error-free state. However, sometimes the consequences of system failure may be catastrophic, or the cost of making a system sufficiently reliable may be very high. Fault tolerance can be achieved by duplication in the following three ways:

- Replication: Providing multiple identical instances of the same system, directing tasks or requests to all of them in parallel, and choosing the correct result on the basis of a quorum
- Redundancy: Providing multiple identical instances of the same system and switching to one of the remaining instances in case of a failure (fall-back or
1. Introduction

- Diversity: Providing multiple different implementations of the same specification, and using them like replicated systems to cope with errors in a specific implementation.

- Fault removal: It can be defined as the means to reduce the number and severity of faults. During development of software, fault removal can be performed by verification (checking whether the system adheres to given properties), diagnosis and correction. Faults can also be removed during the operational life of a system by:
  - Corrective Maintenance: Changes made to a system to repair flaws in its design, coding, or implementation that have produced one or more errors.
  - Preventive Maintenance: Maintenance specifically intended to prevent faults from causing errors during subsequent operation.

- Fault forecasting: It can be defined as the means to estimate the present number, the future incidence, and the likely consequences of faults. The two aspects of evaluation are:
  - Qualitative or ordinal evaluation: Classifies the failure modes or the event combination that leads to system failure. Examples are failure mode and effect analysis.
  - Quantitative or probabilistic evaluation: Evaluates the extent up to which some of the attributes of dependability are satisfied. Two main approaches for deriving dependability measures are modeling and testing. These two approaches are complementary since modeling needs data obtained from testing or by processing failure data. Examples are Markov chain and stochastic Petri nets.

1.4 Motivation

Dependability can be achieved through fault tolerance by recovering from failures when they occur, or by masking failures on-the-fly. From the point of view of applications, neither the sources of failures nor the design schemes employed to combat them matter; what matters is the end result in terms of reliability and availability properties of the distributed system services these applications need. That is, whether the service that the application provides is dependable or not. The widespread use of mission-critical
applications in areas such as banking, manufacturing, video conferencing, air traffic control, military applications and space exploration has demonstrated a great need for highly available and reliable computing systems. Apart from availability and reliability, these systems must also ensure that system information (in the form of data) should be available (in correct form too) to valid users only. Thus, security is another key issue that affects dependability of systems. An insecure system is vulnerable to attacks where intruders can maliciously change system behaviour or abruptly stop a distributed application making a service unavailable. The success of the above systems depend on the assurance of their security, for example in banking, confidentiality of sensitive data (such as account details) of the customers is necessary. These systems typically have their resources geographically distributed, and are required to remain available for use with very high probability at all times. Long-lived computations and long-term data storage place an additional burden of reliability [21].

Many of these above mentioned applications run on mobile computing system (for example mobile banking, mobile ticketing etc.) and consequently require greater concern as underlying wireless environment is highly unreliable. In such systems the nodes access information through wireless data communication any time and everywhere (motion and location independence) [24]. Therefore, this environment itself introduces new features and aspects to dependability, affecting both availability and reliability of the services of distributed systems.

Thus the dependability of the underlying network becomes a factor that may affect the performance, availability, and strategy of mobile computing systems [25]. Consequently before such a system can be commercially implemented, thorough dependability analysis needs to be done.

As is mentioned in [26] although the importance of dependable systems is widely recognized nowadays, there are no mature implementations of these concepts available yet. The existing systems offer only partial solutions, and often the approaches separate the issues of reliability, availability, security etc. In mobile computing system, the need is even more intense because of its constraints described in the beginning of this chapter and the implementations are scarce.

In the present research work, we mostly concentrate on agent based mobile computing systems as mobile agents seem to better cope with network dynamics than message passing systems. But to design or deploy agents for mobile computing systems following traditional approaches is not worth as perfect information about the environment is not available. Rather in such environments uncertainty, and possibly inaccurate data, are most common. For instance, in MANET each node is responsible for relaying
packets to other nodes. Here the agents can effectively operate if they are aware of the underlying environment and hence can reason about complex network information like dynamically changing topology, resource-constrained computing devices, multi-hop routes, signal strength, packet loss, jitter, and bandwidth limitations [27]. The MANET might become severed, possibly denying agents access to key services requiring agents to deal with uncertain communications. Moreover, packets might be lost in transit, signal strength between any two hosts can change at any moment, and the number of hops or even the existence of routes between hosts can change dynamically. These issues emphasize the agents to be performance aware. Nonetheless in a hostile MANET agents might have to detect and react to network intruders and malicious insiders. Thus agents need to be information aware as well.

