ISSN 2229-5518

A Survey on Pervasive Computing

Er. Manita Gorai, Er. Kamna Agarwal

Abstract— This paper discusses the emerging field of pervasive computing which implements the information and communication technologies of daily life. It aims to make life simple by using various tools, which easily manages the information. It focuses on any one any where, at all time concept. It first examines the relationship of this field to its predecessors which are distributed system and mobile computing. Its main focuses on effective use of smart spaces, invisibility, localized scalability, and uneven conditioning. Next, it demonstrates different hypothetical pervasive scenarios which are used to identify the key capability missing from today's system.

Keywords— Distributed Systems, Mobile Computing, Pervasive Computing, Example Scenarios, Drilling Down

1 INTRODUCTION

''The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it.'' So began Mark Weiser's seminal 1991 paper [1] that described his vision of *ubiquitous computing*, now also called *pervasive computing*. The essence of that vision was the creation of environments saturated with computing and communication capability, yet gracefully integrated with human users.

The goal of this paper is to understand the challenges in computer systems research posed by pervasive computing. The beginning is by examining its relationship to the closely-related fields of *distributed systems* and *mobile computing*. Next, there is a sketch of two pervasive computing scenarios, and ask why they are fiction rather than fact today. To preserve focus on computer systems issues, avoid digressions into other areas important to pervasive computing such as human-computer interaction, expert systems and software agents.

2 RELATED FIELDS

Two distinct earlier steps in this evolution are distributed systems and mobile computing. In the rest of this section, sorts out this complex intellectual relationship and to develop taxonomy of issues characterizing each phase of the evolution.

2.1. Distributed Systems

A **distributed system** consists of multiple independent computers that communicate through a computer network. The computers interact with each other in order to achieve a common goal. Field of distributed systems arose at the intersection of personal computers and local area networks. The research that followed have created a conceptual framework and algorithmic base that has proven to be of enduring value in all work involving two or more computers connected by a network — whether mobile or static, wired or wireless, sparse or pervasive.

This body of knowledge spans many areas that are foundational to pervasive computing and is now well codified in textbooks [2-4]:

- Remote communication, including protocol layering, remote procedure call [5], the use of timeouts, and the use of end-to-end arguments in placement of functionality [6].
- Fault tolerance, including atomic transactions, distributed and nested transactions, and two-phase commit [7].
- High availability, including optimistic and pessimistic replica control [8], mirrored execution [9], and optimistic recovery [10].
- Remote information access, including caching, function shipping, distributed file systems, and distributed databases [11].
- Security, including encryption-based mutual authentication and privacy [12].

2.2. Mobile Computing

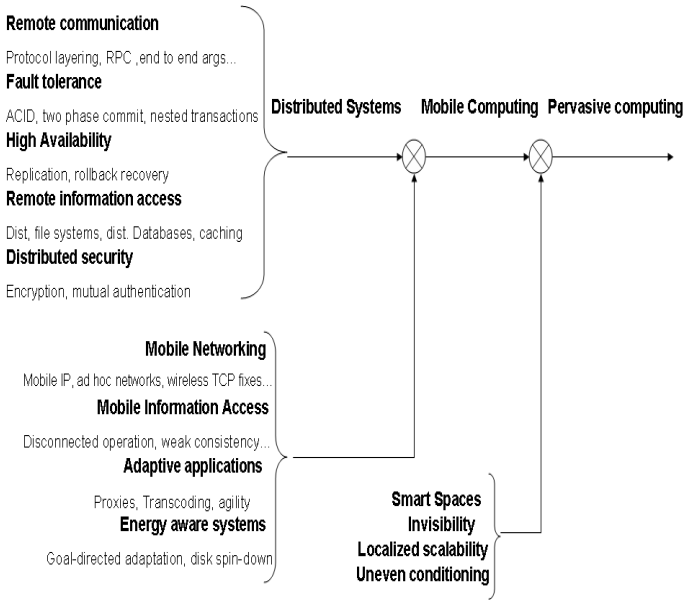
Mobile computing is about physical and logical computing that moves. It is a form of human and computer intera tion through which a computer is expected to be transported while using. Mobile computing has three basic natures: mobile communication, mobile hardware, and mobile software. The mobile communication addresses communication issues in ad-hoc and infrastructure networks as well as communication properties, protocols, data formats and concrete technologies. The second aspect is on the hardware, e.g., mobile devices or device components. The third aspect deals with the characteristics and requirements of mobile applications.It aims in building a distributed system with mobile clients. The field of mobile computing was thus born. Although many basic principles of distributed system design continued to apply, four key constraints of mobility forced the development of specialized techniques. These constraints are: unpredictable variation in network quality, lowered trust and robustness of mobile elements, limitations on local resources imposed by weight and size constraints, and concern for battery power consumption [31]. The results achieved so far can be grouped into the following broad areas:

- *Manita Gorai, M.S in software engineering from BITS Pilani, Rajasthan India, Email: manita.gorai@gmail.com*
- *Kamna Agarwal, pursuing M.Tech, Computer Science from Rajasthan Technical University, Rajasthan , India, E-mail: kamnaagarwal7@gmail.com*

- Mobile networking, including Mobile IP [14], ad hoc protocols [15], and techniques for improving TCP performance in wireless networks [16, 17].
- Mobile information access, including disconnected operation [18], bandwidth-adaptive file access [19], and selective control of data consistency [20, 21].
- Support for adaptative applications, including transcoding by proxies [22] and adaptive resource management [23].
- System-level energy saving techniques, such as energyaware adaptation [24], variable-speed processor scheduling [25], and energy-sensitive memory management [26].
- Location sensitivity, including location sensing [27, 28] and location-aware system behavior [29, 30, 31].

2.3. Pervasive Computing

Pervasive computing is an environment which is saturated with computing and communication capability, yet so gracefully integrated with users that it becomes a ''technology that disappears.'' Such technology must support mobility; otherwise, a user will be acutely aware of the technology by its absence when he moves. Pervasive computing is an application-oriented field and is also not distinctlyu defined and is unstructured for the lack of definitions, algorithms and architectures.Its research is primarily guided by the visions and beliefs because it is tough to justify, in a scientific way, which sytems or applications will permeate in the current market. To bring multiple set of applications into reality, research has to be in multiple disciplines, which will add to the complexity. This exploratory and unstructured nature of pervasive computing research is, therefore, justified to some extent in this paper.Hence, the research agenda of pervasive computing subsumes that of mobile computing, but goes much further. Specifically, pervasive computing incorporates four additional search thrusts into its agenda, as illustrated by Figure 1.



This fig. shows how pervasive computing is related to mobile computing and distributed computing systems. New problems are encountered as we move from left to right in this figure.modulation symbols suggests that the complexity increases in multiplicative rather then additive .it is much difficult to design and implement a pervasive system rather than distributed systems.

Figure 1: Taxonomy of computer systems research problems in pervasive computing.

2.3.1. Effective Use of Smart Spaces

The first research thrust is the *effective use of smart spaces.* A space may be an enclosed area such as a meeting room or corridor, or it may be a well-defined open area such as a courtyard or a quadrangle. By embedding computing infrastructure in building infrastructure, a smart space brings together two worlds that have been disjoint until now [32]. The fusion of these worlds enables sensing and control of one world by the other. A simple example of this is the automatic adjustment of heating, cooling and lighting levels in a room based on an occupant's electronic profile. Influence in the other direction is also possible — software on a user's computer may behave differently depending on where the user is currently located. Smartness may also extend to individual objects, whether located in a smart space or not.

2.3.2. Invisibility

The second thrust is *invisibility.* The ideal expressed by Weiser is complete disappearance of pervasive computing technology from a user's conciousness. In practice, a reasonable approximation to this ideal is *minimal user distraction.* If a pervasive computing environment continuously meets user expectations and rarely presents him with surprises, it allows him to interact almost at a subconcious level [33]. At the same time, a modicum of anticipation may be essential to avoiding a large unpleasant surprise later — much as pain alerts a person to a potentially serious future problem in a normally-unnoticed body part.

