MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE ABBES LAGHROUR KHENCHELA
FACULTE DES SCIENCES ET DE LA TECHNOLOGIE

Département des Mathématiques et de l'Informatique

N° de série : 04

Mémoire de fin d'études

Pour l'obtention du diplôme de Master (L.M.D)

Spécialité : Informatique
Option : Sécurité et Technologies Web

Proposing a novel SaaS contracts negotiation model.

Réalisé par *: -KHELLAF Douaae*
- KENATEF Hana

Dirigé par *: Dr. HIOUAL Ouassila*

Soutenu le 26/06/2018

DEDICATION 1

To my beloved late mother

She taught me to persevere and prepared me to face the challenges with faith and humility. She was constant source of inspiration to my life. Although she is not here to give me strength and support I always feel her presence that used to urge me to strive to achieve my goals in life.

To my dear father

Who always had confidence in me and offered me encouragement and support in all my endeavors.

To my lovely sister Romaissa and my dear brothers Hassan and Oualaddin

who stand by me when things look bleak, believe in me and surround me with their care and love and to whom I wish successful lives.

To my dear aunt Barko

to my supervisor and role model Dr. Hioual Ouassila

To my dear partner Hana

To all the members of my beautiful family

My grandma Mebarka, my late grandma Nouna , my grandpa Mohand, my cousin Sifeddine, my aunts, my uncles and my cousins .

To all my friends

Especially Sara, Afraa, Sara and their family, Amina , Dunia, Jahida Bouchra and Khadidja.

I dedicate this research

.

Douaae

DEDICATION 2

As we take our last steps in university life,

For all those who planted optimism in our path.

My parents, who had the greatest credit after Allah in helping me to reach my goal.

For each one we have benefited from since the first year till now.

All words of thanks can not be summed up to explain our gratitude, our appreciation.

To my dear partner Douaae .

To all my family my grandparents .

my brother and my dear sisters .

my uncles my aunts and their families to all my cousins .

To my dear companion Ayoub .

To all my friends , classmates who have estimated me .

To all of you i dedicate this work.

Hana

ACKNOWLEDGMENTS

At the end of this work, we first want to thank Allah for guiding us and giving strength, courage and patience to finish this work :

Thank you "ALLAH".

Through this humble work, we would like to warmly thank our supervisor,Dr.Hioual Ouassila for the assistance, advises, encouragement and support that we have been given throughout the process of making this Master's thesis, and her patience and availability that made us love our work.

At the same time we would like to thank all the people who helped us with their recommendations and moral supports.

Furthermore, we would like to warmly thank the jury members who have done the honour to accept to participate in our defence of this humble work.

Douaae and Hana

ABSTRACT

ملخص

البرمجيات كخدمة (SaaS) هي نموذج خدمة سحابية يحظى بالكثير من الاهتمام في سوق الحوسبة السحابية. وهو يتيح لمقدمي الخدمات تقديم التطبيقات كخدمة للعملاء عند الطلب عبر الإنترنت. نظرا للعدد المتزايد من العملاء وموفري الخدمات السحابية ، فقد أصبح من المستحيل تقريبا تسهيل المفاوضات وجها لوجه بشأن عقدSaaS. وبالتالي ، هناك حاجة إلى التفاوض الآلي للتوصل إلى اتفاق بين مقدمي الخدمات والمستهلكين دون معرفة سابقة لبعضهم البعض.

الهدف من هذا العمل هو اقتراح نموذج للتفاوض بشأن عقود (SaaS). تم تصميم نموذجنا آليا من خلال استخدام وكلاء أذكياء يمثلون العملاء ومقدمي الخدمات في بيئة نظام متعدد الوكلاء. تتفاعل هذه العوامل باتباع بروتوكول التفاوض CNP يتكون نموذجنا من ثلاث مراحل: الاكتشاف والاختيار والتفاوض. يستند التقييم اللازم لاتخاذ قرار إلى كل من مشكلة تقييد الرضا(CSP) ونظرية المنفعة متعددة المعايير (MAUT) . أخيرا ، ننفذ أعمالنا باستخدام منصة (JADE) التي تعتمد على لغة برمجة Java.

الكلمات الرئيسية : الحوسبة السحابية ، عقد (SaaS) ، التفاوض ، بروتوكول التفاوض CNP، الوكلاء الأذكياء ، نظام الوكلاء المتعددين ،نظرية المنفعة متعددة المعايير(MAUT) ، مشكلة تقييد الرضا (CSP) ، JADE.

