A Agent Learning Model For Service Composition In Mobile Cloud Computing

Nazeer Unnisa Nazima, B. Hima Bindu, M. Lakshmi Madhuri
[1]Assistant Professor, Department Of Cse, Bhoj Reddy Engineering College For Women, India.
[2,3]B. Tech Students, Department Of Cse, Bhoj Reddy Engineering College For Women, India.

ABSTRACT
Mobile cloud computing has challenges like limited resources, openness, and uncertainty, affecting service quality and security. This paper presents a three-layer trust-based service composition model using fuzzy evaluation to manage trust. It includes a learning module to understand user preferences, even with incomplete service requests. A prototype built on the JADE platform shows improved success rates and user satisfaction. Mobile cloud computing (MCC) offers significant opportunities for delivering scalable and flexible services, but it faces critical challenges, including limited computational resources, the open and dynamic nature of mobile environments, and inherent uncertainties that impact service quality, reliability, and security. To address these issues, this paper proposes a novel three-layer trust-based service composition model that leverages fuzzy logic-based evaluation to effectively manage trust in MCC systems. The proposed model consists of three distinct layers: a service discovery layer that identifies and filters available services based on functional and non-functional requirements; a trust evaluation layer that employs fuzzy logic to assess the trustworthiness of services by considering multiple trust attributes, such as reliability, security, and performance; and a service composition layer that optimally combines services to meet user demands while maximizing trust and quality of service (QoS). A key feature of the model is its integrated learning module, which utilizes machine learning techniques to analyse historical user interactions and infer user preferences, even in cases of incomplete or ambiguous service requests. This enhances the model's ability to deliver personalized and context-aware service compositions. To validate the proposed approach, a prototype was developed using the JADE (Java Agent Development Framework) platform, enabling agent-based service interactions in a simulated MCC environment. Experimental results demonstrate that the proposed model significantly improves the success rate of service composition, enhances user satisfaction, and outperforms existing approaches in terms of trust management and adaptability to dynamic mobile cloud environments. The model's ability to handle uncertainty and resource constraints makes it a promising solution for advancing the reliability and efficiency of MCC systems.

1. INTRODUCTION

Mobile cloud computing faces challenges like limited resources, uncertainty, and security risks. A Trust-Based Learning Agent model is introduced to improve service selection and composition. The model uses trust scores and learns user preferences, even with incomplete requests. It improves service quality, user satisfaction, and system performance.

Mobile cloud computing (MCC) has emerged as a transformative paradigm, enabling resource-constrained mobile devices to leverage the computational power and storage capabilities of cloud infrastructure. By offloading complex tasks to the cloud, MCC facilitates the delivery of sophisticated services, such as real-time data processing, multimedia streaming, and context-aware applications. However, the dynamic and open nature of MCC environments introduces significant challenges, including limited device resources, fluctuating network conditions, service heterogeneity, and security concerns. These factors often lead to uncertainties that degrade service quality and user satisfaction. Effective service composition, which involves selecting and combining multiple services to meet user requirements, is critical to overcoming these challenges and ensuring reliable, high-quality service delivery.

Traditional service composition approaches in MCC often rely on static or predefined criteria, which struggle to adapt to the dynamic nature of mobile environments and the evolving preferences of users. Moreover, incomplete or ambiguous service requests from users further complicate the composition process, as systems must infer user intent while maintaining trust and optimizing resource utilization. To address these limitations, this project introduces an innovative **Agent Learning Model for Service Composition in Mobile Cloud Computing**. The proposed model employs intelligent software agents equipped with machine learning. capabilities to dynamically compose services in a trust-aware and user-centric manner. By integrating a three-layer architecture—comprising service discovery, trust evaluation using fuzzy logic, and adaptive service composition—the model ensures robust handling of uncertainties and resource constraints.

Existing System

Mobile cloud computing has limitations like low resources, openness, and uncertainty, which affect service quality and security. Complex service needs require a reliable and efficient way to combine (compose) different services. User preference learning is important to improve how well the system meets user needs.

Proposed System

A CP-ABE (Ciphertext-Policy Attribute-Based Encryption) method is used to make data access secure and efficient. The system keeps the size of public data and the effort needed to decrypt fixed, making it faster. It uses a strong encryption method (dual system encryption) to ensure high security under standard rules.

