Chapter 2

Literature Survey
Chapter Overview

This chapter begins by describing the evolution of cellular network and its basic concepts. It introduces call admission control and channel borrowing and their various strategies by which service level agreement can be met. The chapter also explains agent architecture and its various models and platforms. It discusses JADE (Java Agent DEvelopment framework), an open source agent platform in detail.

2.1 Cellular Network: Evolution

The idea of using cells for wireless communication originated in 1947 from the US AT&T Bell Laboratories and was published in 1979. The first generation (1G) cellular systems (of which the Nordic Mobile Telephone System (NMT) and the American Mobile Phone System (AMPS) are examples) are also known as analogue cellular systems because frequency modulation was used with no digital coding. In 1G mobile, the focus was on speech and basic mobility, but the networks could also support data communication albeit with a very low data rate (upto 2.4 kbit/s). Because of the low data rate, it was only suitable for a few applications (such as paging, reading text emails and downloading small files), although it has to be remembered that, at the time, the data rate offered by wired modems was of the same order of magnitude. In addition to the low bit rate for data applications, the most serious limitation of 1G network was the lack of substantial roaming capability, apart from in the Nordic countries. This was exacerbated by the different frequencies and communication protocols being used.

As mobile communications became more popular, the need for global roaming arose and this was one of the main drivers in the second generation mobile systems. Other drivers were overcoming the capacity limitation in 1G network with lower cost and better speech quality, as well as the need for privacy, security and measures to stop cloning. With large-scale integrated circuit technology, digital communications became more practical and more economical than analogue technology [15].

The second generation, 2G, mobile systems began to be deployed in the early 1990s with Global System for Mobile communications (GSM) being introduced in
Europe and later becoming a de facto world standard. In this generation, more advanced mobility and a greater variety of services were implemented to solve the problems of the first generation, with data service being supported up to 9.6 kbit/s. This system also solved the incompatibility problem as European countries dedicated the same frequency bands for cellular telephone service and this was followed by most of the world. GSM900 was introduced first in Europe and followed by GSM1800 to increase the system capacity. GSM has then been adopted in the US, albeit on a different frequency band, GSM1900. Changes in services and networks happened during the 1990s. The demand from users increased radically, including the demand for using wireless Internet access through mobile terminals. As a result of this overall increase in demand, congestion in radio traffic became more of a problem, leading to network operators having to install more and more base stations.

The 2.5G technologies were released to provide more satisfactory Internet service by offering higher data rate and uninterrupted web access while making voice calls. Examples of 2.5G technologies are General Packet Radio Service (GPRS), High Speed Circuit-Switch Data (HSCSD), and Enhanced Data rate for GSM Evolution (EDGE). HSCSD and GPRS were introduced first to increase the user data rate by allowing a mobile station to access more than one time slot per TDMA frame. HSCSD is a circuit-switched technology, which offers a data channel limited to a single 64 kbit/s circuit while GPRS delivers packet-based service over GSM networks. The GPRS radio channel is flexible: ranging from one to eight radio interface time slots in a TDMA frame and delivering data bit rates range from 9 kbit/s up to more than 150 kbit/s per user.

EDGE was introduced in 1999 to offer more efficient modulation, coding, and retransmission schemes, so allowing enhanced data rates and better overall system capacity. EDGE is designed as an add-on to enhance the existing service such as GPRS and HSCSD as the technique can be used to transmit both packet-switched and circuit-switched voice and data services. In general, for the higher bit rate service, a maximum bit rate of 521 kbit/s is achievable for users moving with speeds up to 100 km/h and up to 182 kbit/s for speeds between 100-500 km/h and 4.7 Mbit/s for an indoor environment. For packet-switched data, EDGE can be introduced as a packet-switched enhancement
for GPRS, known as Enhanced GPRS or EGPRS. For the circuit-switched data enhancement, it is known as Enhanced Circuit-Switch Data (ECSD) [15, 16].

However, there was still the problem that the very nature of GSM meant that really high bit rate services were not possible. Hence, the third generation (3G) cellular system (also called the Universal Mobile Telecommunication System (UMTS) [17]), has been introduced with the aim of (a) being spectrally more efficient and (b) allowing more Internet-like services at higher bit rates. 3GPP (The Third Generation Partnership Project) was established in December 1998 as a collaboration agreement. The six Organizational Partners are ARIB (Association of Radio Industries and Businesses), CCSA (The China Communications Standards Association), ETSI (European Telecommunications Standard Institute), ATIS (Alliance for Telecommunications Industry Solutions), TTA (Telecommunications Technology Association), and TTC (Telecommunication Technology Committee). It is developing technical specifications for IMT-2000, the International Telecommunication Union's (ITU) framework for third-generation standards.

UMTS networks offer better service performance by using an advanced access technology called wideband CDMA (Code Division Multiple Access). The system is designed for multimedia communication and integrated services. The two IMT-2000 radio interface proposals are W-CDMA (ARIB/ETSI) and CDMA-2000 (TIA) [18].

New radio access network architecture, GSM/EDGE Radio Access Network (GERAN), has been developed in the 3GPP Release 5 in 2002 based on GSM/EDGE technologies to support UMTS networks. It has been designed to be fully integrated with the UTRAN (Universal Terrestrial Radio Access Network) of the UMTS system and can be connected to the UMTS core network through the Iu interface. The two legacy interfaces are Gb interface (interface between BSS and SGSN for EGPRS) and A interface (interface between BSS and MSC for ECSD).
2.2 Cellular Concepts

The main aim of cellular concept was to provide an efficient use of spectrum and at the same time expanding the coverage area by allowing spectrum to be reused. The basic concept came from the fact that as the distance between transmitter and receiver increases, the signal strength decays. Hence, same frequency can be reused again without causing interference to the existing communication as long as the distance between the two transmitters is sufficient. From this idea, the whole area can be divided into small hexagonal cells. A number of cells, \( N \), are clustered and the whole bandwidth is evenly distributed among \( N \) cells. The range of \( N \) can be calculated using equation below.

\[
N = i^2 + ij + j^2
\]

Here, \( i \) and \( j \) are positive integers. This gives the typical value of \( N \) to be 1, 3, 4, 7, 9, 12, 19 etc. Figure 2.1 illustrates the frequency reuse in GSM.

