VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590018

A Seminar Report on

"ROVER Technology"

Submitted in partial fulfillment of the requirements for the award of degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING

Submitted by:

POOJA D

4JD17CS028

Under the Guidance of

Shrikant Pujar M.Tech
Assistant Professor,
Dept. of CSE,
Jain Institute of Technology, Davangere

JAIN INSTITUTE OF TECHNOLOGY
Department of Computer Science and Engineering
Davangere-577005 2020-2021

Certificate

This is to Certify that the seminar work entitled **“ROVER Technology”** is bonafide work carried out by **Pooja D (4JD17CS028),** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University, Belagavi**, during the year 2020-2021. It is certified that all the corrections/suggestions indicated for internal assessment have been incorporated in the report. The Seminar report has been approved as it satisfies the academic requirements in respect of seminar work prescribed for the Bachelor of Engineering degree.

Signature of the Coordinator	**Signature of the Guide**	**Signature of the HOD**
Shafiulla Shariff **Assistant Professor,** **Dept. of CSE** **JIT, DVG**	**Shrikant Pujar** **Assistant Professor,** **Dept. of CSE** **JIT, DVG**	**Dr. Prashantha G R** **HOD,** **Dept. of CSE** **JIT, DVG**

ACKNOWLEDGEMENT

It is my proud privilege and duty to acknowledge the kind of help and guidance received from several people in preparation of this seminar. It would not have been possible to prepare this report in this form without their valuable help, cooperation and guidance.

I wish to record my sincere gratitude to my seminar guide **Prof. Shrikant Pujar** of Computer Science and Engineering Department for guiding me in investigations for this seminar and providing encouragement, constant support and guidance which was of a great help to complete this seminar successfully.

My sincere thanks to **Prof. Shafiulla Shariff** Seminar Coordinator for supporting the work related to this seminar. Their contributions and technical support in preparing the seminar are greatly acknowledged.

I am grateful to **Dr. Prashantha G R**, Head of the Department of Computer Science and Engineering for giving me the support and encouragement that was necessary for the completion of this seminar.

I would also like to express my gratitude to **Dr. Manjunatha T S ,** Principal, for providing us pleasant environment to work in library and for providing laboratory facilities needed to prepare this report.

Last but not the least, we wish to thank our **parents** for financing our studies in this college as well as for constantly encouraging us to learn engineering. Their personal sacrifice in providing this opportunity to learn engineering is gratefully acknowledged.

Place: Name of the Student :

POOJA D(4JD17CS028)

ABSTRACT

Rover stands for Remotely Operated Video Enhanced Receiver. Rover is the next generation of Bluetooth, infrared & cellular services. Rover technology can range from powerful laptops to simple cellular phones. The technology which enables the scalable location aware computing. This involves automation availability of information & services based on the current location of the user. This user makes avail location aware computing through his PDA.PDA is a hand held computer used as a palmtop computer. PDA's commonly have colour screen & audio capabilities to be used as mobile phone, web browsers. Many PDA's can access the internet, intranet & extranet via Wi-Fi. Many PDA's employ touch screen technology. We have designed and implemented Rover, a system that enables location-based services, as well as the traditional time-aware, user-aware and device-aware services. To achieve system scalability to very large client sets, Rover servers are implemented in an "action-based" concurrent software architecture that enables fine-grained application-specific scheduling of tasks.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

Introduction

Location-aware, in addition to the more traditional notions of time-aware, user-aware, and device-aware.

Rover has a location service that can track the location of every user, either by automated location determination technology (for example, using signal strength or time difference) or by the user manually entering the current location (for example, by clicking on a map).

Available via a variety of wireless access technologies (IEEE 802.11 wireless LANs, Bluetooth, Infrared, cellular services, etc.) and devices (laptop, PDA, cellular phone, etc.), and allows roaming between the different wireless and device types.

Rover dynamically chooses between different wireless links and tailors application-level information based on the device and link layer technology.Scales to a very large client population, for example, thousands of users.

Rover achieves this through fine-resolution application-specific scheduling of resources at the servers and the network.

1.1 Objective

We believe that Rover Technology will greatly enhance the user experience in a large number places, including visits to museums, amusement and theme parks, shopping malls, game fields, offices and business centers.

The system has been designed specifically to scale to large user populations.Therefore, we expect the benefits of this system to be higher in such large user population environments.

1.2 Rover services

Rover offers two kinds of services to its users. We refer to them as basic data services and transactional services.

1. Basic data services: Rover enables a basic set of data services in different media formats, including text, graphics, audio, and video.

Users can subscribe to specific data components dynamically through the device user interface.Depending on the capabilities of the user's device, only a select subset of media formats may be available to the user.

This data service primarily involves one-way interaction; depending on user subscriptions, appropriate data is served by the Rover system to the client devices.