Abstract

Software as a service (SaaS) is a cloud service model that wins a lot of attention in the cloud computing market. It allows providers to offer applications as a service to customers on-demand over the internet. Due to the increasing number of cloud customers and providers, it is becoming almost impossible to facilitate face to face negotiations over the SaaS contract. Thus, automated negotiation is needed to reach an accord between service providers and consumers with no previous knowledge of each other.
The objective of this work is to propose, a SaaS contract negotiation model. Our model is automated through the use of intelligent agents representing the customers and providers in a multi agent system environment. These agents interact by following CNP negotiation protocol. Our model is composed of three phases: discovery, selection and negotiation. The evaluation needed to make a decision is based on both Constraint Satisfaction Problem (CSP) and Multi-Attribute Utility Theory (MAUT).Finally, we implement our work using the JADE platform which is based on the Java programming language.

Key words : Cloud computing, SaaS contract, Negotiation, CNP Negotiation protocol, Intelligent agents, Multi agent system, Multi-Attribute Utility Theory (MAUT), Constraint Satisfaction Problem (CSP), JADE.

Résumé

Le logiciel en tant que service (SaaS) est un type de services Cloud qui suscite beaucoup d'attention sur le marché du Cloud Computing. Il permet aux fournisseurs de proposer, aux clients, des applications sous forme de service

qui peuvent les demander via Internet. En raison du nombre croissant des clients et des fournisseurs Cloud, il devient presque impossible de faciliter la négociation, en face à face, des contrats SaaS. Une négociation automatisée est donc nécessaire pour parvenir à un accord entre les fournisseurs et les consommateurs de services Cloud n'ayant aucune connaissance préalable les uns des autres.

L'objectif de ce travail est de proposer un modèle de négociation des contrats SaaS. Notre modèle est automatisé en utilisant le paradigme des agents intelligents qui représentent les client et les fournisseurs Cloud, ces agents interagissent dans un environnement multi-agents en suivant le protocole de négociation CNP . Notre modèle est composé de trois phases : la découverte, la sélection et la négociation. Et, l'évaluation nécessaire pour prendre une décision est basée à la fois sur le problème de satisfaction de contrainte (CSP) et sur la théorie de l'utilité multi- attributs (MAUT). Enfin, nous mettons en œuvre notre travail en utilisant la plate-forme JADE qui est basée sur le langage de programmation Java.

Mots clés : : Cloud computing, Contrat SaaS, Négociation, Protocole de négociation CNP, Agents Intelligents, Système Multi-agents, la théorie de l'utilité multi-attributs (MAUT), le problème de satisfaction de contrainte (CSP) , JADE.

Contents

List of Tables

List of Figures

Listings

List of Acronyms

MAS Multi Agent System

CFP Call For Proposal

AI Artificial Intelligence

CNP Contract Net Protocol

MALE Multi-agent Learning Environment

SANP Speech Act Negotiation Protocol

SaaS Software as a Service

PaaS Platform as a Service

IaaS Infrastructure as a Service

HaaS Hardware as a Service

NIST National Institute of Standards and Technology

SLA Service Level Agreement

ENISA The European Network and Information Security Agency

CSC Cloud Service Customer

CSP Cloud Service Provider

CSP Constraint Satisfaction Problem

MAUT Multi-Attribute Utility Theory

GUI Graphic User Interface

UI User Interface

GENERAL INTRODUCTION

Over the years, the way computing is provided has changed from mainframe and terminal to PC, and then networked PC, leading to clients and servers and, lately to cloud computing via the Internet. The difference between the early stages of computing and the more recent cloud computing is that computational resources used to be delivered as product (hardware or software). Now computing is delivered as a service in a more customized way. In the past, the customers needed to adapt to what was available in the market and look for the seller that offered a product that fitted most closely to his/her requirements. Also, in the past the relationship between the seller and buyer ended just after the purchasing deal was made. Now, in the cloud market, the seller has become a provider of computing as an utility and the customer has become a client. Also, the relationship and the negotiation between them will continue until the contract ends. Contracts in cloud computing can take many forms according to the nature of the delivered service , this form mainly depends on the chosen cloud service model; Platform as a Service (PaaS), Infrastructure as a Service(IaaS), or Software as a Service (SaaS).Furthermore, this contract is the legal contract between the provider and the client, in order to ensure that both parties are on the same page concerning the delivered service. In cloud computing the provider and client will negotiate before signing the contract.

Currently there are only a relatively small number of cloud providers in the cloud computer market and all of them offer solely "off-the-shelf" contracts (“ take it or leave it ”). Introducing customized contracts to the cloud market would offer customers and providers, added benefits. Customized contracts can only be established by negotiation. The negotiation needs to be automated to handle the dynamic and complex environment of cloud computing.And to ensure this aspect our solution needs to be based on intelligent agents and the interaction between them needs to be in a suitable environment which is multi-agent system.