Figure 2.1: Frequency Reuse in GSM
From Figure 2.1, the cells that have the same colour and name (A-F) utilise the same frequency; they are called co-channel cells. The distance between the two neighbouring co-channel cells, D, or the *reuse distance* can be calculated as follows.

\[
\frac{D}{R} = \sqrt{3N}
\]

2.2

Where, R is the cell radius and N is a number of cells in a cluster. In actual cell planning, a vital concern is the trade-off between co-channel interference and the capacity of the frequency reuse. As the number of cells in a cluster increases, less interference will be obtained as the reuse distance becomes longer, but the capacity will be decreased since the spectrum available in each cell will be reduced.

A number of technologies have been developed in order to increase the capacity; these include techniques such as cell splitting and sectoring. A number of smaller cells are created within an ordinary cell for the cell splitting technique, which means that more base transceiver stations will be located in a cell and hence the existing spectrum can be reused more often, resulting in increasing capacity. For the sectoring technique, several directional antennas are used to replace a single omni-directional antenna; each antenna radiates within a particular section of the cell. Usually, a cell is partitioned into three 120° sectors or six 60° sectors. By doing so, the co-channel interference can be decreased as the interference only comes from the co-channel cells that have antennas radiating toward. Hence, the system capacity is increased without reducing the transmit power. Although techniques such as these provide higher capacity, the cell design can be more complicated and also the hand-off rate is increased [15].

2.2.1 Connection Level Parameters

When a mobile terminal (mobile user) requests service, it may either be granted or denied service. This denial of service is known as call blocking, and its probability as *call blocking probability* (Pb). An active terminal in a cellular network may move from one cell to another. The continuity of service to the mobile terminal in the new cell requires a successful handoff from the previous cell to the new cell. A handoff is successful if the required resources are available and allocated for the mobile terminal. The probability of
a handoff failure is called *handoff failure probability* (Phf). During the life of a call, a mobile user may cross several cell boundaries and hence may require several successful handoffs. Failure to get a successful handoff at any cell in the path forces the network to discontinue service to the user. This is known as call dropping or forced termination of the call and the probability of such an event is known as call dropping *probability* (Pd). In general, dropping a call in progress is considered to have a more negative impact from the user's perspective than blocking a newly requested call.

### 2.2.2 Channel Assignment Schemes

Channels are managed at each cell by channel assignment schemes based on co-channel reuse constraints. Under such constraints, three classes of channel assignment schemes have been widely investigated [6, 19]:

1. Fixed channel assignment (FCA)

2. Dynamic channel assignment (DCA)

3. Hybrid channel assignment (HCA)

In FCA schemes, a set of channels is permanently assigned to each base station. A new call can only be served if there is a free channel available in the cell. Due to non-uniform traffic distribution among cells, FCA schemes suffer from low channel utilization. DCA was proposed to overcome this problem at the expense of increased complexity and signaling overhead. In DCA, all channels are kept in a central pool to be shared among the calls in all cells. A channel is eligible for use in any cell provided the co-channel reuse constraint is satisfied. Although DCA provides flexibility, it has less efficiency than FCA under high load conditions [19]. To overcome this drawback, hybrid allocation techniques, which are a combination of FCA and DCA, were proposed. In HCA each cell has a static set of channels and can dynamically borrow additional channels.
2.2.3 Handoff Schemes

The handoff schemes can be classified according to the way the new channel is set up and the method with which the call is handed off from the old base station to the new one. At call-level, there are two classes of handoff schemes, namely hard handoff and soft handoff [20, 21, 22].

- **Hard handoff:** In hard handoff, the old radio link is broken before the new radio link is established and a mobile terminal communicates at most with one base station at a time. The mobile terminal changes the communication channel to the new base station with the possibility of a short interruption of the call in progress. If the old radio link is disconnected before the network completes the transfer, the call is forced to terminate. Thus, even if idle channels are available in the new cell, a handoff call may fail if the network response time for link transfer is too long. Second generation mobile communication systems based on GSM fall in this category.

- **Soft handoff:** In soft handoff, a mobile terminal may communicate with the network using multiple radio links through different base stations at the same time. The handoff process is initiated in the overlapping area between cells some short time before the actual handoff takes place. When the new channel is successfully assigned to the mobile terminal, the old channel is released. Thus, the handoff procedure is not sensitive to link transfer time. The second and third generation CDMA-based mobile communication systems fall in this category. Soft handoff decreases call dropping at the expense of additional overhead (two busy channels for a single call) and complexity (transmitting through two channels simultaneously). Two key issues in designing soft handoff schemes are the handoff initiation time and the size of the active set of base stations the mobile is communicating with simultaneously. This study focuses on cellular networks implementing hard handoff schemes.
2.3 Service Level Agreement (SLA)

Resource management is becoming increasingly important in mobile networks as the resources at the air interface are necessarily limited. Providing flexible higher bandwidth services in a mobile environment leads to increased complexity in resource control and management due to the variable bandwidth requirements of the applications, the new radio architecture and the varying demands on the fixed part of the infrastructure. Such complexity requires the use of sophisticated control and management techniques. This situation is further complicated by deregulation of telecommunications networks, as users buy services from Service Providers (SP), who in turn buy capacity from Network Providers (NP). Resource management is, therefore inextricably linked to the business strategies of these providers [23].

In setting up such an approach the service provider at least will require some sort of SLA with the network operators and will use these SLAs as part of the decision-making process when allocating traffic. SLA is a contract between SP and NP, or between SP and Customers, that specifies, usually in measurable terms, what services the service provider will furnish (the supporting services, service parameters, acceptable/unacceptable service levels, liabilities, and action to be taken in specific circumstances) and what penalties the service provider will pay if it cannot meet the committed goals.

Corporate customers will also want SLAs between themselves and the service provider; the service provider will also monitor the service it delivers to all its customers. SLAs allow service providers to differentiate themselves from their competitors and allow them to offer different levels of service guarantees. The interactions amongst the different players in SLA management are as shown in figure 2.2 [1, 2, 23].

Specific terms have been defined for use with SLAs:

- **User**: Those who make use of the telecommunication services provided by the service provider; they can be organisations, companies, or individuals.
- **Customer**: Those who pay for the telecommunication services provided by the service provider; they can be organisations, companies, or individuals. In general the user and customer can be separate entities (for example an employee in a corporate customer is the user, but the corporate itself is the customer).

