The concept of an agent can be traced back to Hewitt's Actor Model - "A self-contained, interactive and concurrently-executing object, possessing internal state and communication capability"[42]. The concept of distributed computing implies a network/internet-work of independent nodes which are logically configured in such a manner as to be
seen as one machine by an application. Over the last two decades, the concept of distributed computing has been implemented in varying configurations and on diverse platforms for the optimal processing of data. In current context, a distributed system implies a networked system of nodes in which a single application is able to execute transparently (and concurrently) on data that is (or may be) spread across heterogeneous (hardware & operating system) platforms. The salient payoffs of distributed computing may be listed as:

- Enhanced performance (in respect of both speed up and scale up).
- Resource sharing (in respect of data as well as hardware and software processing elements).
- Fault tolerance and enhanced reliability[56].
- Serves as a basis for grid computing.

The other relevant issues while assessing the relevance of distributed computing vis-à-vis the current computing environment and this Thesis are:

- Interoperability in a heterogeneous environment will continue to be the dominating theme in future applications.
- Advances in Communication technology will continue to fuel the need for more bandwidth and enhanced Quality of service Specifications.
- The rate of increase in data processing and storage is greater than that of data transfer.
- Most users are reluctant to switch platforms, as are developers to switch technology paradigms.
The individual behaviour of the vast number of interconnected nodes based on individual workstations mandate that any service acting upon them universally must be dynamic in nature.

In many computer installations/complexes, a lot of state of the art hardware is not used round the clock. There are times when it is idle, or is under-utilized. When this happens, it may be used by other applications for processing purposes remotely. Networking enables this. Inter-networking further emphasizes the same.

In view of the above, it is forecast that distributed processing of applications and data will no longer be restricted to high end research and scientific applications, but will become as normal as any other service provided over an inter-network. The Internet and the Web themselves are a viable distributed computing domains. Distributed computing however, has yet to gain the type of proliferation mandated by enhanced rates of data processing as well as transfer.

1.1 OBJECTIVES

The primary objective is

1. To optimize the internet-works (including networks), by the use of distributed computing concepts, an agent based Dynamically Distributed Service (DDS) is proposed, which can be made available on demand, in an intranet/inter network. This service will conceptually migrate an application on to different nodes using agents. This Dynamically Distributed Service (DDS) should not be
at variance with distributed paradigms used till date, no changes to the underlying hardware or OS are proposed in its implementation.

There are also a few additional research objectives, as stated below

2. To Identify, the best technology for agent migration.
3. To ensure, that inter-platform migration is as straightforward as intra-platform migration.
4. To evaluate, the performance of DDS.
5. To design efficient, decentralized and fault-tolerant data propagation model which accomplishes tasks with no access to global network information, using the mechanisms of agent propagation, provided by DDS. Operation without a central coordinator eliminates possible bottlenecks in terms of scalability and reliability.
6. To show, that migration helps to improve the efficiency of task completion using a Distributed application.

1.2 PROPOSED MODEL
A model of an on-demand, voluntary Dynamically Distributed Service (DDS) to be used on an intranet/inter-network is proposed. The salient features of the service are:

- **Transparency and Availability.** The most fundamental feature of the DDS is that once the appropriate options have been exercised, it will be fully transparent to the user in the discharge of its functions. Also when a process is moving, it should be available. For example, messages sent to the process during its movement should be received.

- **No Reliance on Central Site.** No reliance on a pre-determined central machine is thought necessary. The machine on which the job
is initially being executed shall dynamically act as the logical processing center. The native Operating System, on an option exercised by the user, shall hand over the job to the DDS, which will proceed to migrate the process to the participant nodes by use of agents. A ‘thread’ is used in the context of not merely a light-weight process but any unit of code that is capable of independent execution on a foreign node. The results of execution shall again converge on the source machine on the termination of execution.

- **Hardware, OS, and Network Independence.** The DDS will interface with heterogeneous platforms, much on the lines of a virtual machine [52][16].

- **Local Autonomy.** The participant nodes within the DDS will be autonomous. This does not imply that processes of a job will not be shared or that they will depend on each other. No site however will depend exclusively on another for the completion of its job. Thresholds based on time and space complexity are set.

- **Distributed Message Passing.** Non-blocking, asynchronous primitives will be used. A ‘send’ or a ‘receive’ will not result in the suspension of the job. Reliability shall be assured by the underlying protocol. Copies of the process shall be made before it is migrated. Buffer management shall be in-built. A standardized format for messages (common objects and data) shall convert the native code on any machine to that fit for migration, and translate into native code for another machine. This process shall be repeated for the return migration. Remote Procedure Calls (RPCs) are[101]relevant to the extent that they merely provide a
special resource (hardware or software) to be exploited by a consumer thread.

The block schematic of the DDS proposed is given in Fig.1.1. It is to be noted that the positioning of the DDS in the scheme shown below is to interface with, not replace, the native OS. In this aspect it varies from other distributed operating systems like Sprite[59] (which is meant to also function as a network operating system), Amoeba[59] (in which the same kernel is implemented using Remote Procedure Call using threads), and Andrew[8] (which differentiates between client and server machines).
1.3 MOTIVATION

Typical workstations show a low average CPU utilization and many idle intervals[2]. These workstations have spare processing capacity and can be accessed by users, who log on at other workstations and feel insufficient computing capabilities at their sites. By using the unused computing capabilities at other sites, high utilization can be achieved.

The other applications where this service can be useful are:

- **Dynamic load distribution**, by migrating processes from overloaded nodes to less loaded ones,

- **Fault resilience**, by migrating processes from nodes that may have experienced a partial failure,

- **Improved system administration**, by migrating processes from the nodes that are about to be shut down or otherwise made unavailable, and

- **Data access locality**, by migrating processes closer to the source of some data.

The efficient utilization of network resources is an important issue. The problem is complex due to the distributed nature of computer networks, high communication demands and the desire for limited communication overheads[48]. One solution to such challenges is to design efficient, decentralized and fault-tolerant data propagation model, which accomplishes tasks with no access to global network information. Mechanisms of agent propagation are useful because agents can be organized into efficient configurations without imposing external centralized controls. Operation
without a central coordinator eliminates possible bottlenecks in terms of scalability and reliability. The other most interesting properties of modern networks are the mechanisms of biologically inspired self-organisation.

With high-performance facilities shifting from super computers to networks of workstations, and with the ever-increasing role of the World Wide Web, DDS is expected to play a more vital role and eventually to be widely adopted[46].

1.4 METHODOLOGY

1.5 CONTRIBUTION AND ORGANISATION OF THE REPORT

In this dissertation, the focus is on migrating agents on to different nodes. The agent technologies and available platforms are studied extensively for this implementation. Its performance is investigated by applying it to distributed calculation of prime numbers. It is also shown how to apply this agent migration to build an efficient, decentralized and fault-tolerant data propagation model which accomplishes tasks with no access to global network information.

In the next chapter, a survey on the issues of Process Migration is presented. Chapter 3 deals with the overview of agents, justification for selection of mobile agents & JADE, for implementing DDS. Chapter 4 provides the details about the design and Chapter 5 is focused on implementation. Chapter 6 consists of discussion on different tests conducted on DDS for evaluating its performance. In Chapter 7, application of this service for solving the prime number generation problem using a
distributed frame work is dealt with. In the last Chapter, the conclusion and future scope are presented.

1.6 OBJECTIVE - CHAPTER MAPPING

<table>
<thead>
<tr>
<th>Objective</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4,5</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>