A Collaborative Resource Discovery Approach for Mobile Systems*

Jameela Al-Jarooodi and Nader Mohamed

College of Information Technology
United Arab Emirates University, P. O. Box 17555, Al Ain, UAE
{j.aljaroodi, nader.m}@uaeu.ac.ae

Abstract: In this paper, we investigate the requirements and challenges of introducing mobility in large distributed systems. One major obstacle is resource discovery in the presence of mobile nodes. Collaborative resource discovery mechanisms are available to handle static to semi-static resources. However, incorporating mobile users introduces a different set of problems. First, how do we know when a resource becomes available? How do we identify the resource, its current location and its moving patterns? A middleware framework to conceptualize the mobile environment and the associated issues is introduced. It involves the collaboration of software agents residing on the group members' nodes to collect, organize and efficiently exchange resources information.

Keywords: Collaborative Agents, Resource Discovery, Middleware, Mobile Computing, Pervasive Computing.

1. Introduction

Pervasive computing is gaining momentum as a driving force for distributed and collaborative applications. One enabling technology for this trend is middleware services that can help form powerful collaborative applications for mobile users thus forming a pervasive computing environment. Meanwhile, resource discovery is an important aspect of efficient utilization of computing resources. Thus it is important to devise efficient resource discovery mechanisms for pervasive computing environments. However, very few approaches address the issues related to physical and computing resources that support distributed environments. More importantly, very few discuss the issues of resource discovery as a middleware-level service. Research in distributed systems middleware [1] provided a basis for developing distributed applications on heterogeneous environments. Furthermore, a framework for collaborative environments in this architecture was developed [2] for static distributed environments.

In this paper we aim to apply similar middleware-based and collaborative techniques for pervasive systems. We explore, in terms of resource discovery, the design issues for these applications and their requirements. Then a middleware framework for the mobile environments is introduced to provide collaborative resource discovery mechanisms. Section 2 provides an overview of some related work and discusses resource discovery concepts. Section 3 introduces the middleware framework, while Section 4 discusses the collaborative resource discovery mechanism. Finally, Section 6 concludes the paper.

2. Related Work

Mobile devices with networking/computing capabilities like Cell phones, PDA, GPS devices, and laptops have made pervasive computing possible at various levels. However, the heterogeneity of these devices and the various available resources pause several challenges that need to be resolved to enable a wide range of pervasive applications. Currently there is a lot of research being conducted in several disciplines that collectively would form the general pervasive model. Examples of this research are: collaborative systems, mobile ad hoc networks, mobile computing, in addition to distributed systems and middleware.

To develop collaborative applications for mobile networks, there are several challenges [21]: Intermittent/unstable network connections, user mobility and dynamic changes in location, and spontaneous and occasional initiatives for mutual actions. MoCA [21] is middleware architecture to support collaborative applications for mobile users. It requires a partially static (structured) network to operate. Mobile Collaborative Systems, which allow collaboration via wireless network and mobile devices, still lack robust functionality and content representation support [22]. The main enablers for such applications are context-awareness (location, time, system and device status, preferences, connectivity properties, and history) and adaptability [17]. The authors discuss the functional requirements for middleware to support context-aware applications. The defining characteristics for such applications are: can adapt to the environment, can use any network type, are multitasking, and are more user-friendly. While the challenges are: adaptability, wireless communication, distribution of functionality, proximity, and dynamic end-user systems.

In terms of middleware support, several research groups are working in the area. For example the authors explore an open approach to middleware implementation [4] with the motivation to accommodate the demanding requirements for quality of service adaptation imposed by mobile multimedia applications. Middleware for wireless mobile networks was also studied in [10]. Mobile computing presents major challenges for middleware to overcome [9]. In particular frequent changes and poor network QoS. In addition, middleware heterogeneity exists, thus mobile client applications developed upon one type of middleware are

* This research is supported by an individual research grant from The United Arab Emirates University (UAEU).
unable to interoperate and utilize services implemented on an alternative type. The authors propose a dynamically reconfigurable middleware platform, named ReMMoC (Reflective Middleware for Mobile Computing), which allows mobile client applications to be developed independently of the underlying middleware technology. Furthermore, mobility raises new issues such as more dynamic context and limited computing resources [8]. The authors propose a middleware, called 3DMA, which introduces three requirements: distribution, decoupling and decomposition.