2.3.3. Localized Scalability

The third research thrust is *localized scalability.* As smart spaces grow in sophistication, the intensity of interactions between a user's personal computing space and his surroundings increases.Scalability, in the broadest sense, is thus a critical problem in pervasive computing. Previous work on scalability has typically ignored physical distance — a web server or file server should handle as many clients as possible, regardless of whether they are located next door or across the country. The situation is very different in pervasive computing. Here, the density of interactions has to fall off as one move away.

Like the inverse square laws of nature, good system design has achieved scalability by severely reducing interactions between distant entities. This directly contradicts the current ethos of the Internet, which many believe heralds the ''death of distance''.

2.3.4. Masking Uneven Conditioning

The fourth thrust is the development of techniques for *masking uneven conditioning* of environments. The rate of penetration of pervasive computing technology into the infrastructure will vary considerably depending on many non-technical factors such as organizational structure, economics and business models. Uniform penetration, if it is ever achieved, is many years or decades away. In the interim, there will persist huge differences in the ''smartness'' of different this large dynamic range of ''smartness'' can be jarring to a user, detracting from the goal of making pervasive computing technology invisible. One way to reduce the amount of variation seen by a user is to have his personal computing space compensate for ''dumb'' environments. As a trivial example, a system that is capable of disconnected operation is able to mask the absence of wireless coverage in its environment. Complete invisibility may be impossible, but reduced variability is well within the user reach.

3. Example Scenarios

Imagine a world with pervasive computing. To help convey the ''look and feel'' of such a world, we sketch two hypothetical scenarios below. These examples use Aura as the pervasive computing system, but the concepts illustrated are of broad relevance.

3.1. Scenario 1

Jane is at Gate 23 in the Pittsburgh airport, waiting for her connecting flight. She has edited many large documents, and would like to use her wireless connection to e-mail them. Unfortunately, bandwidth is miserable because many passengers at Gates 22 and 23 are surfing the web. Aura observes that at the current bandwidth Jane won't be able to finish sending her documents before her flight departs. Consulting the airport's network weather service and flight schedule service, Aura discovers that wireless bandwidth is excellent at Gate 15, and that there are no departing or arriving flights at nearby gates for half an hour. A dialog box pops up on Jane's screen suggesting that she go to Gate 15, which is only three minutes away. It also asks her to prioritize her e-mail, so that the most critical messages are transmitted first. Jane accepts Aura's advice and walks to Gate 15. She watches CNN on the TV there until Aura informs her that it is close to being done with her messages, and that she can start walking back. The last message is transmitted during her walk, and she is back at Gate 23 in time for *her boarding call.*

3.2. Scenario 2

Fred is in his office, frantically preparing for a meeting at which he will give a presentation and a software demonstration. The meeting room is a ten-minute walk across campus. It is time to leave, but Fred is not quite ready. He grabs his PalmXXII wireless handheld computer and walks out of the door. Aura transfers the state of his work from his desktop to his handheld, and allows him to make his final edits using voice commands during his walk. Aura infers where Fred is going from his calendar and the campus location tracking service. It downloads the presentation and the demonstration software to the projection computer, and warms up the projector. Fred finishes his edits just before he enters the meeting room. As he walks in, Aura transfers his final changes to the projection computer. As the presentation proceeds, Fred is about to display a slide with highly sensitive budget information. Aura senses that
This might be a mistake: the room's face detection and recognition capability indicates that there are some unfamiliar faces present. It therefore warns Fred. Realizing that Aura is right, Fred skips the slide. He moves on to other topics and ends on a high note, leaving the audience impressed by his polished presentation.