- **Service Provider**: These are the companies providing the communication services.

- **Network Provider**: These are the companies that own/operate the underlying networks.

![Service Level Agreement Model](image)

**Figure 2.2: Service Level Agreement Model**

*Types of SLA*

- **A Horizontal SLA** is “an SLA between two providers being at the same OSI layer (for example two IP domains or two optical transport network domains)” Here; it would be between SP, or between Network Operators/NP (figure 2.3).

- **A Vertical SLA** is an SLA between two providers at two different OSI layers. Here it would be between the SP and NP, or SP and Customer.
The Studies of Multi-Agent System in Mobile Computing for Mobile Application

- **Service Level Specification** (SLS) is the technical specification deriving from the SLA: it can be a precise specification directly related to the SLA, but it can also be an interpretation of the SLA, an adaptation depending on the provider or on the service. SLA/QoS management requires a definition of services, SLS parameters and a classification of these services depending on the SLS parameters.

- **Service Level Management** (SLM) refers to the process of negotiation, SLA articulation, checks and balances, and reviews between the supplier and customer regarding the services and service levels that support the consumer's business path.

Figure 2.3: Types of Service Level Agreement

SLAs are usually generated with the following processes:

1. To understand the business path, business objectives and the user requirements.

2. To compare actual performance against the long-term objectives.
To translate strategy into service and the service into performance metrics that is measurable and meaningful. The generation of an SLA can be a very complicated task but once it has been created, good management is also important in maintaining an efficient service. To provide a reliable service that meets an SLA, it is necessary to be able to monitor reliably how the SLA is being met – and in this network context that means resource admission control, allocation management, and resource allocation handling. In the case of an unsatisfactory scenario, the SLA management has to be able to react rapidly to reinstate the promised Quality of Service (QoS).

### 2.4 Call Admission Control

Call admission control (CAC) [3, 4, 5] is a technique to provide QoS in a network by restricting the access to network resources. An admission control mechanism accepts a new call request provided there are adequate free resources to meet the QoS requirements of the new call request without violating the committed QoS of already accepted calls. There is a tradeoff between the QoS level perceived by the user (in terms of the call dropping probability) and the utilization of scarce wireless resources. The call blocking probability (Pb) and the call dropping probability (Pd) are the relevant QoS parameters based on which three CAC related problems can be identified [24].

1. **MINO**: Minimizing a linear objective function of the two probabilities (Pb and Pd).
2. **MINB**: For a given number of channels, minimizing the new call blocking probability subject to a hard constraint on the handoff dropping probability.
3. **MINC**: Minimizing the number of channels subject to hard constraints on the new and handoff calls blocking/dropping probabilities.

As mentioned before, channels could be frequencies, time slots or codes depending on the radio technology used. Each base station is assigned a set of channels and this assignment can be static or dynamic. MINO tries to minimize penalties associated with blocking new and handoff calls. Thus, MINO appeals to the network provider since minimizing penalties results in maximizing the net revenue. MINB places
a hard constraint on handoff call blocking thereby guaranteeing a particular level of service to already admitted users while trying to maximize the net revenue. MINC is more of a network design problem where resources need to be allocated apriori based on, for example, traffic and mobility characteristics [25]. Since dropping a call in progress is more annoying than blocking a new call request, handoff calls are typically given higher priority than new calls. This preferential treatment of handoffs increases the blocking of new calls and hence degrades the bandwidth utilization. The most popular approach to prioritize handoff calls over new calls is by reserving a portion of available bandwidth in each cell to be used exclusively for handoffs. Figure 2.4 presents the two categories of CAC schemes in cellular networks:

- **Deterministic CAC**: QoS parameters are guaranteed with 100% confidence. Typically, these schemes require extensive knowledge of the system parameters such as user mobility which is not practical, or sacrifice the scarce radio resources to satisfy the deterministic QoS bounds [26].

- **Stochastic CAC**: QoS parameters are guaranteed with some probabilistic confidence. By relaxing QoS guarantees, these schemes can achieve a higher utilization than deterministic approaches. Figure 2.5 depicts a classification of Stochastic CAC schemes proposed for cellular networks [27, 28, 40, 46].

![Figure 2.4: Classification of Call Admission Schemes](image-url)
We classify CAC schemes as follows:

### 2.4.1 Prioritized Schemes

![Diagram of Stochastic Call Admission Control]

In this section we discuss different handoff prioritization schemes, focusing on reservation schemes. Channel borrowing, call queuing and reservation are studied as the most common techniques.

- **Channel Borrowing Schemes**: In a channel borrowing scheme, a cell (an acceptor) that has used all its assigned channels can borrow free channels from its neighboring cells (donors) to accommodate handoffs. A channel can be borrowed by a cell if the borrowed channel does not interfere with existing calls. When a channel is borrowed, several other cells are prohibited from using it. This is called channel locking and has a great impact on the performance of channel borrowing schemes. The number of such cells depends on the cell layout and the initial channel allocation. A complete survey on channel borrowing schemes is provided by Katzela and Naghshinehin [6]. Here Simple Channel Borrowing scheme review is presented as they discussed in the research.
Simple Channel Borrowing Schemes – In the simple borrowing (SB) strategy [29-34], a nominal channel set is assigned to a cell, as in the FCA case. After all nominal channels are used; an available channel from a neighboring cell is borrowed. To be available for borrowing, the channel must not interfere with existing calls. Although channel borrowing can reduce call blocking, it can cause interference in the donor cells from which the channel is borrowed and prevent future calls in these cells from being completed. As shown in [33], the SB strategy gives lower blocking probability than static FCA under light and moderate traffic, but static FCA performs better in heavy traffic conditions. This is due to the fact that in light and moderate traffic conditions, borrowing of channels provides a means to serve the fluctuations of offered traffic, and as long as the traffic intensity is low the number of donor cells is small. In heavy traffic, the channel borrowing may proliferate to such an extent, due to channel locking, that the channel usage efficiency drops drastically, causing an increase in blocking probability and a decrease in channel utilization. Because the set of borrow able channels in a cell may contain more than one candidate channel, the way a channel is selected from the set plays an important role in the performance of a channel borrowing scheme as in [34].