2. Transactional services: These services have commit semantics that require coordination of state between the clients and the Rover servers. A typical example is e-commerce interactions.

Services that require location manipulation are a particularly important class of data services in Rover.The technique used to estimate the location of an object significantly affects the granularity and accuracy of the location information.

Therefore an object's location is identified by a tuple of Value, Error, and Timestamp. The error identifies the uncertainty in the measurement (value).The timestamp identifies when the measurement was completed.

The accuracy of the location information is relevant to the context of its use. For example, an accuracy of _ meters is adequate to provide walking directions from the user's current location to another location about 500 meters away.

However, this same accuracy is inadequate to identify the exhibit in front of the user. User input in these cases, helps significantly improve the accuracy of user location information.

Chapter 2

Rover architecture

2.1 Architecture

A Rover system, depicted in Figure 2.1, consists of the following entities:

2.1.1 End-users of the system

End-users of the system Rover maintain a user profile for each end user, that defines specific interests of the user and is used to customize the content served.

2.1.2 Rover-clients

Rover-clients are the client devices through which users interact with Rover. They are typically small wireless handheld units with great diversity of capabilities in regard to processing, memory and storage, graphics and display, and network interface.

Rover maintains a device profile for each device, identifying its capabilities and thus, the functionality available at that device.

2.1.3 Wireless access infrastructure

Wireless access infrastructure provides wireless connectivity to theRover clients. Possible wireless access technologies include IEEE 802.11 based wireless LANs, Bluetooth and Cellular services.

For certain QoS guarantees, additional mechanisms need to be implemented at the access points of these technologies for controlled access to the wireless interface.

2.1.4 Servers

Servers implement and manage the various services provided to the end-users. The servers consist of the following:

The server system consists of the following set of devices:

– Rover controller: is the "brain" of the Rover system. It provides and manages the different services requested by the Rover clients. It schedules and filters the content sent to the clients based on user and device profiles and their current locations.