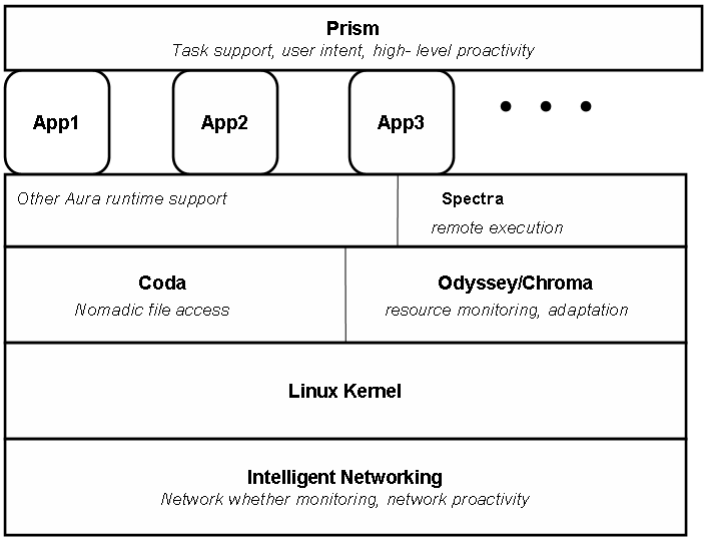
3.3. Missing Capabilities

These scenarios embody many key ideas in pervasive computing. Scenario 1 shows the importance of *proactivity:* Jane is able to complete her e-mail transmission only because Aura had the foresight to estimate how long the whole process would take. She is able to begin walking back to her departure gate before transmission completes because Aura looks ahead on her behalf. The scenario also shows the importance of combining knowledge from different layers of the system. Wireless congestion is a low-level system phenomenon; knowledge of boarding time is an application or user-level concept. Only by combining these disparate pieces of knowledge can Aura help Jane. The scenario also shows the value of a smart space. Aura is able to obtain knowledge of wireless conditions at other gates, flight arrival/departure times and gates, and distance between gates only because the environment provides these services.
Scenario 2 illustrates the ability to move execution state effortlessly across diverse platforms — from a desktop to a handheld machine, and from the handheld to the projection computer. *Self-tuning,* or automatically adjusting behavior to fit circumstances. The scenario embodies many instances of proactivity: The value of smart spaces is shown in many ways: the location tracking and online calendar services are what enable Aura to infer where Fred is heading; the software-controlled projector enables warmup ahead of time; the camera-equipped room with continuous face recognition is key to warning Fred about the privacy violation he is about to commit.
Perhaps the biggest surprise in these scenarios is how simple and basic all the component technologies are. The hardware is all here today. The component software technologies have also been demonstrated: location tracking, face recognition, speech recognition, online calendars, and so on.
Why then do these scenarios seem like science fiction because the real research is in the *seamless integration of component technologies* into a system like Aura? The difficult problems lie in architecture, component synthesis and system-level engineering.

4. Drilling Down

Practical realization of pervasive computing will require us to solve many difficult design and implementation problems. The goal is only to convey an impressionistic picture of the road ahead. There is no claim of completeness or exclusiveness — this specific set of topics is merely a sampling of the problem space, presented in no particular order.
In this discussion, assumption is that each user is immersed in a personal computing space that accompanies him everywhere

and mediates all interactions with the pervasive computing elements in his surroundings. This personal computing space is likely to implement on a body-worn or handheld computer. Refer to this entity as a ''client'' of its pervasive computing environment, even though many of its interactions may be peer-to-peer rather than strictly client-server. As indicated by the discussion below, the client needs to be quite sophisticated and, hence, complex. Figure 2, illustrating the structure of an Aura client, give a concrete example of this complexity.



This figure shows the components of an Aura client and their logical relationships. The text in italics shows the role played by each component. Coda and odyssey were created prior to Aura, but are being modified substantially to meet the demands of pervasive computing.

Figure 2: Structure of an Aura Client.

4.1. User Intent

For proactivity to be effective, it is crucial that a pervasive computing system track user intent. Otherwise, it will be almost impossible to determine which system actions will help rather than hinder the user.

Today's systems are poor at capturing and exploiting user intent.

On the one hand are generic applications that have no idea what the user is attempting to do, and can therefore offer little support for adaptation and proactivity. On the other hand are applications that try to anticipate user intent but do so very badly.

4.2. Cyber Foraging

The need to make mobile devices smaller, lighter and have longer battery life means that their computing capabilities have to be compromised. *Cyber foraging,* construed as ''living off the land'', may be an effective way to deal with this problem. The idea is to dynamically augment the computing resources of a wireless mobile computer by exploiting wired hardware infrastructure. As computing becomes cheaper and more plentiful, it makes economic sense to ''waste'' computing resources to improve user experience. In the forseeable future, envision of public spaces such as airport lounges and coffee shops being equipped with compute servers or data staging servers for the benefit of customers, much as table lamps are today. These will be connected to the wired Internet through high-bandwidth networks. When hardware in the wired infrastructure plays this role, it is called as *surrogate* of the mobile computer it is temporarily assisting.