Borrow from the Richest (SBR) - In this scheme, channels that are candidates for borrowing are available channels nominally assigned to one of the adjacent cells of the acceptor cell [31]. If more than one adjacent cell bas channels available for borrowing, a channel is borrowed from the cell with the greatest number of channels available for borrowing. Channel borrowing can cause channel locking. The SBR scheme does not take channel locking into account when choosing a candidate channel for borrowing.

Basic Algorithm (BA) - This is an improved version of the SBR strategy which takes channel locking into account when selecting a candidate channel for borrowing [31, 32]. This scheme tries to minimize the future call blocking probability in the cell that is most affected by the channel borrowing. As in
the SBR case, channels that are candidates for borrowing are available channels nominally assigned to one of the adjacent cells of the acceptor cell.

Basic Algorithm with Reassignment (BAR) - This scheme provides for the transfer of a call from a borrowed channel to a nominal channel whenever a nominal channel becomes available. The choice of the particular borrowed channel to be freed is again made in a manner that minimizes the maximum probability of future call blocking in the cell most affected by the borrowing, as in the BA scheme [32].

Borrow First Available (BFA) - Instead of trying to optimize when borrowing, this algorithm selects the first candidate channel it finds [31]. Here, the philosophy of the nominal channel assignment is also different. Instead of assigning channels directly to cells, the channels are divided into sets, and then each set is assigned to cells at reuse distance. These sets are numbered in sequence. When setting up a call, channel sets are searched in a prescribed sequence to find a candidate channel.

Call Queuing Schemes: Queuing of handoff requests, when there is no channel available, can reduce the dropping probability at the expense of higher new call blocking. If the handoff attempt finds all the channels in the target cell occupied it can be queued. If any channel is released it is assigned to the next handoff waiting in the queue. Queuing can be done for any combination of new and handoff calls. The queue itself can be finite or infinite as shown in figure 2.6. Although finite queue systems are more realistic, systems with infinite queue are more convenient for analysis [28, 35].
- **Reservation Schemes**: The notion of guard channels was introduced in the mid 80s as a call admission control mechanism to give priority to handoff calls over new calls. In this policy, a set of channels called the guard channels are permanently reserved for handoff calls. Hong and Rappaport showed that this scheme reduces handoff dropping probability significantly compared to the non prioritized case. They found that $P_b$ decreases by a significantly larger order of magnitude compared to the increase of $P_d$ when more priority is given to handoff calls by increasing the number of handoff channels [28].

### 2.4.2 Dynamic Reservation Schemes

There are two approaches in dynamic reservation schemes: local and distributed (collaborative) depending on whether they use local information or gather information from neighbors to adjust the reservation threshold. In local schemes, each cell estimates the state of the network using local information only, while in distributed schemes each cell gathers network state information in collaboration with its neighboring cells.

- **Local Schemes**: We categorize local admission control schemes into *reactive* and *predictive* schemes. By reactive approaches those admission policies are referred that adjust their decision parameters, i.e. threshold and reservation level, as a result of an event such as call arrival, completion or rejection. Predictive
approaches refer to those policies that predict future events and adjust their parameters in advance to prevent undesirable QoS degradations.

**Reactive Approaches:** The well-known guard channel (cell threshold, cut-off priority or trunk reservation) scheme (GC) is the first one in this category. GC has a reservation threshold and when the number of occupied channels reaches this threshold, no new call requests are accepted. One natural extension of this basic scheme is to use more than one threshold (e.g. two thresholds [35]) in order to have more control of the number of accepted calls. It has been shown [36] that the simple guard channel scheme performs remarkably well, often better than more complex schemes during periods in which the load does not differ from the expected level.

**Predictive Approaches:** Local admission control schemes are very simple but they suffer from the lack of global information about the changes in network traffic. On the other hand, distributed admission control schemes have access to global traffic information at the expense of increased computational complexity and signaling overhead induced by information exchange between cells. To overcome the complexity and overhead associated with distributed schemes and benefit from the simplicity of local admission schemes, predictive admission control schemes were proposed [37, 38, 39].

**Distributed Schemes:** The fundamental idea behind all distributed schemes [27, 40-47] is that every mobile terminal with an active wireless connection exerts an influence upon the cells in the vicinity of its current location and along its direction of travel. A group of cells which are geographically or logically close together form a *cluster*, as shown in Figure. 2.7. Either each mobile terminal has its own cluster independent of other terminals or all the terminals in a cell share the same cluster. Typically, the admission decision for a connection request is made in cooperation with other cells of the cluster associated to the mobile terminal asking for admission.
A cluster can be either static or dynamic. In the static approach, the size and shape of the cluster is the same regardless of the network situation. In the dynamic approach however, shape and/or size of the cluster changes according to the congestion level and traffic characteristics. The virtual connection tree of [44] is an example of a static cluster while the shadow cluster introduced in [45] is a dynamic cluster. A shadow cluster is defined for each individual mobile terminal based on its mobility information, e.g. trajectory, and changes as the terminal moves. Typically, dynamic clusters have a better performance at the expense of increased complexity. In general, distributed CACs can be categorized into implicit or explicit based on the involvement of cells in the decision making process:

*Implicit Approach*: In this approach, all the necessary information is gathered from the neighboring cells, but the processing is local. The virtual connection tree [45] concept introduced in is an example of an implicitly distributed scheme. In this scheme each connection tree consists of a specific set of base stations where each tree has a network controller. The network controller is responsible for keeping track of the users and resources. Despite the fact that information is gathered from a set of neighboring cells, the final decision is made locally in the network controller.
Explicit Approach: In this approach, not only information is gathered from the neighboring cells, but also the neighboring cells are involved in the decision making process. The shadow cluster [45] concept introduced in is an example of an explicitly distributed scheme. In this scheme a cluster of cells, the shadow cluster, is associated with each mobile terminal in a cell. Upon admitting a new call, all the cells in the corresponding cluster calculate a preliminary response which after processing by the original cell will form the final decision.

2.4.3 Optimal Control

Although optimal policies are more desirable, near-optimal policies are more useful in practice due to the complexity of optimal policies which usually leads to an intractable solution.