2. REQUIREMENT ANALYSIS

Functional Requirements

Functional requirements define the specific features and capabilities the system must provide to meet user needs. The system involves interactions among Provider Agents, User Agents, Broker Agents, and the Cloud Server, with a focus on trust-based service composition and user preference learning.

1. Provider Agent Module:

Registration: The system must allow Provider Agents to register with the application, providing details such as name, email, mobile, address, username, password, and location.

Authorization: The system must enable the Broker Agent to authorize registered Provider Agents before they can access system functionalities.

File Upload and Management: Provider Agents must be able to upload files to the Cloud Server and view their uploaded files.

File Verification: The system must provide functionality for Provider Agents to verify the safety of uploaded files.

File Recovery: Provider Agents must have the ability to retrieve and secure files that have been attacked or compromised.

Profile Management: Provider Agents must be able to view and update their profile information.

Logout: The system must allow Provider Agents to securely log out.

2. User Agent Module:

Registration: The system must allow User Agents to register with the application, providing details such as name, email, mobile, address, username, password, and location.

Authorization: The system must require Broker Agent authorization for User Agents to access the system.

File Search: User Agents must be able to search for files using queries.

Secret Key Request: The system must allow User Agents to request secret keys for accessing secure files.

File Response and Download: User Agents must be able to view file responses and download files from the Cloud Server.

Profile Management: User Agents must be able to view and manage their profile information.

Logout: The system must provide a secure logout feature for User Agents.

Non-Functional Requirements

Non-functional requirements specify the system's performance, usability, scalability, and other quality attributes. These are derived from the project's goals of improving trust, performance, and user satisfaction in mobile cloud computing.

1. Performance:

- The system must process service composition requests within 1 second to ensure real-time responsiveness for mobile users.
- The trust evaluation process using fuzzy logic must complete within 500 milliseconds per service to minimize delays.
- The system must achieve a throughput of at least 1000 service composition requests per hour to handle high demand.

2. Scalability:

- The system must support up to 10,000 concurrent users (Provider Agents, User Agents, and Broker Agents) without significant performance degradation.
- The system must scale to handle an increasing number of services (e.g., 1000+ services) in the service discovery layer.

3. Usability:

- The user interface must be intuitive and user-friendly, accessible on both desktop and mobile devices.
- The system must support multilingual interfaces to cater to diverse users in different regions.
- The registration, login, and file management processes must require minimal steps (e.g., no more than three steps for file upload or search).

4. Accuracy:

- The trust evaluation module must achieve at least 95% accuracy in identifying trustworthy services based on fuzzy logic assessments.
- The learning module must correctly infer user preferences with at least 90% accuracy for incomplete or ambiguous requests.

5. Security:

- The system must implement CP-ABE (Ciphertext-Policy Attribute-Based Encryption) to ensure secure data access and protect user privacy.
- The system must maintain constant public data size and decryption effort to optimize performance without compromising security.
- All communications between agents and the Cloud Server must be encrypted using secure protocols (e.g., TLS/SSL).

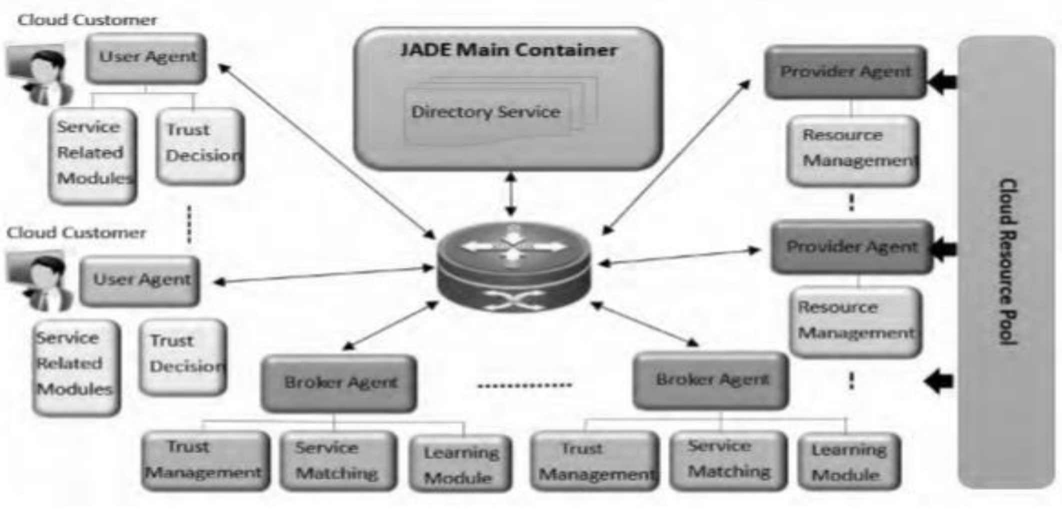
6.Portability:

- The system must be deployable on various cloud platforms compatible with the JADE framework.
- The system must support cross-platform access for users on different mobile operating systems (e.g., Android, iOS).

3. ARCHITECTURE

System Architecture

Project architecture represents number of components we are using as a part of our project and the flow of request processing i.e. what components in processing the request and in which order. An architecture description is a formal description and representation of a system organized in a way that supports reasoning about the structure of the system.



4 IMPLEMENTATION

1.PROVIDER AGENT

The Provider Agent first registers in the application and must be authorized by the Agent Broker. Once approved, the agent can upload and view files. They can also verify if the uploaded files are safe. If any file is attacked, the Provider Agent has the ability to retrieve and secure the file to make it safe again**.**

2. CLOUD SERVER

The cloud server can directly login with the application. Then the cloud server can view files and can check the transactions and also check the attackers.

3.USER AGENT

- The user agent should register with the application and the user should authorize by the agent broker then only the user can able to login.
- Then the user can check his profile, can search file by the query and request secret key, view file response, download the files.

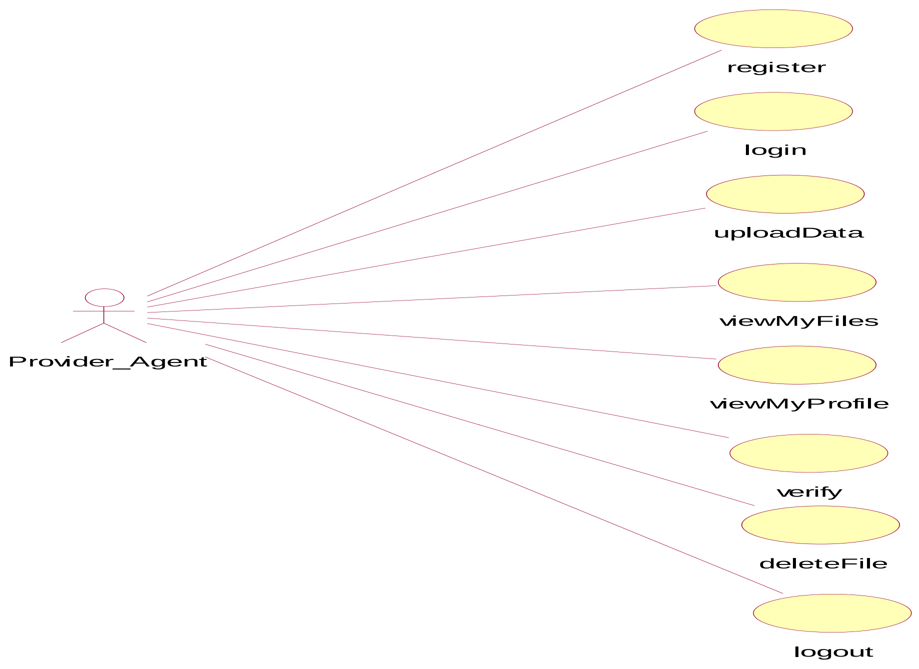
4.BROKER AGENT

- The broker agent can access his home page directly without the register.