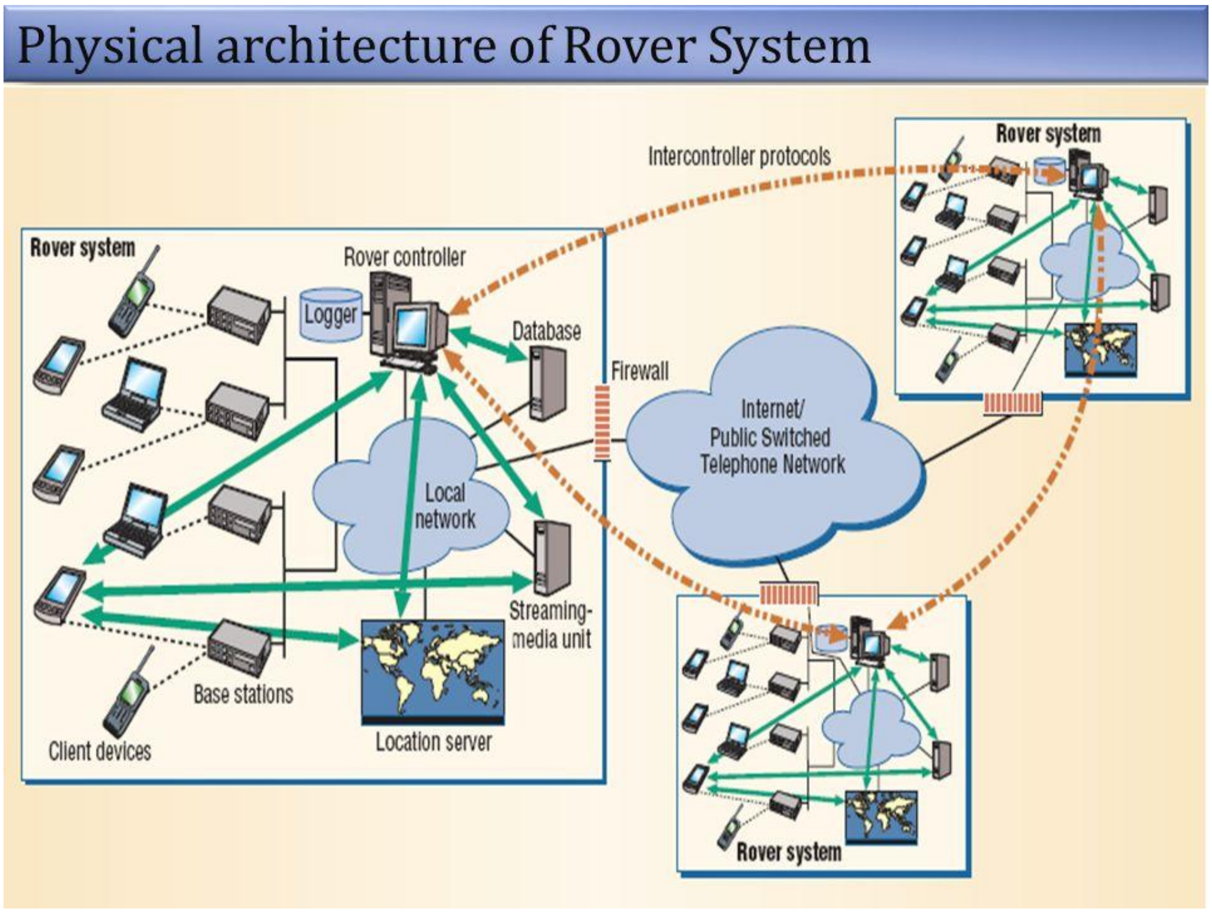

Fig 2.1 Physical architecture of the Rover System

– Location server: is a dedicated unit responsible for managing the client device location services within the Rover system. Alternatively, an externally available location service can also be used.

– Media streaming unit: provides the streaming of audio and video content to the clients. In fact,it is possible to use many of the off-the-shelf streaming-media units that are available today and integrate them in the Rover system.

– Rover database: stores all content delivered to the Rover clients. It also serves as the stable store for the state of the users and clients that is maintained by the Rover controller.

– Logger: interacts with all the Rover server devices and receives log messages from their instrumentation modules.

2.2 System Scalability

There are two potential bottlenecks that can hinder the scalability of such a system to large user populations. One is the server system because it needs to handle a very large number of client requests with tight real-time constraints.

Another potential bottleneck is the bandwidth and latency of the wireless access points.For a server to handle such a large volume of real time requests, in addition to adequate compute power and appropriate data structures, it must have fine grained real-time application-specific scheduling of tasks to efficiently manage the available resources, both processing and bandwidth.

This leads us to divide server devices into two classes' primary servers, which directly communicate with the clients, and secondary servers, which do not directly communicate with clients but interact with primary servers to provide back end capabilities to the system.

The Rover controller, location server and media streaming unit are examples of primary servers, while the Rover database and the logger are examples of secondary servers.

In order to meet the performance objectives, only the primary servers need to implement the fine-grained real-time task scheduling mechanism. We have defined a concurrent software architecture called the Action model that provides such a scheduling mechanism, and implemented the Rover controller accordingly.

A Rover system represents a single domain of administrative control that is managed and moderated by its Rover controller. A large domain can be partitioned into multiple administrative domains each with its own Rover system, much like the existing Domain Name System

For this multi-Rover system, we define protocols that allow interaction between the domains. This enables users registered in one domain to roam into other domains and still receive services from the system

2.3 Action Model

In order to achieve fine-grained real-time application specific scheduling, the Rover controller is built according to concurrent software architecture we call the action model. In this model, scheduling is done in “atomic” units called actions.

An action is a “small” piece of code that does not have any intervening I/O operations.Once an action begins execution, it cannot be preempted by another action.

Consequently, given a specific server platform, it is easy to accurately bind the execution time of an action.The actions are executed in a controlled manner by an Action Controller.

We use the term server operation to refer to a transaction, either client- or administrator-initiated, that interacts with the Rover controller; examples in the museum scenario would be register Device, get Route and locate User.

A server operation consists of a sequence (or more precisely, a partial order) of actions interleaved by asynchronous I/O events.

Each server operation has exactly one “response handling” action for handling all I/O event responses for the operation; i.e the action is eligible to execute whenever an I/O response is received.

A server operation at any given time has zero or more actions eligible to be executed. A server operation is in one of the following three states:

- Ready-to-run: At least one action of the server operation is eligible to be executed but no action of the server operation is executing.
- Running: One action of the server operation is executing (in a multi-processor setup, several actions of the operation can be executed simultaneously).
- Blocked: The server operation is waiting for some asynchronous I/O response and no actions are eligible to be executed.

The Action Controller uses administrator-defined policies to decide the order of execution of the set of eligible actions.

The scheduling policy can be a simple static one, such as priorities assigned to server operations, but it can equally well be time based, such as earliest deadline-first or involving real-time cost functions.

In any case, the controller picks an eligible action and executes it to completion, and then repeats, waiting only if there are no eligible actions (presumably all server operations are waiting for I/O completions).

The management and execution of actions are done through a simple Action API defined as follows:

- In it (action id, function ptr): This routine is called to initialize a new action (identified by action id) for a server operation. Function ptr identifies the function (or piece of code) to be executed when the action runs
- Run (action id, function parameters,deadline, deadline failed handler ptr): This routine is called to mark the action as eligible to run. Function parameters are the parameters used in executing this instance of the action.Deadline is optional and indicates the time (relative to the current time) by which the action should be executed
- Run (action id, function parameters,deadline, deadline failed handler ptr): This routine is called to mark the action as eligible to run. Function parameters are the parameters used in executing this instance of the action.
- Cancel (action id, cancel handler ptr): This routine is called to cancel a ready-to-run action provided it is not executing.