Decision theoretic approaches based on Markov Decision Process (MDP) [47] have been extensively studied to find the optimal CAC policy using standard optimization techniques [48]. The CAC receives a reward, which depends on the action and the state. The goal is to find a policy which specifies which action to take in each state, so as to maximize some function (e.g. the mean or expected sum) of the sequence of rewards [48-51].

2.5.4 Other Admission Control Schemes

- Multiple Services Schemes: Moving from single service systems to multiple services systems raises new challenges. Particularly, wireless resource management and admission control become more crucial for efficient use of wireless resources [25, 52-55, 60]. Despite the added complexity to control mechanisms, multiple services systems are typically more flexible in terms of resource management. Usually there are some low priority services, e.g. best effort service, which can utilize unused bandwidth. This bandwidth can be released and allocated to higher priority services upon request, e.g. when the system is fully loaded and a high priority handoff arrives. Figure 2.8 shows a
classification of guard channel based CAC schemes in single service and multiple services systems. In the figure, *multiple cutoff priority* [52] and *thinning scheme* [54] are the multiple services counterparts of GC and FGC schemes in single service systems respectively. In this context, the thinning scheme is proposed as a generalization of the basic FGC for multiple class prioritized traffic.

![Diagram of Guard Channel Schemes](image)

**Figure 2.8: Single and Multiple Service Guard Channel Schemes**

- **Hierarchical Schemes:** Micro/pico-cell systems can improve spectrum efficiency better than macro-cell systems because they can provide more spectrum resources per unit coverage area. However, micro-/pico-cell systems are not cost effective in areas with low user population (due to base station cost) and areas with high user mobility (leading to a large number of handoffs). As a consequence, hierarchical architectures [56-59] were proposed to take advantage of both macro-cell and micro-cell systems.

- **Complete Knowledge Schemes:** User mobility has an important impact in wireless networks. If the mobility pattern is partially [25, 26] or completely [43, 60] known at the admission time then the optimal decision can be made rather easily. Many researchers believe that it is not possible in general to have such mobility information at admission time. Even for indoor environments complete knowledge is not available [26]. Nevertheless, such an imaginary perfect knowledge scheme is helpful for benchmarking purposes [60].
• **Economic Schemes:** Economic models are widely discussed as a means for traffic management and congestion control in providers networks [61-64]. Through pricing, the network can send signals to users to change their behavior. It has been shown that for a given wireless network there exists a new call arrival rate which can maximize the total utility of users [64]. Based on this, the admission control mechanism can adjust the price dynamically according to the current network load in order to prevent congestion inside the network.

In terms of economics, utility functions describe user’s level of satisfaction with the perceived QoS; the higher the utility, the more satisfied the users. It is sometimes useful to view the utility functions as of money a user is willing to pay for certain QoS.

### 2.5 Agents

Agents [9, 10] are software component involving a host of properties by which their functionality can be defined. These properties help further classify agents in useful ways as listed in table 2.1 [65].

<table>
<thead>
<tr>
<th>Property</th>
<th>Other Names</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>reactive</td>
<td>(sensing and acting)</td>
<td>responds in a timely fashion to changes in the environment</td>
</tr>
<tr>
<td>autonomous</td>
<td></td>
<td>exercises control over its own actions</td>
</tr>
<tr>
<td>goal-oriented</td>
<td>pro-active</td>
<td>does not simply act in response to the environment</td>
</tr>
<tr>
<td>temporally</td>
<td>purposeful</td>
<td>is a continuously running process</td>
</tr>
<tr>
<td>continuous</td>
<td></td>
<td></td>
</tr>
<tr>
<td>communicative</td>
<td>socially able</td>
<td>communicates with other agents, perhaps including people</td>
</tr>
<tr>
<td>learning</td>
<td>adaptive</td>
<td>changes its behavior based on its previous experience</td>
</tr>
<tr>
<td>mobile</td>
<td></td>
<td>able to transport itself from one machine to another</td>
</tr>
<tr>
<td>flexible</td>
<td></td>
<td>actions are not scripted</td>
</tr>
<tr>
<td>character</td>
<td></td>
<td>believable &quot;personality&quot; and emotional state.</td>
</tr>
</tbody>
</table>
Agents may be usefully classified according to the subset of these properties. Every agent satisfies the first four properties. Adding other properties produces potentially useful classes of agents, for example, mobile, learning agents. Thus a hierarchical classification based on set inclusion occurs naturally. E.g. Mobile, learning agents are a subclass of mobile agents.

2.5.1 Agent and Multi Agent based Systems

A Multi Agent System (MAS) [67-71] is a loosely coupled network of problem-solver entities that work together to find answers to problems that are beyond the individual capabilities or knowledge of each entity.

MAS are ideally suited to representing problems that have multiple problem solving methods, multiple perspectives and/or multiple problem solving entities. Such systems have the traditional advantages of distributed and concurrent problem solving, but have the additional advantage of sophisticated patterns of interactions. Examples of common types of interactions include: cooperation (working together towards a common aim); coordination (organising problem solving activity so that harmful interactions are avoided or beneficial interactions are exploited); and negotiation (coming to an agreement which is acceptable to all the parties involved). It is the flexibility and high-level nature of these interactions which distinguishes multi-agent systems from other forms of software and which provides the underlying power of the paradigm.

More recently, the term multi-agent system has been given a more general meaning, and it is now used for all types of systems composed of multiple autonomous components showing the following characteristics:

1. Each agent has incomplete capabilities to solve a problem;

2. There is no global system control;

3. Data is decentralized;
4. Computation is asynchronous;

2.5.2 Agent Architecture

To design an agent two basic models were developed. The first was Touring Machines model [71, 72] consisted of three layers: reactive, planning, and modeling. Second model was that of INTERRRAP [73] - which supported modeling of an agent’s environment at different levels of abstraction, time scales and levels of knowledge. Different agent architectures [73, 74] can be used to realize an agent.

2.5.2.1 Deliberative Architecture

Deliberative agent architecture possesses a symbolic representation of decision which are made via logical reasoning, based on pattern matching and symbolic manipulation. This is also known as Beliefs, Desires and Intentions (BDI) architectural model. This architecture is shown in figure 2.9. The basic idea of BDI approach is to describe the internal processing state of an agent by means of a set of mental categories, and to define a control architecture by which the agent rationally selects its course of action based on their representation. These mental categories of belief, desire, and intentions are supplemented by goals and plans. These concepts are described as follows:

- **Belief** of an agent expresses its expectations about the current state of the world and about the likelihood of a course of action achieving certain effects. ‘Beliefs’ are modeled using possible worlds semantics, where a set of possible world is associated with each situation, denoting the worlds the agents belief is possible.