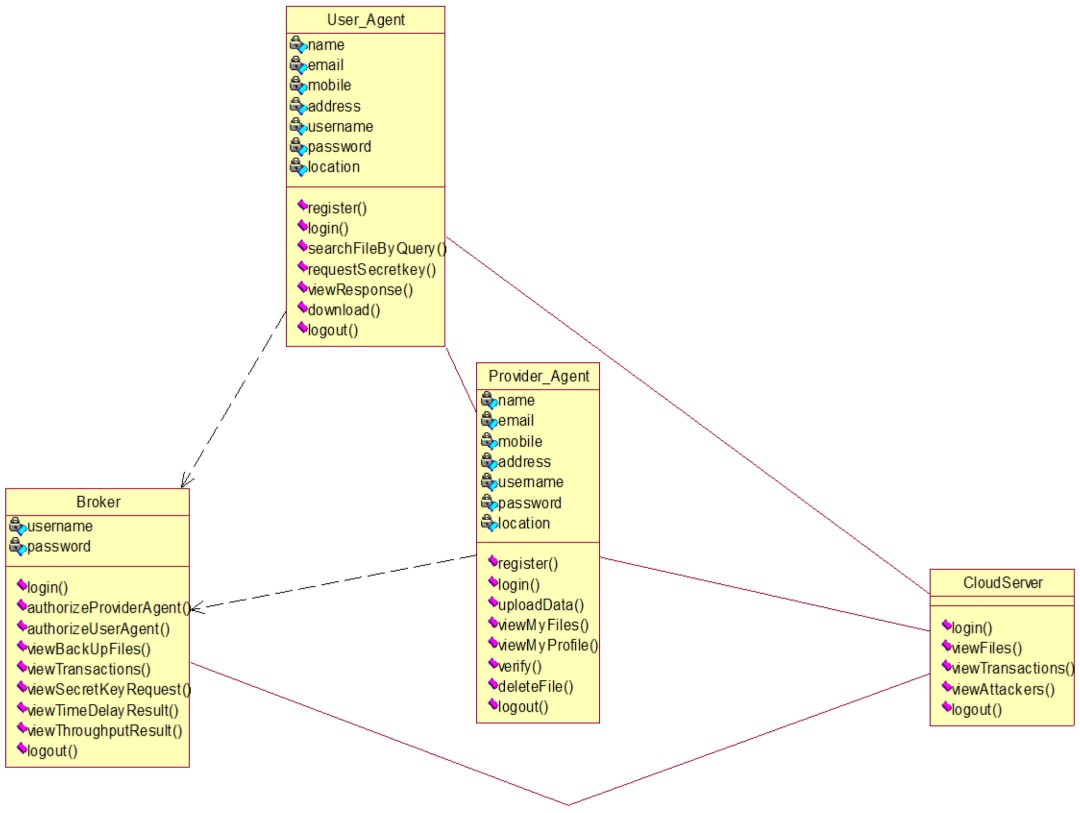
- The broker can authorize the user and also authorize the provider then the broker can view Buck up files, view transactions, view secret request, view time delay result, view through put result.