2.4 Actions vs. Threads

There are several ways to use a thread model to implement the Rover controller. One is to implement each server operation as a separate thread. Another is to have a separate thread for each user.

Both of these imply a large number of simultaneously active threads as we scale to large user populations,resulting in large overheads for thread switching.

A more sensible approach is to create a small set of “operator” threads that execute all operations, for example, one thread for all register Device operations, one for all locate User operations, and so on.

Here the thread switching overhead is modest but there are drawbacks. One is that, depending on the threads package, it restricts our ability to optimize thread scheduling, especially as we transit to time based(rather than priority based) scheduling.

More importantly, because each operator thread executes its set of operations in sequence, this approach severely limits our ability to optimally schedule the eligible actions within an operation and across operations.

Of course, each thread could keep track of all its eligible actions and do scheduling at the action level, but this is essentially recreating the action model within each thread.

We compare the performance of action-based and thread-based systems through implementation.

In this paper, we consider two kinds of server operations, one processor-bound and the other, I/O-bound, both of which appear in the context of the Rover controller.

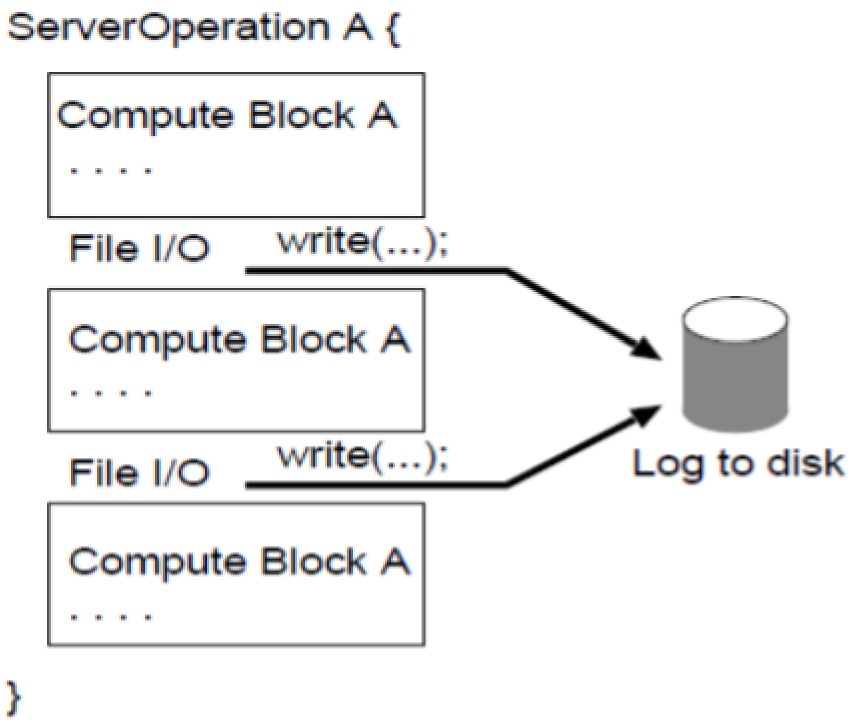

Fig 2.2 Scenario A has 10,000 processor-bound server operations where computation is interleaved with file write operations

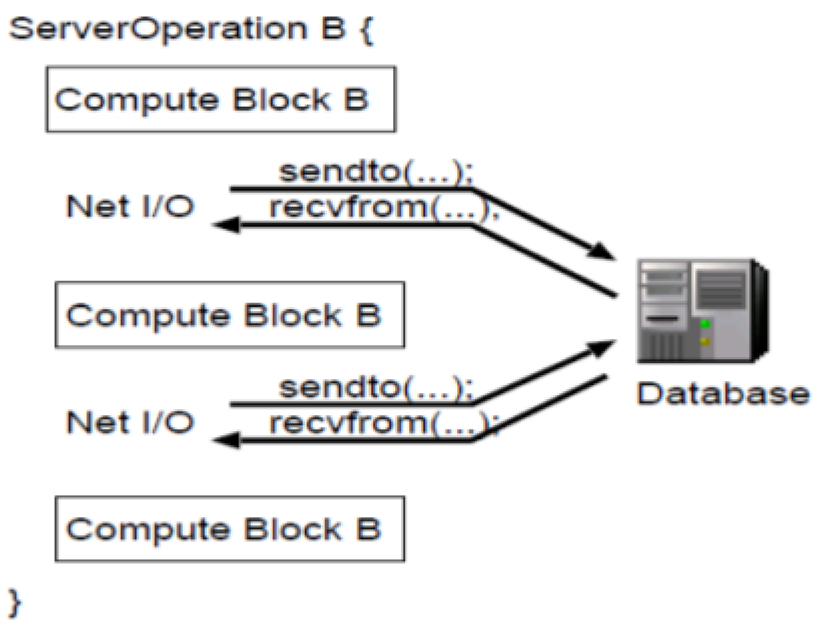

Figure 2.3: Scenario B has 100 I/O bound server operations where computation is interleaved with network I/O interactions