Envision of a typical scenario is as follows. When a mobile computer enters a neighborhood, it first detects the presence of potential surrogates and negotiates their use. Communication with a surrogate is via short-range wireless peer-to-peer technology, with the surrogate serving as the mobile computer's networking gateway to the Internet. When an intensive computation accessing a large volume of data has to be performed, the mobile computer ships the computation to the surrogate; the latter may cache data from the Internet on its local disk in performing the computation. Alternatively, the surrogate may have staged data ahead of time in anticipation of the user's arrival in the neighborhood. In that case, the surrogate may perform computations on behalf of the mobile computer or merely service its cache misses with low latency by avoiding Internet delays. When the mobile computer leaves the neighborhood, its surrogate bindings are broken, and any data staged or cached on its behalf are discarded.

4.3. Adaptation Strategy

Adaptation is necessary when there is a significant mismatch between the supply and demand of a resource. The resource in question may be wireless network bandwidth, energy, computing cycles, memory, and so on. There are three alternative strategies for adaptation in pervasive computing.

First, a client can guide applications in changing their behavior so that they use less of a scarce resource. This change usually reduces the user-perceived quality, or *fidelity,* of an application. Odyssey [24, 23] is an example of a system that uses this strategy.

Second, a client can ask the environment to guarantee a certain level of a resource. This is the approach typically used by reservation-based QoS systems [34]. From the viewpoint of the client, this effectively increases the supply of a scarce resource to meet the client's demand.

Third, a client can suggest a *corrective action* to the user. If the user acts on this suggestion, it is likely (but not certain) that resource supply will become adequate to meet demand. An example of this approach was described earlier in the paper: in Scenario 1, Aura advised Jane to walk to Gate 15 in order to obtain adequate wireless bandwidth. While conceptually promising, no real system has implemented this approach yet.

All three strategies are important in pervasive computing. The existence of smart spaces suggests that some of the environments encountered by a user may be capable of accepting resource reservations. At the same time, uneven conditioning of environments suggests that a mobile client cannot rely solely on a reservation-based strategy — when the environment is

uncooperative or resource-impoverished, the client may have no choice but to ask applications to reduce their fidelities. Corrective actions broaden the range of possibilities for adaptation by involving the user, and may be particularly useful when lowered fidelity is unacceptable.

4.4. High-level Energy Management

Sophisticated capabilities such as proactivity and self-tuning increase the energy demand of software on a mobile computer in one's personal computing space. At the same time, relentless pressure to make such computers lighter and more compact places severe restrictions on battery capacity. There is growing consensus that advances in battery technology and low-power circuit design cannot, by themselves, reconcile these opposing constraints.

How does one involve the higher levels of a system in energy management? One example is energy-aware memory management [26], where the operating system dynamically controls the amount of physical memory that has to be refreshed. Another example is energy-aware adaptation [24], where individual applications switch to modes of operation with lower fidelity and energy demand under operating system control.

4.5. Client Thickness

How powerful does a mobile client need to be for a pervasive computing environment? The answer will determine many of the key constraints imposed on the hardware design of the client. In trade press jargon, a *thick* client is a powerful client, while a *thin* client is a minimal one.

Thick clients tend to be larger, heavier, require a bigger battery, and dissipate more heat — all negative factors from the viewpoint of the user who has to carry or wear the client. However, those improvements will translate to an even smaller and lighter thin client. For a mobile user, a client can never be too small, too light or have too much battery life!

A wide range of feasible designs has been demonstrated for thin clients. These bare-bones devices are little more than high-resolution displays connected through high-bandwidth wireless links to nearby computer servers. At the other extreme are fullfunction clients capable of standalone or disconnected operation. Such designs can make use of wireless connectivity when available, but are not critically dependent on it.