In another research the authors present generic interaction techniques for smoothly combining the physical and digital worlds of a mobile user in the context of a collaborative situation [20]. The approach combines techniques used for mobile mixed systems and collaborative mixed systems. To include location awareness more issues arise such as finding the location and providing the required services accordingly. Companies like Intel are working towards making devices aware of their location [7]. The expected characteristics of these devices is that they can use the right resources, are secure, allow sharing and collaboration and can help find the device if lost or stolen. The MINEMA [13] programme aims to unify European research on middleware for mobile environments to foster the definition and implementation of widely recognized middleware abstractions for new and emerging mobile applications. The programme identifies several challenges such as the development of ad hoc, specialized middleware solutions that are hard to re-use in different contexts and the difficulty to create a unified abstraction for this middleware.

The introduction of mobility in large distributed systems requires, among other things, efficient resource discovery and recovery mechanisms. Researchers introduced several approaches for resource discovery in relation to mobile hosts. A statistical framework [3] is described for resource discovery in distributed computing environments with mobile hosts. Denny et al. provide another example [6], where the authors describe Mobiscope, a service for discovering moving resources and identifying their locations. Finally, there is also some limited research in systems resource discovery as part of a larger set of management and allocation services to support distributed computing in various environments such as clusters, the Grid, sensor networks, and networked systems. A mechanism for rapid resource discovery using metadata trading [14] is discussed. SWORD [15][16] is a scalable resource discovery service for wide-area distributed systems using a query mechanism to probe resources. Kong and Berry [12] use the architecture model proposed by DSTC’s architecture unit to specify a general resource discovery system. NEVRLATE [5] is another approach to provide resource discovery mechanisms in peer-to-peer networks using a two-dimensional grid for resource registration and lookup. The classified advertisement (classasad) matchmaking framework [19] provides a flexible and general approach to resource management in distributed environment with decentralized ownership of resources. Based on the examples introduced here and many others explored, we arrive at some interesting observations:(1) Most resource discovery approaches address the issues of information resources. (2) Very few address the resource discovery issues of these resources to support large distributed environments. (3) Very few approach the issues of resource discovery as a middleware-level service. (4) Generally the approaches introduced do not address the introduction of mobility in these environments. In this paper, we approach the issues related to collaborative resource discovery for mobile environments as an integral part of the middleware framework.

Pervasive systems have a unique set of properties [11] that affect the design and implementation process of their applications. Many of these are contextual, thus requiring awareness and reaction to the environment changes. Some of these properties include dynamic computing environments, variable network properties, changing use environments and heterogeneous user platforms. In addition, non-contextual properties are also important and must be addressed. For example; burst-type user access, need for immediate availability of services, and small user devices, thus low power, small memory and limited interface capabilities. Furthermore, middleware design for mobile applications differs significantly from that for large scale distributed systems. Several approaches were discussed for middleware design and in particular for mobile systems and various requirements were identified when considering the design issues [18]. In this paper we define the combined requirements that need to be addressed when designing mobile pervasive collaborative applications.

One of the main components that would support mobile collaboration is the existence of an efficient resource discovery mechanism. This mechanism will allow various nodes interested in collaboration to discover the others and establish the collaboration sessions. In addition, the discovery process will be capable of detecting changes to the system during the process. From the pervasive system’s view point, it is the physical resources and the computing services that need to be discovered. As a result the search domains and allocation techniques are relatively smaller and require different approaches from those used for information discovery. Discovery is an essential step in the process of allocating and managing the mobile systems resources and providing applications with the optimal set of resources. Furthermore, the discovery process requires continuous access to all available resources in the system. As a result, the discovery process becomes more complex due to reasons (1) the increasing size of the group, and (2) the mobility of the group members.