Using these server operations, we construct two corresponding scenarios:

- Scenario A: This is a computation-intensive scenario and has 10,000 processor-bound server operations, where each of the server operations has three compute blocks, interleaved with two file write operations (see Figure 2.2). In each of these server operations, the second and the third I/O compute block does not need to wait for the prior file I/O write operation to complete?
- Scenario B: This is an I/O-intensive scenario andhas 100 I/O-bound server operations, where each of the server operations has three compute blocks, interleaved with two network I/O operations (see Figure 2.3). In each of these server operations, the second and third compute blocks can be initiated only after the completion of the prior network I/O operation. The network I/O interaction was implemented using UDP. Since our focus is on the comparison of the action-based versus the thread based systems, we avoided issues of packet loss and re-transmissions by only considering those experiments where no UDP packets were lost in the network.

As can be observed in both scenarios, the action-based implementation still achieves significantly (about an order of magnitude) less overhead as compared to the best thread-based implementation.

2.5 Rover Clients

The client devices in Rover are handheld units of varying form factors, ranging from powerful laptops to simple cellular phones.

They are categorized by the Rover controller based on attributes identified in the device profiles, such as display properties,screen size and color capabilities, text and graphics capabilities, processing capabilities,ability to handle vector representations and image compression.audio and video delivery capabilities and user interfaces.

For the wireless interface of client devices, we have currently considered two link layer technologies — IEEE 802.11 Wireless LAN and Bluetooth. Bluetooth is power efficient and is therefore better at conserving client battery power

According to current standards, it can provide bandwidths of up to 2 Mbps. In contrast, IEEE 802.11 wireless is less power-efficient but is widely deployed and can currently provide bandwidths of up to 11 Mbps.

In areas where these high bandwidth alternatives are not available, Rover client devices will use the lower bandwidth air interfaces provided by cellular wireless technologies that use CDMA or TDMA based techniques.

Different air-interfaces may be present in a single Rover system or in different domains of a multi-Rover system. In either case, software radios is an obvious choice to integrate different air-interface technologies.

While the location management system is not tied to a particular air interface, certain properties of specific air interfaces can be leveraged to better provide location management.

2.6 Rover Controller

The interaction of the Rover controller with all other components of the system is presented in Figure 2.4. The Rover controller interacts with the external world through the following interfaces:

Location Interface: This interface is used by the Rover controller to query the location service about the positions of client devices.

The location of a device is defined as a tuple representing the estimate of its position (either absolute or relative to some well-known locations), the accuracy of the estimate, and the time of location measurement.