For a given application, the minimum acceptable thickness of a client is determined by *the worst-case environmental conditions under which the application must run satisfactorily.* A very thin client suffices if one can always count on high-bandwidth, low-latency wireless communication to nearby computing infrastructure, and if batteries can be recharged or replaced easily. If there is even a single location visited by a user where these assumptions do not hold, the client will have to be thick enough to compensate at that location.

With a client of modest thickness, it may be possible to preserve responsiveness by handling simple cases locally and relying on remote infrastructure only for more compute-intensive situations. Alternatively, it may be possible to execute part of the application locally and then ship a much-reduced intermediate state over a weak wireless link to a remote compute server for completion.

The hybrid mode of speech recognition in Odyssey [23] is an example of this approach. Another approach would be for the client to recognize that a key assumption is not being met, and to alert the user with an intelligible message. The client could also suggest possible corrective actions such as moving to a nearby location that is known to be suitable for the application. Uneven conditioning of environments implies that an extreme thin-client approach will be unsatisfactory for pervasive computing in the foreseeable future. At the same time, there is considerable merit in not having to carry or wear a client thicker than absolutely necessary.

4.6. Context Awareness

A pervasive computing system that strives to be minimally intrusive has to be *context-aware.* In other words, it must be cognizant of its user's state and surroundings, and must modify its behavior based on this information. A user's context can be quite rich, consisting of attributes such as physical location, physiological state (such as body temperature and heart rate), emotional state (such as angry, distraught, or calm), personal history, daily behavioral patterns, and so on. If a human assistant were given such context, he or she would make decisions in a proactive fashion, anticipating user needs. In making these decisions, the assistant would typically not disturb the user at inopportune moments except in an emergency. Can a pervasive computing system emulate such a human assistant?

A key challenge is obtaining the information needed to function in a context-aware manner. In some cases, the desired information may already be part of a user's personal computing space. For example, that space may include schedules, personal calendars, address books, contact lists, and to-do lists. More dynamic information has to be sensed in real time from the user's environment. Examples of such information include position, orientation, and the identities of people nearby, locally observable objects and actions, and emotional and physiological state.

Implementing a context-aware system requires many issues to be addressed.

4.8. Privacy and Trust

Privacy, already a thorny problem in distributed systems and mobile computing, is greatly complicated by pervasive computing, the use of location tracking, smart spaces, and surrogates imonitor user actions on an almost continuous basis-complicates pervasive computing. As a user becomes more dependent on a pervasive computing system, it becomes more knowledgeable about that user's movements, behavior patterns and habits. Exploiting this information is critical to successful proactivity and self-tuning. At the same time, unless use of this information is strictly controlled, it can be put to a variety of unsavory uses ranging from targeted spam to blackmail. Indeed, the potential for serious loss of privacy may

deter knowledgeable users from using a pervasive computing system, Greater reliance on infrastructure means that a user must trust that infrastructure to a considerable extent. Conversely, the infrastructure needs to be confident of the user's identity and authorization level before responding to his requests. It is a difficult challenge to establish this mutual trust in a manner that is minimally intrusive and thus preserves invisibility. Privacy and trust are likely to be enduring problems in pervasive computing.

4.9. Impact on Layering

A recurring theme in the earlier sections of this paper has been the merging of information from diverse layers of a system to produce an effective response. For example, Scenario 1 showed the value of combining low-level resource information (network bandwidth) with high-level context information (airport gate information). Proactivity and adaptation based on corrective actions seem to imply exposure of much more information across layers than is typical in systems today. Layering cleanly separates abstraction from implementation and is thus consistent with sound software engineering. Layering is also conducive to standardization since it encourages the creation of modular software components. Deciding how to decompose a complex system into layers or modules is nontrivial, and remains very much an art rather than a science. The two most widely-used guidelines for layering are Parnas' principle of information hiding [35] and Saltzer et al's end-to-end principle [6]. However, these date back to the early 1970's and early 1980's are respectively, long before pervasive computing was conceived [36].

5. Conclusion

Pervasive computing makes ones lives simpler by using digital environments and such environment should be sensitive, adaptive, and human computer interactive. Through the use of mobile computing, Pervasive computing will change the nature of computing, allowing daily life objects to be aware, interacting with users in both the physical and virtual worlds.