But not all agents must be network, information, or performance aware. The application that deploys the agents decide the kind of awareness needed by the agents. It is needless to mention that a network, information, and/or performance aware agent system is expected to perform more reliably over the corresponding unaware counterpart.

1.5 Contribution

Dependability analysis of mobile computing system is a complicated problem for which little attention has been paid. The system can use message passing or mobile agents for communication and can run on both infrastructure or ad hoc networks. In this thesis, we mainly focus on dependability issues of mobile agent based distributed system in MANET. The effectiveness of agent paradigm over message passing is also shown in terms of system reliability in infrastructured network. Here only the technical aspects of dependability are considered, that is dependability is viewed as a combination of availability and reliability while security affects both.

In order to avoid the huge computational complexity arising because of dynamic topology of MANET, all reliability and/or dependability estimation models are developed using Monte Carlo (MC) simulation method. The metric used for reliability prediction of the wireless networks are all operating terminal reliability [28] and k-terminal reliability [29]. For reliability estimation of the agents, we have defined a new metric. It is shown that agents can effectively operate even when the underlying network is unreliable. The quality of service demands of distributed applications are also taken care of. Reliability is estimated for agents that demand a minimum link capacity or tolerate a maximum delay. Thus performance of agents demanding for specific network conditions are taken care of.
Securing agents and the nodes from irrational agents or nodes can increase availability of the agents which in turn make the application more dependable. Consequently trust based reputation schemes are developed to keep the agents aware about trusted nodes and dependability attributes like confidentiality and integrity are taken care of. The reputation based schemes that are proposed are based on Dempster-Shafer belief theory [30] and also address uncertainty associated with MANET environment. In this work, trust measure of a node is viewed as a combination of belief, disbelief and uncertainty. Our schemes can protect the nodes and agents from both active and passive attacks using direct feedbacks from the agents and indirect feedbacks collected by them or from trusted neighborhood.

Both reputation schemes and reliability estimation methods presented are effective for applications where gracefully degraded system performance is acceptable and hence may not always be particularly suitable for all e-commerce based applications. But the proposed reliability prediction or estimation methods can be readily extended to such applications.

As is already mentioned in the previous section, an agent is supposed to provide greater service reliability if it is aware of its environment and can take certain decisions dynamically. But due to network dynamics a migration strategy that may seem to be suitable at a time instant may be rendered useless or even harmful at some other point of time. So a mobile agent needs to continuously learn and update itself about the underlying network to ensure better reliability. Here comes the need for a reinforcement learning technique to decide about a suitable migration strategy in an unpredictable and hostile MANET environment. In this work, on-policy Monte Carlo method [31] is used on the mobile agents to decide about a suitable migration strategy in a hostile MANET. Here an agent while en’route prefers to choose its next destination host following greedy strategy (learned and provided by the owner). This method not only avoids malicious nodes but also selfish nodes as well. The effect of such strategies are evaluated with respect to dependability of the services the agent system provides.

Finally the reliability estimation model is applied to estimate the reliability of service discovery protocol using agents in MANET. Subsequently a new service discovery protocol is developed that ensures sufficient agent and hence protocol reliability.

Thus this thesis aims at providing a framework for dependability evaluation of distributed systems running on wireless networks (with or without infrastructure) with mobile nodes and also investigating ways of achieving higher dependability. In a nutshell, the work in this thesis presents:

- a reliability estimation method for infrastructured network and also for the agents
1. Introduction

operating on it;

• reliability and hence dependability estimation techniques to predict the performance of (agent-based) mobile computing system subject to QoS demands;

• trust based reputation models to protect the agents and nodes from both active and passive attacks;

• reinforcement learning based mobile agents for better dependability;

• reliability estimation technique for a service discovery protocol that uses mobile agents for MANET and design reliable mobile agents for service discovery.