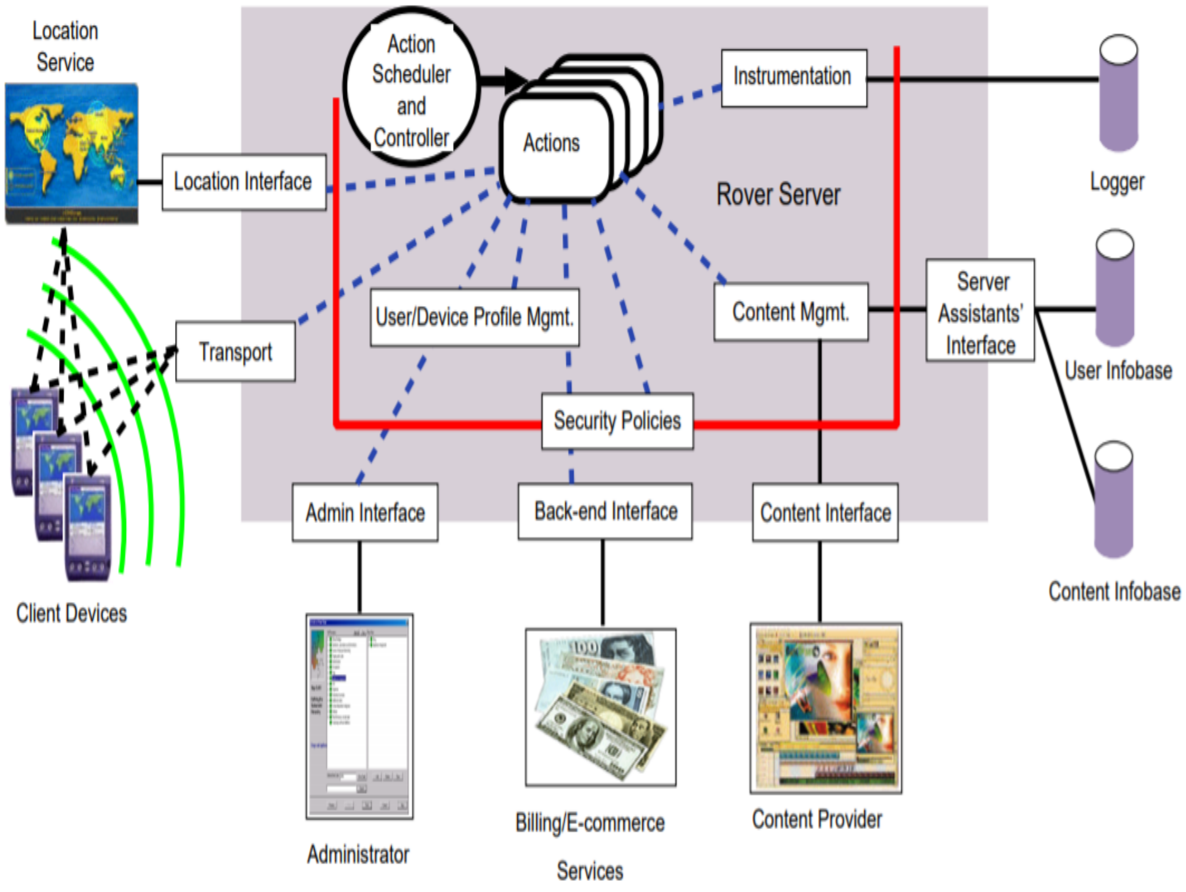

Figure 2.4: Logical architecture of a Rover system

Depending on the technology being used to gain position estimates, The accuracy of the estimate depends on the particulars of the location technology.

for example, GPS , IEEE 802.11 signal strength, signal propagation delays, etc. Rover takes into account this accuracy information when making location-based decisions.

Admin Interface: This interface is used by system administrators to oversee the Rover system, including monitoring the Rover controller, querying client devices, updating security policies, issuing system specific commands, and so on.

-Content Interface: This interface is used by the content provider to update the content that is served by the Rover controller to the client devices. Having a separate content interface decouples the data from the control path.

Back-end Interface: This interface is used for interaction between the Rover controller and certain external services as may be required.

One such service is e-commerce, by which credit card authorization for various purchases can be made. These services would typically be provided by third-party vendors.

Server Assistants Interface: This interface is used for interaction of the Rover controller with the secondary servers. e.g. the database and the streaming media unit.

Transport Interface: This is the communication interface between the Rover controller and the clients, which identify data formats and interaction protocols between them.

2.7 Rover Database

The database in Rover consists of two components, which together decouple client-level information from the content that is served.One component of the database is the user info base, which maintains user and device information of all active users and devices in the system.

It contains all client-specific contexts of the users and devices, namely profiles and preferences, client location, and triggers set by the clients. This information changes at a fairly regular rate due to client activities, e.g. the client location alters with movements.

The Rover controller has the most updated copy of this information and periodically commits this information to the database. For many of these data items (e.g. client location), the Rover controller lazily updates the database.

These are termed as volatile data since any change to these data items are not guaranteed to be accurately reflected by the system across system crashes.

For some others, (e.g. new client registration) the Rover controller commits this information to the database before completing the operation. These are termed as non-volatile data.

The Rover controller identifies some parts of the data to be volatile, so as to avoid very frequent database transactions. The Rover controller does not guarantee perfect accuracy of the volatile data, and thus trades off accuracy with efficiency for these data components.

The transactions of the Rover controller with the database are executed on behalf of the different server operations.

The transactions, by definition, are executed atomically by the database. Additionally, each transaction is identified by two different flags that identify certain properties for execution, as follows:

Lock-Acquiring: If this flag is set, the transaction is required to acquire relevant locks, on behalf of the server operation, to read or write data to the database

Blocking: If a transaction issued by a server operation is unable to access or modify some data due to locks being held by other server operations, it can either block till it successfully reads the data, or it returns immediately to the server operation without successful execution.

2.8 Multi-Rover System

A single Rover system comprises a single Rover controller, other server devices (e.g.,
Rover database and Rover streaming media unit), and a set of Rover clients.

A single system is sufficient for management of Rover-clients in a zone of single administrative control. For example, consider a Rover system in a single museum.

All artifacts and objects on display in the museum are managed by a single administrative entity.There is a single content provider for this system and a single Rover system is appropriate to serve all visitors to this museum.

However, each separate museum has its independent administrative authority. Therefore, we can have a separate Rover system for each of the different museums that are administered separately by each museum authority.