- **Desire** is an abstract notion that specifies preferences over future world states or courses of action. An important feature of desire is that an agent is allowed to have inconsistent desire, and that it does not have to believe that its desires are achievable.

- **Intention**: Since an agent is resource-bounded, it cannot pursue all its goals or options at once. It is often necessary to select a certain goal to commit, even if the set off goal is consistent.
**Goals:** The weak definition of desire mentioned above enforces an additional step of selecting a consistent subset of desires that an agent might pursue. However, there is not yet any commitment to specific course of action. The notion of commitment describes the transition from goals to intentions. Further it is required that the agent believes its goals to be achievable.

**Plans:** Plans are very important for a pragmatic implementation of intentions. Intentions are partial plans of actions that the agent is committed to execute in order to achieve its goals.

This architecture has high capability of general intelligent action towards a goal and agent’s behaviour could be made optimal and highly adaptable to different contingencies. However it has disadvantages like difficulty in logical representation of belief, desire, time, and reasoning is hardly tractable.
2.5.2.2 Reactive Architecture

To overcome the problems of BDI agents, reactive architecture is evolved. In this architecture agents have very limited amount of information and their run-time decisions are based on sensory input and simple situation action rules. These agents are aimed for faster response and robust behavior instead of optimal behavior. The agents of this architecture are not able to execute a complex task that depends on long-term goals or cooperation of agents.

2.5.2.3 Hybrid Architecture

Drawbacks of deliberative and reactive agents can be overcome by combining both these architectures. These Hybrid architectures are designed to respond rapidly to changes in the environment and to provide means to achieve long-term goals. Different functionalities and goals are arranged in different layers that interact in a well-defined control interface. This architecture is discussed in figure 2.10.
2.5.3 Agent Development Standards

Intelligent agents and typically Multi Agent paradigm is targeting adaptive and flexible co-operation, particularly for interoperability between or within distributed systems in a dynamically changing and mobile environment. This section discusses the efforts and the differences between the two groups, Foundations for Intelligent Physical Agents (FIPA) [14] and Object Management Group (OMG) for developing specifications for Mobile Agent System Interoperability Facility (MASIF) [76] agent interaction and mobility standards [77].

FIPAs intelligent agents take less time and transport capacity to migrate due to interoperability provided by agent communication language which has richer set of semantically standardised interactions between static software systems than the mobile agent paradigm. Agent communication paradigm and its languages can be more easily associated to a formal theory for agent interactions. Such theory enables the formal analysis and verification of the global distributed systems and can further increase the reliability of agent-based applications. Also it is easier to analyze the behaviour of an intelligent agent message. Therefore, the receiving intelligent agent can check the messages for subtle security and contract violations. And finally Mobile agents require homogeneous platforms for interoperability, while the intelligent agent paradigm supports the interoperability among heterogeneous environments. Thus FIPA specifications target at efficiency in terms of transport time and capacity, syntactical and semantic interoperability, richness of interaction protocols, security and reliability.

Whereas OMGs started working on a standard called Mobile Agent Facility (MAF), in order to promote interoperability among agent platforms. Later the standards name was changed to MASIF. It is based on agent platforms and it enables agents to migrate from one platform to another. The MASIF identifies a Distributed Agent Environment (DAE) and a Distributed Processing Environment (DPE). In a DAE, there are the following elements:

- **Place**: A place is a context in which an agent can execute, so a place is an execution environment.
Agency: An agency is an agent system. An agency can have several places. An agent system represents a platform that can create, interpret, execute, and transfer agents.

Region: A region is a group of agencies that belong to a single authority.

Two interfaces represent the core of the MASIF standard. The MAFAgentSystem is associated with every agency and provides operations for the management and transfer of agents. The MAFFinder: It is associated with a region. It supports localization of agents, agencies, and places in the scope of a region.

The following agent functionalities are covered by MASIF:

Agent management: This comprises the creation, termination, suspension, and resumption of agents. The MAFAgentSystem provides several methods for this purpose.

Agent tracking: Agencies, places and agents are registered in a region registration component via MAFFinder.

Agent transport: MAFAgentSystem offers two methods to support agent migration.

Agent and agency naming: Standardized syntax and semantics of agent and agency names enable agents and agencies to identify each other and allow clients to identify agents and agencies.

Agent type and location syntax: Agency types provide information about important aspects of specific agencies, such as the used implementation language. The location is standardized in order to enable to locate each other.

MASIF relies on CORBA to handle agent security. MASIF does not address the agent communication aspect. MASIF adopts a mobile agent paradigm, which is more
appropriate in situations where dynamic and autonomous swapping, replacement, modification, and updating of application components are required.

2.5.4.1 FIPA Complaint public available Platforms

- **Agent Development Kit [78] (Tryllian BV):** The ADK is a mobile component based development platform using lightweight runtime environment based on Java that allows to build reliable and scalable industrial strength applications.

- **Agentcities Forum [79]:** Agentcities is an initiative that was first conceived in January 2000 to create a next generation Internet that is based upon a worldwide network of services that use the metaphor of a real or a virtual city to cluster services. These services, ranging from ecommerce to integrating business processes into a virtual organization, can be accessed across the Internet, and have an explicit representation of the capabilities that they offer. The ultimate aim is to enable the dynamic, intelligent and autonomous composition of services to achieve user and business goals, thereby creating compound services to address changing needs. Since its inception the test bed network has grown rapidly to support a wide range of small prototype and small test systems to large demonstrators involving over 170 deployed agent-based services.

- **April Agent Platform [79] (Jonathan Dale and Johnny Knottenbelt):** It is FIPA-compliant agent platform that is designed to be a lightweight and gives powerful solution for developing agent based systems. It is written using the April programming language and the InterAgent Communication System (IMC).