3. The Framework Design

Within a static group of users a hierarchical structure for self organization and resource discovery was introduced in [2]. The proposed structure can support a large group of resources and efficiently manage them. However this structure relies on the static location of resources and the underlying infrastructure. The introduction of mobile users adds to the complexity of the process and makes it difficult to maintain a
fixed hierarchical structure. Collaborating groups usually involve a mix of static and mobile members. Therefore, we base the framework design on the existence of at least one static reachable node in the group. The remaining nodes may be static or mobile and can join/leave the group dynamically in an ad hoc manner. The proposed framework is designed to take advantage of the hierarchical structure while being able to monitor and accommodate mobile resources as well. Each node in the group would be equipped with a software agent. The agents in the static nodes will initiate the process by establishing connections with each other. The group could then elect one node to be the group leader. Leadership in this case is limited to topology and resource management issues and has nothing to do with the actual functionality of the group or the structure they use. Therefore, the framework’s software agents are designed to be generic and would be usable in a wide range of applications. In the general types of collaborative and pervasive applications the group size is usually small and can be easily managed by one leader. However, if the group size grows and the interaction between the members is high, relying on a single leader is not efficient and not feasible. As a result, the leader agent is designed to have organization criteria that will allow it to decide when and how the group should be reorganized based on two cases:

1. When the size of the group reaches a given threshold.
2. When the amount of interaction between the leader and group members increases to a defined threshold value.

As a result, the structure will be divided into subgroups that will each elect their own leader, while the top-level leader maintains a high level view of the overall resources. In this case the second level leaders will handle the local resource management issues, while providing the top level leader with a collective view of their subgroups. At this stage the interaction between members will remain local if all members belong to the same subgroup. In case members need to interact with members in other subgroups, the connections are established through the second level leaders who in turn will interact through the top level leader. When the resources are discovered members are informed of the best way to contact other members and they could communicate directly.

An agent on a mobile node will be responsible for advertising its existence and request to join the group. As soon as the initial organization is established each agent would listen to see if there are mobile nodes in range. Three possible scenarios may occur:

1. The top level leader makes the discovery. In this case the mobile agent is directly linked to the group through that leader and it is provided with the necessary connectivity information to collaborate with the rest of the group.
2. A second level leader makes the discovery. In this case the mobile agent joins that particular subgroup and receives its connectivity information its leader to collaborate with other members. Here the second-level leader needs to update the top level leader with the new member information such that other subgroups would be aware of its existence.
3. A group member makes the discovery. In this case there are two options:

   a. The group member checks if the mobile agent is within range of its second level leader. If that is true, the mobile agent is directed to contact the second level leader and join the subgroup as in case 2.
   b. If the mobile agent is not within range of any of the other leaders, the agent that made the discovery forms a subgroup (third level) with the mobile agent and pass that information to its leader.

Interaction among members can be performed directly (when nodes have direct links to each other) or by using the routing capabilities of the agents to direct their communication when they have no direct connections. In addition, the mobile agent needs to maintain connectivity information including its distance from the leader and the strength of the signal. If the distance increases and/or the strength of the signal decrease, the mobile agent starts a new discovery process to find if there is another agent closer than its current leader. The mobile agent then contacts the new agent and establishes a connection with it following one of the scenarios described above. When the connection is established, the new agent takes over and the old connection with the previous leader is disconnected. As a result, the mobile agents will be able to adapt to their changing location and establish dynamic connections with the rest of the members as they move. The result of this interaction will be the formation of a hierarchical structure where the software agent will be handling the connectivity and availability issues and providing up-to-date information to the participating group members. The members will use this information to perform their collaborative activities.

![Figure 2. Example of the hierarchical structure of the agents.](image)

4. **Dynamic Resource Discovery**

The agents take advantage of the established hierarchical structure to simplify many of their operations. They also utilize automatic startup and configuration mechanisms and dynamic agent allocations that reduce user involvement. The hierarchical structure also provides other advantages such as:
(1) providing scalable mechanisms to easily expand the system. (2) Providing the update and recovery mechanisms for automatic detection of agent failures or relocation and techniques to report errors and adapt to changes. (3) Providing routing capabilities in the leaders, when necessary, to facilitate process communications across subgroups over multi-hop links. (4) Making the agents management and monitoring operations more efficient and less dependent on the full connectivity of the group members. In addition, the framework provides a robust basis for providing resource discovery and recovery services.