This allows a decentralized administration of the independent Rover systems, locally by each museum authority.However, it is important to provide a seamless experience to visitors as they roam from museum to museum.

A multi-Rover system is a collection of independent Rover systems that peer with each other to provide this seamless connectivity to the user population.

Chapter 3

Implementation

3.1 Initial Implementation

A preliminary test implementation was developed on Windows based systems (Windows 2000 for the controller and Windows CE for the client devices).

The current implementation of the Rover system has been developed under the Linux operating system.

The Rover controller is implemented on an Intel Pentium machine running RedHat Linux 7.1 and the clients are implemented on Compaq iPAQ Pocket PC (model H3650) running the Familiar distribution (release versions 0.4 and 0.5) of Linux for PDAs1.

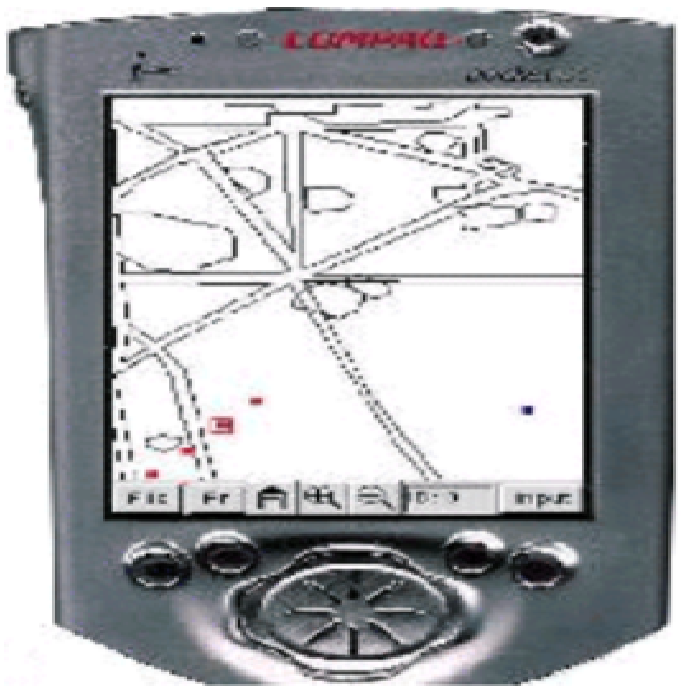

Figure 3.1: View of the display of a Rover-client.

Wireless access is provided using IEEE 802.11 wireless LANs. Each Compaq iPAQ is equipped with a wireless card which is attached to the device through an expansion sleeve.

We have experimented with a set of 8 client devices and have tested various functionalities of the system.

For our outdoor experiments, we interfaced a GPS-device (Garmin eTrex) to the Compaq iPAQs and obtained device location accuracy of between 3-4 meters

The display of the iPAQ Rover-client displays the locations of the different users (represented by the dots) on the area map as shown in Figure 5.

The indoor Rover system is implemented for the 4th floor of the A.V.Williams Building (where the Computer Science Department is located), whose map is shown in Figure 3.2

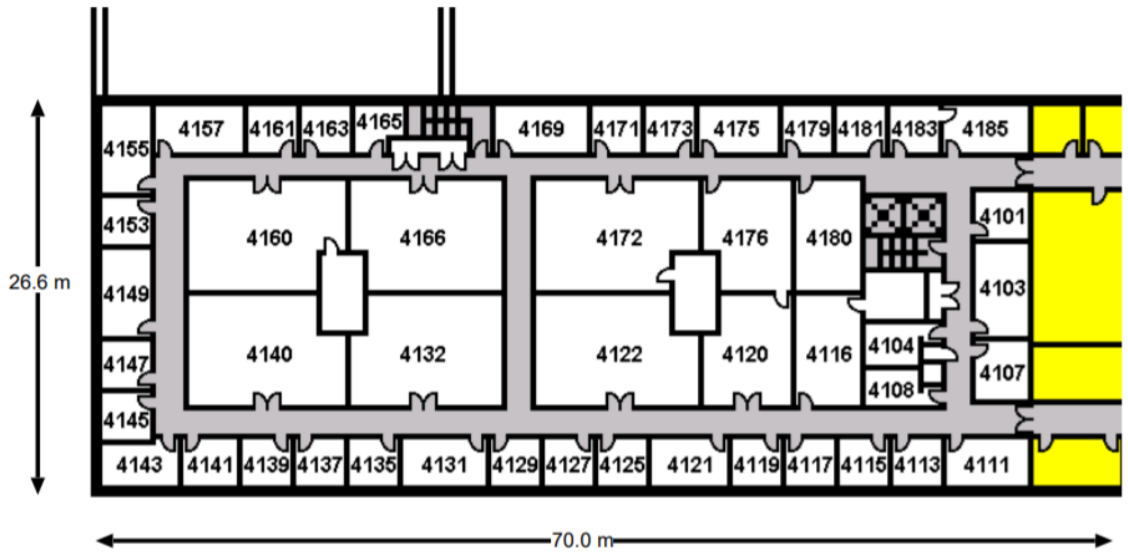

Figure 3.2: The indoor Rover system is currently implemented in the 4th floor of the A.V. Williams Building at the University of Maryland.