- **Comtec Agent Platform [80]:** Comtec Agent Platform is an open-source, Java based, free implementation of FIPA97-98 agent communication, agent management, agent message transport and some of the applications. Unique feature of it is the implementation of FIPA Ontology Service and Agent/Software Integration in SL2 as the content language.
FIPA-OS [81] (Emporhia): FIPA-OS was the first Open Source implementation of the FIPA standard and has already recorded thousands of downloads. FIPA-OS now supports most of the FIPA experimental specifications currently under development. FIPA-OS is a component-based toolkit implemented in 100% pure Java. One of the most significant contributions recently is a small-footprint version of FIPA-OS (FIPA-OS), aimed at PDAs and smart mobile phones.

Grasshopper [82] (Germany): Grasshopper is an open 100% Java-based mobile intelligent agent platform, which is compliant to both available international agent standards, namely the OMG MASIF and FIPA specifications. Grasshopper includes two optional open source extensions providing the OMG MASIF and FIPA standard interfaces for agent/platform interoperability.

Jack [83] (The Agent Oriented Software Group): JACK Intelligent Agent is an environment for building, running and integrating commercial-grade multi-agent systems using a component-based approach. JACK is based upon the company’s Research and Development work on software agent technologies. The JACK Agent Language is a programming language that extends Java with agent-oriented concepts.

JADE [13] (TILAB, formerly CSELT): JADE simplifies the development of multi-agent applications, which comply with the latest FIPA 2000 specifications. While appearing as a single entity to the outside world, a JADE agent platform can be distributed over several hosts. Agents can also migrate or clone themselves to other hosts of the platform, regardless of the OS. The communication architecture tries to offer (agent transparent) flexible and efficient messaging by choosing, on an as-needed-basis, the best of the FIPA-compliant Message Transport Protocols (MTP) that are activated at platform run time. JADE is implemented in version 3.1 of JAVA and has no further dependency on third-party software.
Java Agent Services [84]: (Fujitsu, Sun, IBM, HP, Spawar, InterX, Institute of Human and Machine Cognition, Comtec, Verizon) The Java Agent Services (JAS) project defines an industry standard specification and API for the deployment of agent platform-service infrastructures. It is an implementation of the FIPA Abstract Architecture within the Java Community. Process [www.jcp.org] initiative and is intended to form the basis for creating commercial grade applications based on FIPA specifications. The API provides interfaces for message creation, message encoding, message transport, directory and naming. This design is intended to ensure that a JAS based system deployment remains transparent to shifts in the underlying technology without causing interruption to service delivery and therefore the business process.

LEAP [85] (Fr): LEAP (Lightweight Extensible Agent Platform) is a development and run-time environment for Intelligent Agents, is the precursor of the second generation of FIPA compliant platforms. It represents a major technical challenge - it aims to become the first integrated agent development environment capable of generating agent applications in the ZEUS environment and executing them on run-time environments derived from JADE, implemented over a large family of devices (computers, PDA and mobile phones) and communication mechanisms (TCP/IP, WAP). In this way LEAP benefits from the advanced design-time features of Zeus and the lightweight and extensible properties of JADE.

ZEUS [86] (BT Labs): ZEUS is an Open Source agent system entirely implemented in Java, developed by BT Labs and can be considered a toolkit for constructing collaborative multi-agent applications. Zeus provides support for generic agent functionality and has sophisticated support for the planning and scheduling of an agent’s actions. Moreover, Zeus provides facilities for supporting agent communication using FIPA ACL as the message transport and TCP/IP sockets as the delivery mechanism. Zeus provides facilities for building agents in a visual environment and support for redirecting agent behavior too.
2.5.3.2 OMG MASIF Complaint Platforms

- **AGLET [87]**: IBM’s mobile agent platform is implemented in Java. An Aglet is actually a Java object that can move from one host on the Internet to another. That is, an aglet that executes on one host can suddenly halt execution, dispatch itself to a remote host, and resume execution there. When the aglet moves, it takes along its program code as well as its data. The hosts need a ATP server to receive aglets.

- **Concordia [88]**: This is Mitsubishi Electric’s Mobile Agent Environment. Concordia is a full-featured framework for the development and management of network-efficient mobile agent applications which extend to any device supporting Java. Concordia is written in Java and is portable to any platform running Java.

- **Grasshopper [82] (Germany)**: Grasshopper is an open 100% Java-based mobile intelligent agent platform, which is compliant to both available international agent standards, namely the OMG MASIF and FIPA specifications. Grasshopper includes two optional open source extensions providing the OMG MASIF and FIPA standard interfaces for agent/platform interoperability.

- **Kafka [89]**: Kafka is an agent library designed for constructing multi-agent based distributed applications from Fujitsu. Kafka is a flexible, extendable, and easy-to-use Java class library for programmers who are familiar with distributed programming. It is based on Java’s RMI. Kafka is now integrated together with Pathwalker, a process-oriented programming library.

2.5.4 JADE (Java Agent Development framework)

As we have used JADE [13, 84, 90, 91] to implement the proposed MAS model in this thesis, this section describes JADE in detail. JADE is an open source framework from TILab (Telecom Italia labs). Telecom Italia has conceived and developed JADE, and originated the open source community in February 2000. In March 2003 Motorola and Telecom Italia created the JADE Governing Board organization with the mission of promoting the evolution and the adoption of JADE by the mobile telecommunications
industry as a java-based de-facto standard middleware for agent-based applications in the mobile personal communication sector.

Advantages of using JADE as the platform for MAS are:

1. JADE is a completely distributed middleware system with a flexible infrastructure allowing easy extension with add-on modules.
2. The agent platform can be distributed on several hosts. Only one Java application, and therefore only one Java Virtual Machine (JVM), is executed on each host. Each JVM is basically a container of agents that provides a complete run time environment for agent execution and allows several agents to concurrently execute on the same host. The distributed nature is again essential considering the highly distributed nature of our cellular networks.
3. Since JADE supports multiple containers and multiple agents within each container, it can help achieve high degree of parallelism.
4. The communication architecture offers flexible and efficient messaging, where JADE creates and manages a queue of incoming Agent Communication Language (ACL) messages, private to each agent; agents can access their queue via a combination of several modes: blocking, polling, timeout and pattern matching based. The full FIPA communication model has been implemented and its components have been clearly made distinct and fully integrated: interaction protocols, envelope, ACL, content languages, encoding schemes, ontologies and, finally, transport protocols. Since the autonomous nature of the agent needs to preserved and at the same time agent communication through FIPA compliant message performatives need to be used, JADE was the most suited and preferred platform.
2.5.4.1 JADE Architecture

The JADE Agent Platform complies with FIPA97 [14, 91] specifications and include components such as Agent Communication Channel (ACC), the Agent Management System (AMS), and the Directory Facilitator (DF) that manage the platform. All agent communication is performed through message passing, where FIPA ACL is the language to represent messages. The JADE architecture is shown in the figure 2.11.