There are research challenges in all the areas of pervasive computing the other basic component techn ologies over sensors, smart appliances , digital signal processing and object oriented programming will also exist. Hence, in the near future, homes can be expected to be network intelligent devices that transparently support information and communication needs.There is a need to address research challenges in areas outside computer systems. These areas include human-computer interaction (especially multi-modal interactions and human-centric hardware designs), software agents (with specific relevance to high-level proactive behavior), and expert systems and artificial intelligence (particularly in the areas of decision making and planning). Capabilities from these areas will need to be integrated with the kinds of computer systems capabilities discussed in this paper. Pervasive computing will thus be the crucible in which many disjoint areas of research are fused.

When describing the vision, Weiser was fully aware that attaining it would require tremendous creativity and effort by many, sustained over many years pervasive computing offers new beginnings for the adventurous and the restless — a rich open space where the rules have yet to be written and the borders yet to be drawn.

ACKNOWLEDGMENT

It gives me immense pleasure to put forward this practical venture. But surely, it would not have been possible without proper guidance and encouragement. So I would like to thank all those people without whose support this paper would not have been a success.

I owe a special thanks to Prof. Kamna Agarwal (Associate Proffessor -CSE) for her able guidance and valuable suggestions for choosing and shaping this topic. Her contribution has helped a great deal in the timely completion of this report.

References

[1] Weiser, M: The Computer for the 21st Century.Scientific American, September, 1991.
[2] Couloris, G., Dollomore, J., Kindberg, T: Distributed Systems Concepts and Design (Third Edition). Addison-wesley, 2001.
[3] Lynch, N. A.: Distributed Algorithms.Morgan Kaufmann, 1993.
[4] MULLENDER: DISTRIBUTED SYSTEMS.ADDISON-WESLEY, 1993.
[5] Birrell, A.D, Nelson, B. J: Implementing Rempte Procedure Calls. ACM Transactions on Computer Systems2 (4), February, 1984.
[6] Saltzer,J. H.,Reed,D.P.,Clark,D,D: End-to-End Arguments in SystemDesign. ACM Transactions on Computer Systems2 (4), November, 1984.
[7] Gray,J.,Reuter,A: Transaction Processing: Concepts and Techniques. Morgan Kaufman, 1993.
[8] Davidson, S.B., Garcia-Molina, H., Skeen, D: Consistency in Partitioned Networks. ACM Computing Surveys 17(3), September, 1985.
[9] Borg, A., Blau, W., Graetsch, W: Fault Tolerance under Unix. ACM Transactions on Computer Systems 7(1), February, 1989.
[10] Strom, R.E., Yemini, S: Optimistic Recovery in Distributed Systems. ACM Transactions on Computer Systems 3(3), August, 1985.
[11] Satyanarayanan, M.: A Survey of Distributed File Systems. In Traub, J.F., Grosz, B., Lampson, B., Nilsson, N.J: Annual Review of Computer Science. Annual Reviews, Inc 1989.
[12] Needham, R.M and Schroeder, M.D: Using Encryption for Authentication in Large Networks of Computers. Communications of the ACM 21(12), December, 1978.
[13] Satyanarayanan, M: Fundamental Challenges in Mobile Computing. In Proceedings of the Fifteenth ACM Symposium on Principles of Distributed Computing. Philadelphia, PA, May, 1996.
[14] Bhagwat, P., Perkins, C., Tripathi, S: Network Layer Mobility: An Architecture and Survey. IEEE Personal Communications 3(3), June, 1996.
[15] Royer, E.M., Toh, C. K: A Review of Current Routing Protocols for Ad Hoc MOBILE WIRELESS Networks. IEEE Personal Communications 6(2), April, 1999.
[16] Bakre, A., Badrinath, B.R: Handoff and System Support for Indirect TCP/IP. In Proceedings of the Second Usenix Symposium on Mobile & Location-Independent Computing. Ann Arbor.
[17] Brewer, E.A., Katz, R.H., Chawathe, Y., Gribble, S.D., Hodes, T., Nguyen, G., Stemm, M., Henderson, T., Amir, E., Balakrishnan, H., Fox, A., Padmanabhan, V.N., Seshan, S: A Network Architecture for Heterogeneous Mobile Computing. IEEE Personal Communications 5(5), October, 1998.
[18] Kistler, J.J., Satyanarayanan, M: Disconnected Operation in the Coda File System. ACM Transactions on Computer Systems 10(1), February, 1992.