Today’s science covers increasingly complex problems and is driven by increasingly powerful

technologies. It is well based on computation, data analysis and collaboration as well as on the efforts of theorists and experimentalists. To this end, the scientific community has developed a wide range of solutions to solve the problem of ever-increasing computing power needs.

Multi-agent systems have become a dominant paradigm in the field of complex distributed systems development thanks to their data distribution and heterogeneity characteristics, decentralized control, asynchronization of treatments and partial vision of each component of these systems [1], also the coordination and communication mechanisms recommended by the agent approach provide satisfactory and elegant solutions.

In this work we propose an automated negotiation model for SaaS contracts in the cloud computing environment, which consists of three phases : discovery, selection, and negotiation. In the discovery phase, the providers of the needed service will be discovered by the client. In the selection phase the candidate providers will be selected based on the client's requests and preferences. In the negotiation phase, the providers and the client will be represented by the intelligent agents, which will negotiate on their behalf and then the client will compare all the outcomes of the negotiation sessions and will choose the best provider to sign the SaaS contract with.

Providers and clients should have the freedom to make their own agents, which will represent them and will perform their negotiation strategy. Developing these intelligent agents is not an easy task, because every negotiation sessions can be different than the other. Also, the agent needs to be able to decide when to accept the opponent's offer, in the same time to keep track of the counter in the negotiation session, and to decide when to end the negotiation without an agreement.

An intelligent agent is expected to be self-interested. The Agents that we will use in this work are Utility-based agents, which mean they use a utility function to make rational decisions. Utility functions are a way of representing agent's preferences with the aim to achieve a goal. The ultimate goal of each agent is to maximize its utility. When two utility-based rational agents try to maximize their utility in the negotiation process, there often occurs a conflict. Here is where the Contract Net Protocol (CNP) might become useful.CNP is a negotiation protocol that works in a multi-agent system environment , and since this protocol is both based on the approaches of game theory and heuristics, we can work with those approaches as well to get

to our goal.Our work deals with a multi-issue negotiation that means that the subject of negotiation is multiple , and since the traditional utility function are concerned with one attribute at a time this leads us to adopt a multi-criteria utility function otherwise known as Multi-Attribute Utility Theory (MAUT)combined with Constraint Satisfaction Problem (CSP)which allows us to express the constraints and preferences through agents.
The rest of this master's thesis is organized as follows :

Chapter 1 : This chapter introduces the reader to a state of art on multi-agent systems .

Chapter 2 : This chapter treats the concept of negotiation, its definitions as well as its different elements, types, approaches and protocols in a multi-agent system environment.

Chapter 3 : In this chapter we present the cloud computing paradigm as well as the SaaS contracts.

Chapter 4 : In this chapter, we introduce the problematic and the objectives of this study. Then, we present in detail, our model.

Chapter 5 : This chapter consists of validating and implementing our model. We first describe the different tools and language used. Then we present the implementation of the different components included in our solution.

Finally, the manuscript ends with a general conclusion that recapitulates the work done and proposes some perspectives.

Chapter 1

AGENTS AND MULTI-AGENT SYSTEMS

1.1 Introduction

Computer engineering has progressed in recent years through the orientation towards an increasingly high abstraction that allows us to model more complex systems. This has led to the emergence of a new computer paradigm named Multi Agent System (MAS).
Multi-agent systems have become a new abstraction approach that can be used for analysis, modelling and development of complex and distributed computer systems. The agent-based vision offers a powerful stock of tools, techniques that have the potential to significantly improve software systems.
Multi-agent systems can be defined as a set of weakly coupled autonomous agents, software components that exploit different techniques of Artificial Intelligence (AI) [2] since its conception.
This chapter intends to familiarize the reader with the key concepts and illustrate the state of art concerning agents and multi-agent systems.

1.2 Agents

1.2.1 Definitions

The concept of an agent like all fundamental concepts is relatively vague. We can distinguish several definitions for this term.

In what follows, some important definitions are presented:

- "An agent is an entity that senses its environment and acts upon it" [3]
- "Most often, when people use the term 'agent' they refer to an entity that functions continuously and autonomously in an environment in which other processes take place and other agents exist." [4]
- " Intelligent agents are software entities that carry out some set of operations on behalf of a user or another program, with some degree of independence or autonomy, and in so doing, employ some knowledge or representation of the user's goals or desires" [5]
- "An agent is a physical or virtual autonomous entity that can act, perceive its environment and communicate with others,and has skills to achieve its goals" [6]

Agents are the physical or virtual entities that are placed in a particular environment and are able to take independent actions according to the feedback received from its surroundings. This ability of agents makes them autonomous and they execute their individual goals at the same time while sharing the common resources in a system.
Figure 1.1 depicts a high-level view of an agent within its environment. An agent receives input from its environment and, through a repertoire of actions available to it, reacts to it in order to modify it.