![Figure 2.11: JADE Architecture](image)

JADE creates multiple containers for agents, each of which can be on the same computing system or different systems. The software architecture is based on the coexistence of several JVM’s and communication relies on Java Remote Method Invocation (RMI) between different VMs and event signaling within a single VM. Each VM is a basic container of agents that provides a complete run time environment for agent execution and allows several agents to concurrently execute on the same host. The agent container controls the life cycle of agents by creating, suspending, resuming and
killing them. Besides, it deals with all the communication aspects by dispatching incoming ACL messages, routing them according to the destination field (receiver) and putting them into private agent message queues; for outgoing messages, instead, the agent container maintains enough information to look up receiver agent location and choose a suitable transport to forward the ACL message.

A complete Agent Platform (AP) is composed of several agent containers as shown in Figure 2.11. Each platform must have a ‘Main Container’ which holds two specialized agents called the AMS agent and the DF agent. The AMS agent is the authority in the platform. It is the only agent that can create and kill other agents, kill containers, and shut down the platform. The DF agent advertises the services of agents in the platform so other agents requiring those services can find them. Distribution of containers across a computer network is allowed, provided that RMI communication between their hosts is preserved.

2.5.4.2 JADE Messages and Performatives

JADE follows FIPA standards so that ideally JADE agents could interact with agents written in other languages and running on other platforms through the exchange of messages. In JADE, messages adhere strictly to the ACL standard which allows several possibilities for the encoding of the actual content. In particular, JADE supports FIPA's semantic language. Message routing is handled by platform and it does not follow a client server communication model. To facilitate communication all agents have a mail box which is used for sending and receiving messages, also a list of pending messages is maintained. All communications are asynchronous. The ACL message includes the following:

1. Sender
2. Receiver(s)
3. Communication Intention (i.e. Performative)
4. Message Content
5. Language used to express the message
6. The Ontology in use
7. A number of optional fields to help keep track of conversations and their current state
   - Conversation ID
   - Reply-with (Performative)
   - In-reply-to
   - Reply-by

JADE provides ‘Get’ and ‘Set’ methods to access all the attributes of JADE ACL message. To create a message, one has to indicate it’s performative in ACL language and set the content.

   Eg. ACLMessage msg = new ACLMessage (ACLMessage.INFORM );
      
      msg.setContent("Hello World");

   - INFORM whereby one agent gives some useful information.
   - QUERY to ask a question, REQUEST to ask the other to do something and PROPOSE to start bargaining. Performatives for answers include AGREE or REFUSE.
   - REQUEST: This performative is used for an agent to know information about the other agent.
   - INFORM: Through this performative the sender informs the desired receiver if the concerned proposition is true.
   - REFUSE: This performative is used by an agent when it is refusing to perform some desired action also explaining the reason for refusal.
   - REJECT PROPOSAL: This performative is used by an agent to refuse any proposal.
   - CONFIRM: This performative is used in case of an affirmative answer after a series of transactions based on negotiations.
• AGREE: This performative is used by any of the cells to agree to some pre-decided format to possibly use in the future.

• CANCEL: This performative is used by any of the agent to cancel an action which was previously agreed upon.

• CALL FOR PROPOSAL: This performative is used for calling of proposals to perform a given action in scenarios where negotiation theory of bargaining is applicable.

• REQUEST WHEN: This performative is used by the sender to elicit a reaction from the receiver on some given condition.

• PROPOSE: This performative is used by any of the agent wishing to submit a proposal to perform a certain action, given certain preconditions.

2.5.4.3 JADE Sniffer

The JADE Sniffer is a pure Java application created for tracking messages exchanged in a JADE based environment. The Sniffer is completely integrated in JADE environment and is particularly useful when debugging agent’s behaviour. The Sniffer is basically a FIPA-compliant Agent with sniffing features. When the user decides to sniff an agent or a group of agents, every message directed to that agent/group or coming from that agent/group is tracked and displayed in the Sniffer GUI. The user can view every message, save it to disk as a text file or serialize it as a binary file for later usage. When the user starts the Sniffer, a new instance of the class jade.tools.sniffer.Sniffer is created. This class extends jade.core.Agent, therefore the Sniffer is like any other agent; interaction with JADE environment and with the Sniffer GUI are the main tasks of this class. In order to let the user choose which agents are to be sniffed, the Sniffer must be constantly informed about birth of agents, death of agents, created and deleted containers. For security's sake this information is not available to a common agent. Hence to get this information, the Sniffer registers itself as a Remote Monitoring Agent (RMA) on the Agent Management System (AMS).
2.6 Chapter Summary

Quality of Service (QoS) provisioning in wireless networks is a challenging problem due to the scarcity of wireless resources, i.e. radio channels, and the mobility of users. The chapter explains the cellular concepts and call admission control which is a key element in the provision of guaranteed QoS in wireless networks. The design of call admission control algorithms for mobile cellular networks is especially challenging given the limited and highly variable resources, and the mobility of users encountered in such networks. A survey of admission control schemes for cellular networks and the research in this area is presented with the goal of providing a broad classification of existing call admission control schemes. These schemes were classified based on factors such as deterministic/stochastic guarantees, distributed/local control and their adaptive nature to traffic conditions.

The concept of an intelligent agent to work on behalf of the user and satisfy the program goals is now well established. The software entities or agents are fed with a certain goal and a set of capabilities (BDI agent). Using knowledge gained from its environment the agent works its way towards its goals under certain constraints imposed by its work environment. This concept of intelligent agents has opened a new vista in the world of computing. The use of agents in distributed systems is now gaining ground as the options available to the developer in such a system is numerous. Plenty of agent development frameworks have been proposed that enable creation, maintenance and destruction of agents. These platforms allow developers to empower the agents with intelligence that helps improve performance. JADE is an open source framework and has been well tested and is robust. Thus the proposed the MAS is simulated using JADE.