[19] Mummert, L.B., Ebling, M.R., Satyanarayanan M: Exploiting Weak Connectivity for Mobile File Access. In Proceedings of the 15th ACM Symposium on Operating Systems Principles. Copper Mountain Resort, CO, December, 1995.
[20] Tait, C.D., Duchamp, D: An Efficient Variable-Consistency Replicated File Service. In Proceedings of the USENIX File Systems Workshop. Ann Arbor, MI, May, 1992.
[21] Terry, D.B., Theimer, M.M., Petersen, K., Demers, A.J., Spreitzer, M.J., Hauser, C.H: Managing Update Conflicts in a Weakly Connected Replicated Storage System. In Proceedings of the 15th ACM Symposium on Operating Systems Principles. Copper Mountain Resort, CO, December, 1995.
[22] Fox, A., Gribble, S.D., Brewer, E.A., Amir, E: Adapting to Network and Client Variability via On-Demand Dynamic Distillation. In Proceedings of the Seventh International ACM Conference ON Architectural Support for Programming Languages and Operating Systems. Cambridge, MA, October, 1996.
[23] Noble, B.D., Satyanarayanan, M., Narayanan, D., Tilton, J.E., Flinn, J., Walker, K.R: Agile Application-Aware Adaptation for Mobility. In Proceedings of the 16th ACM Symposium on Operating Systems Principles. Saint-Malo, France, October, 1997.
[24] Flinn, J., Satyanarayanan, M: Energy-aware Adaptation for Mobile Applications. In Proceedings of the 17th ACM Symposium on Operating Systems and Principles. Kiawah Island, SC, December, 1999.
[25] Weiser, M., Welch, B., Demers, A., Shenker, S: Scheduling for reduced CPU energy. In Proceedings of the First USENIX Symposium on Operating System Design and Implementation. Monterey, CA, November, 1994.
[26] Lebeck, A.R., Fan, X., Zheng, H., Ellis, C.S: Power Aware Page Allocation. In Proceedings of the Ninth International Conference on Architerctural Support for Programming Languages and Operating Systems. November, 2000.
[27] Want, R., Hopper, A., Falcao, V., Gibbons, J: The Active Badge Location System. ACM Transactions on Information Systems 10(1), January, 1992.
[28] Ward, A., Jones, A., Hopper, A: A New Location Technique for the Active Office. IEEE Personal Communications 4(5), October, 1997.
[29] Schilit, B., Adams, N., Want, R: Context-Aware Computing Applications. In Proceedings of the Workshop on Mobile Computing Systems and Applications. Santa Cruz, CA, December, 1994.
[30] Spreitzer, M., Theimer, M: Providing Location Information in a Ubiquitous Computing Environment. In Proceedings of the 14th ACM Symposium on Operating System Principles. December, 1993.
[31] Voelker, G.M., Bershad, B.N. Mobisaic: An Information System for a Mobile Wireless Computing Environment. In Proceedings of the Workshop on Mobile Computing Systems AND APPLICATIONS. Santa Cruz, CA, December, 1994.
[32] Katz, R.H., Long, D., Satyanarayanan, M., Tripathi, S: Workspaces in the Information Age. In Report of the NSF Workshop on Workspaces in the Information Age. Leesburg, VA, October, 1996.
[33] Weiser, M, Brown, J.S: The Coming Age of Calm Technology. In Denning, P.J., Metcalfe, R.M. (editors), Beyond Calculation: The Next Fifty Years of Computing. Copernicus, 1998.
[34] Nahrsted, K., Chu, H., Narayan, S: QoS-aware Resource Management for Distributed Multimedia Applications. Journal on High-Speed Networking 7(3/4), 1998
[35] Parnas, D.L: On the Criteria to be Used in Decomposing Systems into Modules. Communications of the ACM 15(12), December, 1972.
[36] M. Satyanarayanan: Pervasive Computing: Vision and Challenges. IEEE Personel Communications, 2001.