UML DIAGRAMS

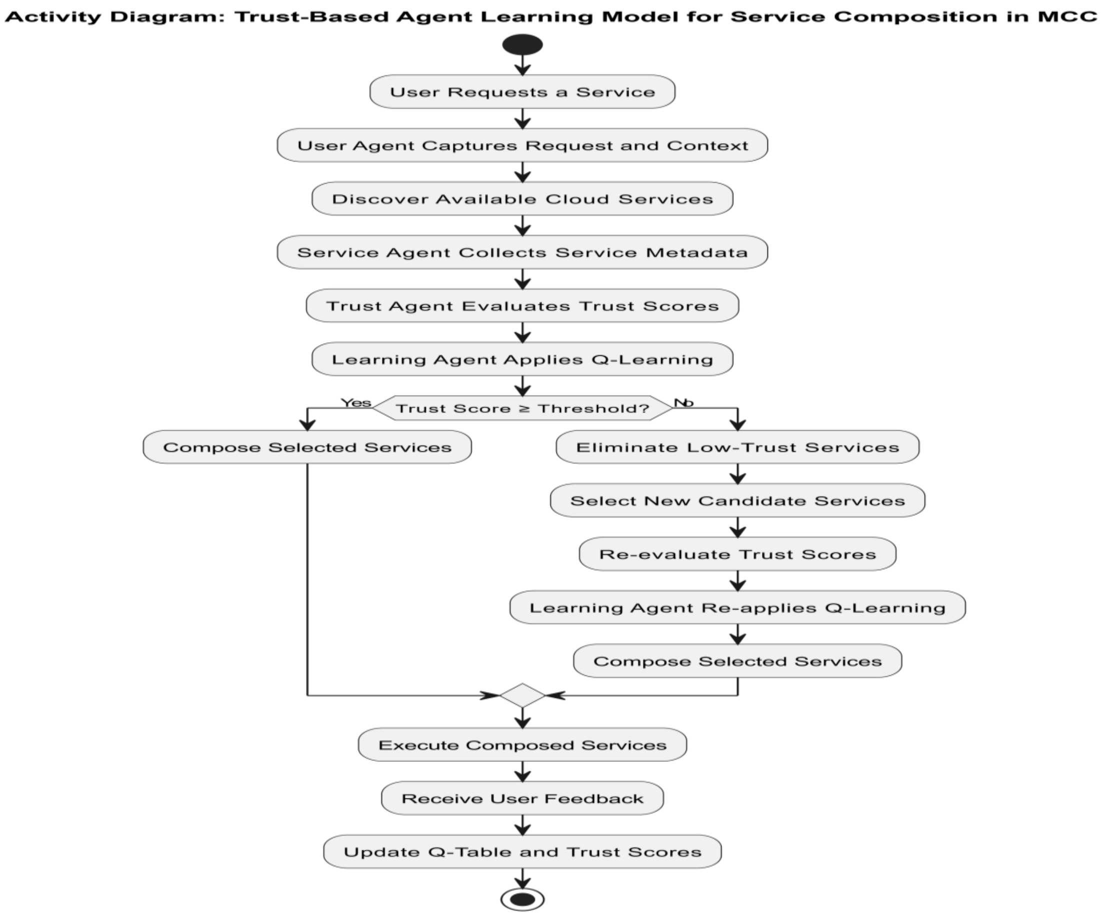
USECASE DIAGRAM



Class Diagram



Activity Diagram


Activity Diagram: Trust-Based Agent Learning Model for Service Composition in MCC
User Requests a Service
User Agent Captures Request and Context
Discover Available Cloud Services
Service Agent Collects Service Metadata
Trust Agent Evaluates Trust Scores
Learning Agent Applies Q-Learning
Yes
Trust Score ≥ Threshold?
No
Compose Selected Services
Eliminate Low-Trust Services
Select New Candidate Services
Re-evaluate Trust Scores
Learning Agent Re-applies Q-Learning
Compose Selected Services
Execute Composed Services
Receive User Feedback
Update Q-Table and Trust Scores

5. TESTING

Software testing is a process, to evaluate the functionality of a software application with an intent to find whether the developed software met the specified requirements or not and to identify the defects to ensure that the product is defect free in order to produce the quality product.

Some of the reasons why software testing becomes very significant and integral part in the field of information technology are as follows.

1.Cost effectiveness

2.Customer Satisfaction

3.Security

4.Maintainability

Dimensions of Testing

There are many different dimensions to consider:

1.Layers of the application (database, APIs, UI)

2.Scale of testing (unit, module, integration, scenario)

3.Type of testing (functional, performance, security, etc.)

4.Methodology (exploratory, scripted manual, automated)

Stages of Testing

Unit Testing

During This first round of testing, the program is submitted to assessments that focus on specific units or components of the software to determine whether each one is fully functional. In this phase, a unit can refer to a function, individual program or even a procedure, and White box testing method is usually used to get the job done. One of the biggest benefits of this testing phase is that it can be run every time a piece of code is changed, allowing issues to be resolved as quickly as possible. It quite common for software developers to perform unit tests before delivering software to testers for formal testing.

Integration Testing

Integration testing allows individuals the opportunity to combine all of the units within a program and test them as a group. This testing level is designed to find interface defects between the modules/functions. This is particularly beneficial because it determines how efficiently the units are running together. Keep in mind that no matter how efficiently each unit is running, if they properly integrated, it will affect the functionality of the software program. In order to run these types of tests, individuals can make use of various testing methods, but the specific method that will be used to get the job done will depend greatly on the way in which the units are defined.