1.6 Organization of the Thesis

While estimating dependability of mobile computing systems, it is evident that mobile agents equipped with security measures can make distributed applications more dependable than message passing systems. Interestingly, dependability can improve even further if agents are made to be environment-aware through some multi-agent learning technique.

Thus the central idea behind this thesis is to develop a mechanism for estimating dependability and finding a way of improving it for distributed systems running on wireless networks. Fostering this idea, the chapter-by-chapter organization is as follows.

1.6.1 Chapter 2: Related Work

This chapter contains the background underlying the developments in the remainder of this thesis. It first reviews existing literatures on the various aspects of dependability in relation to wireless environment with a focus on MANET. These concepts are used to formulate a method for reliability analysis of wireless networks. The reliability metric to be used are also identified.

Later, works on mobile agents and their applications in wireless networks and their reliability and security aspects are discussed. The usage of approximation algorithms like Monte Carlo simulation method for analysing reliability of both the environment and agents form the basis for our reliability estimation models described in later chapters.

Moreover, the concept of reputation based trust models form the basis of our security models for agents. The notion of trust is used to prevent the agents from visiting a malicious node thus improving its availability and reliability.
1.6.2 Chapter 3: Mobile Computing System Reliability

This chapter proposes a network reliability estimation model for infrastructured wireless networks followed by the reliability estimation of its agents. The simulation experiments done indicates the fact that mobile agents can tolerate poor network connectivity and helps an application to provide a decent reliability even when the underlying environment goes unreliable.

1.6.3 Chapter 4: Agent Reliability Estimate

This chapter details the problem of reliability estimation of mobile agents on MANET and forms the basis for our reliability estimation model being developed. Moreover, our abstraction of MANET is defined in this chapter using both non homogeneous Poisson distribution and mobility models. Multipath propagation of radio signals is considered for assessing link characteristics. This abstraction is used to define MANET in subsequent chapters as well.

1.6.4 Chapter 5: Agents for QoS MANET Applications

This chapter is based on the reliability estimation model developed in previous chapter. Here reliability is evaluated with respect to the Quality of Service (QoS) demanded by the distributed applications and hence the agents. Two QoS parameters are considered - link capacity and delay. It is found that reliability of the Mobile Agent based System (MAS) manages to attain a steady state even when agents with heterogeneous QoS demands roam around the network. But obviously reliability of applications having low QoS demands are found to supersede others.

1.6.5 Chapter 6: Agent Security in MANET

This chapter discusses about security, considered to be another aspect of dependability. The threats to a mobile agent in MANET are discussed. Distributed trust based reputation model is used to overcome the same. Here only active attacks are considered where either an agent’s code and/or data can be modified or the agent could not retract back to its owner in time due to some routing layer attacks. DoS attack is also considered.

1.6.6 Chapter 7: Reliable and Secure Agent based System on MANET

The contribution of this chapter is threefold: (i) securing agents using reputation scheme from active as well as passive attacks; (ii) estimating reliability of these agents and (iii)
estimating reliability of those nodes in MANET that the agents may migrate to. The metric used to calculate MANET reliability is k-terminal reliability. The results show that the agents spawned by various rational nodes with differing trails can better tolerate MANET uncertainties and ensure a gracefully degraded behavior.

1.6.7 Chapter 8: A Dependable Agent based System

This chapter finally quantifies the term dependability as a combination of attributes like availability and reliability along with security. This chapter presents another interesting dimension to the work - that of using reinforcement learning techniques on agent migration in order to further improve MAS performance.

1.6.8 Chapter 9: Reliable Service Discovery in MANET

This chapter applies the reliability estimation model developed in chapters 4 and 5 to service discovery application in MANET that uses mobile agents. A novel service discovery protocol for MANET is developed that takes care of the agent reliability and attempts to make the entire process of discovering (and spreading) services more reliable.

1.6.9 Chapter 10: Conclusions and Future Work

This chapter summarizes the contributions of this thesis and identifies scope for future research. The limitations and outstanding issues are also discussed.