5.1 System Resources

To manage resources during the application’s lifetime, information about the member nodes need to be collected and kept up-to-date. Any resource would have two types of attributes to describe it; quantitative and descriptive attributes. The quantitative attributes include measurable parameters such as processor speed and utilization, storage type and capacity, network bandwidth, latency and delays, and service execution time. In addition, mobile nodes also have signal strength and distance from other nodes as attributes. Resources also have descriptive attributes that are very helpful for system management. Information regarding the resource access policies, security levels, ownership, authentication rules, and reliability levels. Many of these are not directly comparable to each other and could lead to difficulties deciding on the utilization of resources when needed. Although some of these attributes are static, many others dynamically change depending on the current state of the system and the applications utilizing it. As a result a continuous monitoring and collection of the status of these attributes is necessary to optimize utilization. However, this process is lengthy and time consuming.

5.2 Discovery Mechanism

We base this mechanism on the hierarchical structure established by the framework. As the agents establish their connections the process of resource discovery also starts. Agents will collaborate to collect the necessary information. Every leader will carry part of the overall responsibility of discovering and keeping track of all the resources under their control. In addition leaders need to effectively communicate this information to other leaders as necessary. At startup initial systems resources information are gathered and shared among leaders, this include information about all members currently in the group including the mobile members. At this phase static attributes are identified, measured, and exchanged and the initial status of the remaining attributes are also collected. When a leader is elected, one of its initial duties is to collect the information as follows:

1. Initialize resources database and identify local resources.
2. Collect resources information from the rest of the group.
3. Sort the information and maintain the two types of information (static and dynamic) separately.
4. Aggregate the information to provide a collective view of resources information to be used by nodes in the group.
5. Send the collective resources information to the rest of the nodes.

When information becomes available, agents and leaders communicate through the hierarchical structure where agents collaborate to satisfy the user application requirements. The second phase is a periodic discovery task that monitors the dynamic attributes and keeps track of their status. In addition, leaders need to exchange some of this information, while maintaining low overhead on the system. In this phase, which continues throughout the agents and leaders lifetime, leaders are responsible for keeping the dynamic resources information updated and ready for use. The second level leaders probe members’ to detect any changes in their attributes. When significant changes occur such as a mobile node changing location and connectivity, the leader will update the top level leader about the changes. The top level leader will provide the updated information to the group members as needed.

The periodic availability checks and updates can be fine-tuned to the system properties to minimize the number of checks performed. This mainly relies on the stability of the system used and the frequency of applications executions. If the system is stable and has low probability of failures and low-level mobility of nodes, then the period between checks can be set to be long, thus reducing the total messages exchanged. However, if the system includes unreliable components, connected through unreliable communications links, or mobile nodes change location frequently, the period should be short enough to discover failures and changes then recover quickly to minimize job failures.

The combined efforts of the agents in the system will provide a complete view of the system resources while minimizing disruption to the applications. The exchange of information is kept to a minimum due to the hierarchical organization of the framework.

5. Conclusion

In this paper we introduced a middleware-based approach for collaborative resource discovery for pervasive applications on mobile distributed systems. The resource discovery process is a costly and complex operation and costs increase drastically for large and dynamic environments. The framework introduced relies on software agents residing on the participating nodes and managing the process of resource discovery. The agents form a hierarchical structure that allows for a localized resource discovery within subgroups, thus reducing the overhead. In addition the subgroups strategically exchange information when needed to provide a complete view of the state of the system during its operations. Furthermore, the periodic monitoring allows for dynamically monitoring changing resources and keeping track of mobile nodes. This allows mobile nodes to maintain connectivity and continuing their collaborative functions. The introduced approach however has one disadvantage, which is relying on the existence of at least one fixed node. Although this is generally the case for many pervasive applications, we aim to
further investigate the technique and develop a more flexible model that can manage the groups even if all nodes are mobile.

References