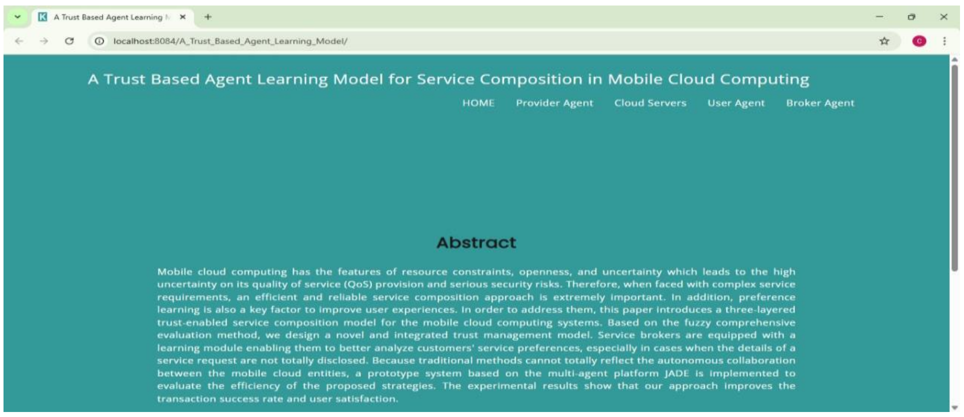
System Testing

System testing is the first level in which the complete application is tested as a whole. The goal at this level is to evaluate whether the system has complied with all of the outlined requirements and to see that it meets Quality Standards. System testing is undertaken by independent testers who haven't played a role in developing the program. This testing is performed in an environment that closely mirrors production. System Testing is very important because it verifies that the application meets the technical, functional, and business requirements that were set by the customer.

Acceptance Testing

The final level, Acceptance testing (or User Acceptance Testing), is conducted to determine whether the system is ready for release. During the Software development life cycle, requirements changes can sometimes be misinterpreted in a fashion that does not meet the intended needs of the users. During this final phase, the user will test the system to find out whether the application meets their business needs. Once this process has been completed and the software has passed, the program will then be delivered to production. The extensiveness of these tests is just another reason why bringing software testers in early is important. When a program is more thoroughly tested, a greater number of bugs will be detected; this ultimately results in higher quality software.

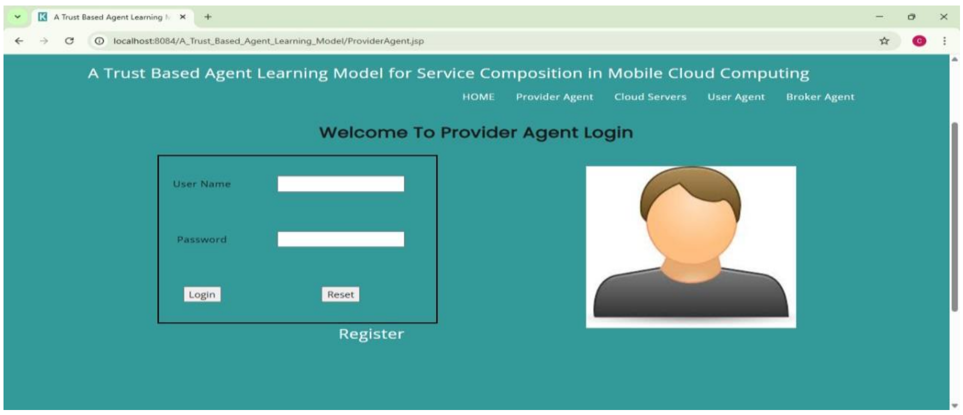
6.RESULTS

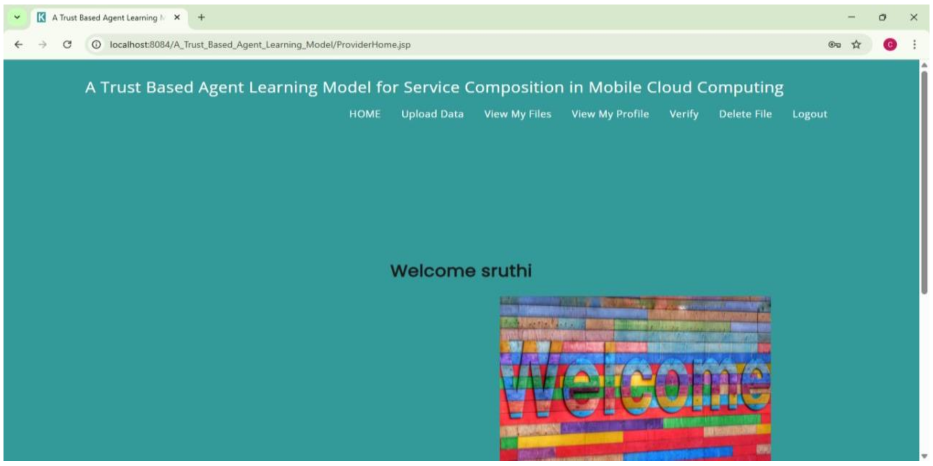

A Trust Based Agent Learning M
localhost:8084/A_Trust_Based_Agent_Learning_Model/
HOME
Provider Agent
Cloud Servers
User Agent
Broker Agent