In this implementation, the location service is being provided using signal strength measurements from different base stations.

There are about 12 base stations that are distributed all over the building and typically the client device can receive beacons from five or six of the base stations.

We are able to get an accuracy of better than a meter in this environment, using very simple signal strength based estimation techniques.

In both these cases, we implemented the basic functionality of the Rover system.

They include:

- User activation/deactivation and device registration/deregistration procedures.
- Periodic broadcast of events of interest from the Rover controller to the users in specific locations.
- Interaction between users. This can be either simple text messaging or voice chat. Users can optionally make their location visible to other users.

  In the museum example, a tour group coordinator can use this feature to locate all the other members of the group.

- Users can request alerts from the Rover controller when certain conditions are met. The conditions may be time, location or context dependent.

  This can be used to provide notification to ticket holders of an approaching show time. Clearly, for the users who are further away from the show venue, this notification needs to be provided early enough, so that they have enough time to reach the venue.

- An administrator's console allows a global view of all users and their locations in the system.

  The administrator can directly interact with all or a specific subset of the users based on the location or other attributes of the users.

Chapter 4

Rover Application

4.1 Rover Application

- Rover is majorly used in a museum.
- They can also use the devices to reserve and purchase tickets to museum events later in the day.
- Software radio technology offers a way to integrate the different interfaces into a single device.

This would allow the device to easily roam between various Rover systems, each with different wireless access technologies.

- Many of today's off-the-shelf streamingmedia units can be integrated with the rover system.

4.2 Rover Advantages

- Location-aware one of the major advantages is the addition to the more traditional notions of time-aware, user-aware, and device-aware.
- Rover has a location service that can track the location of every user, either by automated location determination technology (for example, using signal strength or time difference) or by the user manually entering the current location (for example, by clicking on a map).
- Available via a variety of wireless access technologies (IEEE 802.11 wireless LANs, Bluetooth, Infrared,cellular services, etc.) and devices (laptop, PDA, cellular phone, etc.), and allows roaming between the different wireless and device types.
- Rover dynamically chooses between different wireless links and tailors application-level information based on the device and link layer technology.

- Scales to a very large client population, for example, thousands of users. Rover achieves this through fine-resolution application-specific scheduling of resources at the servers and the network.

4.3 Rover Disadvantages

- There is a hindrance to its ubiquitous deployment is the lack of system-wide integration of these components in a manner that scales with large user populations.
- Scalability hinders two potential bottlenecks that can hinder the system's scalability.
- One is the server system, which must handle a large number of client requests with tight real-time constraints.
- Another hindrance is the wireless access points, which have limited bandwidth.
- In areas where the high-bandwidth alternatives are not available,communication will be a huge problem.

Conclusion

Rover is currently available as a deployable system using specific technologies, both indoors and outdoors. Our final goal is to provide a completely integrated system that operates under different technologies, and allows a seamless experience of location-aware computing to clients as they move through the system.We believe that Rover Technology will greatly enhance the user experience in a large number places, including visits to museums, amusement and theme parks, shopping malls, game fields, offices and business centers. The system has been designed specifically to scale to large user populations. Therefore, we expect the benefits of this system to be higher in such large user population environments.

References

[1] http://www.bluetooth.com.

[2] http://www.irda.org.

[3] J. Agre, D. Akenyemi, L. Ji, R. Masuoka, and P. Thakkar. A Layered Architecture for LocationbasedServices in Wireless Ad Hoc Networks. In Proceedings of IEEE Aerospace Conference, March 2002.

[4] P. Bahl and V.N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In Proceedings of Infocom, Tel Aviv, Israel, March 2000.

[5] N. Davies, K. Cheverst, K. Mitchell, and A. Efrat. Using and Determining Location in a ContextSensitive Tour Guide. IEEE Computer, 34(8), August 2000.

[6] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. GPS: Theory and Practice. SpringerVerlag,Wein, NY, 1997.

[7] IEEE. Wireless LAN medium access control (MAC) and physical layer (PHY) specification, Standard 802.11,1999.

[8] J. Mitola. The Software Radio Architecture. IEEE Communications Magazine, 5, May 1995.

[9] P. Mockapetris. Domain names - implementation and specification, RFC 1035, November 1987. [10] C.E. Perkins. IP mobility support, RFC 2002, October 1996.

[11] A.J. Viterbi. CDMA: Principles of Spread Spectrum Communications. Addison-Wesley, 1995