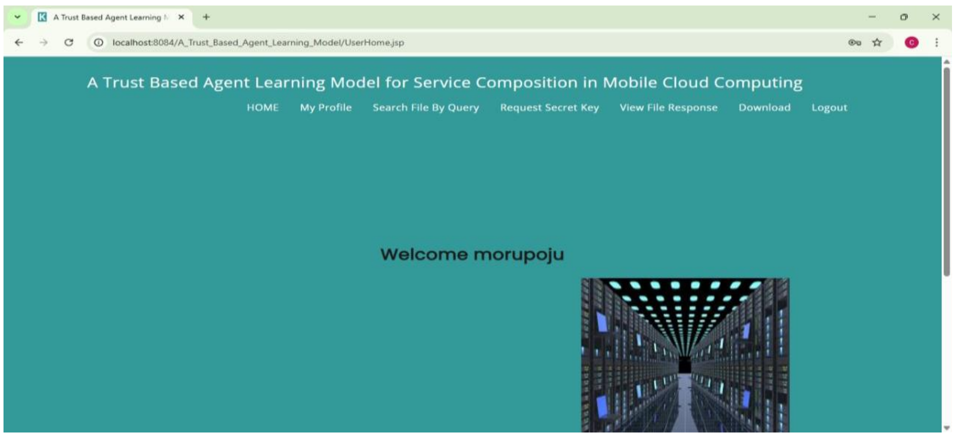
A Trust Based Agent Learning Model for Service Composition in Mobile Cloud Computing

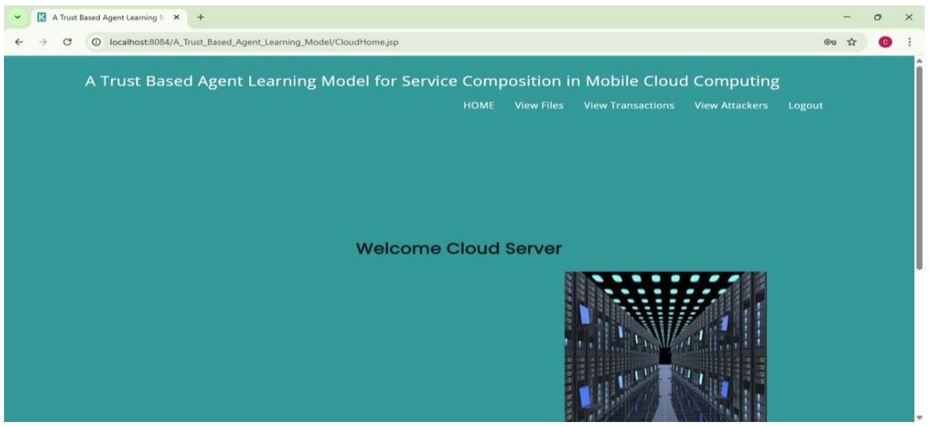
Abstract

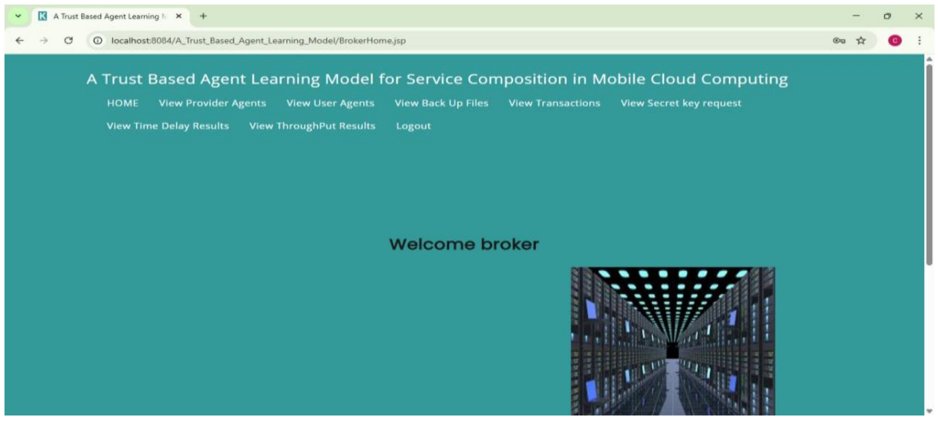
Mobile cloud computing has the features of resource constraints, openness, and uncertainty which leads to the high uncertainty on its quality of service (QoS) provision and serious security risks. Therefore, when faced with complex service requirements, an efficient and reliable service composition approach is extremely important. In addition, preference learning is also a key factor to improve user experiences. In order to address them, this paper introduces a three-layered trust-enabled service composition model for the mobile cloud computing systems. Based on the fuzzy comprehensive evaluation method, we design a novel and integrated trust management model. Service brokers are equipped with a learning module enabling them to better analyze customers' service preferences, especially in cases when the details of a service request are not totally disclosed. Because traditional methods cannot totally reflect the autonomous collaboration between the mobile cloud entities, a prototype system based on the multi-agent platform JADE is implemented to evaluate the efficiency of the proposed strategies. The experimental results show that our approach improves the transaction success rate and user satisfaction.


A Trust Based Agent Learning
localhost:8084/A_Trust_Based_Agent_Learning_Model/ProviderAgent.jsp
A Trust Based Agent Learning Model for Service Composition in Mobile Cloud Computing
HOME
Provider Agent
Cloud Servers
User Agent
Broker Agent
Welcome To Provider Agent Login
User Name
Password
Login
Reset
Register


A Trust Based Agent Learning
localhost:8084/A_Trust_Based_Agent_Learning_Model/ProviderHome.jsp
A Trust Based Agent Learning Model for Service Composition in Mobile Cloud Computing
HOME
Upload Data
View My Files
View My Profile
Verify
Delete File
Logout
Welcome sruthi
Welcome


A Trust Based Agent Learning
localhost:8084/A_Trust_Based_Agent_Learning_Model/UserHome.jsp
A Trust Based Agent Learning Model for Service Composition in Mobile Cloud Computing
HOME
My Profile
Search File By Query
Request Secret Key
View File Response
Download
Logout
Welcome morupoju


A Trust Based Agent Learning M
localhost:8084/A_Trust_Based_Agent_Learning_Model/CloudHome.jsp
A Trust Based Agent Learning Model for Service Composition in Mobile Cloud Computing
HOME
View Files
View Transactions
View Attackers
Logout
Welcome Cloud Server


A Trust Based Agent Learning M
localhost:8084/A_Trust_Based_Agent_Learning_Model/BrokerHome.jsp
A Trust Based Agent Learning Model for Service Composition in Mobile Cloud Computing
HOME
View Provider Agents
View User Agents
View Back Up Files
View Transactions
View Secret key request
View Time Delay Results
View ThroughPut Results
Logout
Welcome broker

7.CONCLUSION & FUTURE SCOPE

Conclusion

- The paper introduces a new system is trust-based service composition model for mobile cloud environments (TALMSC), that helps mobile cloud users find the best services using brokers who use trust and learning methods to make better matches.
- Tests show the system improves user satisfaction, and future work will focus on better combining trust with other features and understanding how brokers can adapt as the market changes**.**

Future Scope

- Integration of advanced machine learning algorithms for accurate predictive crime trend analysis, and it is expansion to include a wider range of crime categories beyond murder.
- Implication of live data updates from law enforcement for timely analysis and addition of interactive tools like heatmaps and geographic crime mapping.
- Development of a mobile-friendly interface for wider accessibility.
- Adapting to technological advancements for improved urban safely planning and user engagement

REFERENCES

1] T. Noora, S. Zeadaliyb, A. Alfazic, and Q. Sheng," Mobile cloud computing: Challenges. and future research directions," Journal of Network and Computer Applications, vol. 115, pp.70-85, Aug. 2018

2] C. euang, X. Mei, G. Zhao, and J. Wu et al, Transaction modelling and execution analysis of uncertainty composition service in mobility computing environments," Science China:

Information Science, vol. 45, no. 1, pp. 70-96, Jan. 2015.

3] S. Deng, L. Huang, H. Wu, W. Tan, J. Taheri, A. Zomaya, and Z. Wu. Toward Mobile Service Computing: Opportunities and Challenges," IEEE Cloud Computing, vol. 3, no.4, pp32-41, 2016.