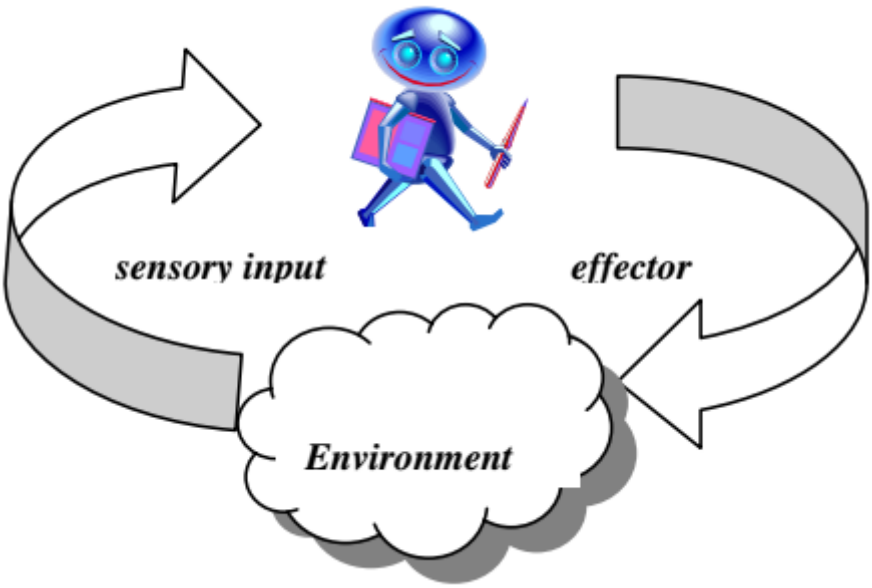

Figure 1.1: Agent interacting with its environment.

1.2.2 Agent Properties

From the definitions that appear in the literature, several properties of agents can be identified:

- **Autonomous :**

  Decision-making about one's behaviour depends solely on one's perceptions, knowledge, and representation of the world (an agent may be dependent and autonomous).

- **Proactive :**

  Generates his goals, takes initiative to meet his goals, not only driven by events.

- **Flexible :**

  Reacts to changes in the environment; adapts to available resources.

- **Social :**

  Able to interact to achieve goals, to help other agents in their activities.

- **Located :**

  Able to perceive the environment and to act in a limited way .

1.2.3 Agents Models

We present below, the main models of agents :
The agents can be classified in general into three types as in Figure 1.2.

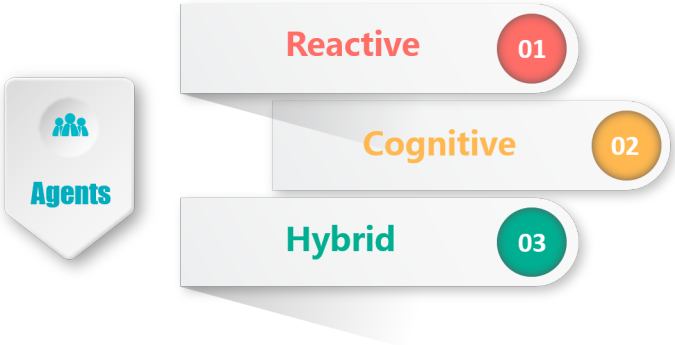

Figure 1.2: General models of agents.

1.2.3.1 Reactive agents

They are such type of agents which possesses no goal at all but still plays an important role in the simulation. These types of agents wait until an instruction is given to accomplish a task and does not get involve in decision making unless it is told to. [7]

1.2.3.2 Cognitive agents

They are complex agents embedded with intentions that take actions by sensing the environment. They are the intelligent blocks of a system that responds to the changes in the environment through learning capabilities. Cognitive agents are autonomous in nature and usually possess adaptive reasoning skill sets. They are capable of decision making and require complex calculation and programming as dynamics is involved. [8]

1.2.3.3 Hybrid agents

Reactive agents may be suitable for some types of problems and less well for others, and this is true for cognitive agents as well .
Chaib [9] [10] [11] [12], Ferguson [13], and Georgef [14] investigated the possibility of combining the two approaches to obtain hybrid architectures.
A hybrid agent architecture is an agent composed of several layers arranged in a hierarchy, that combines and integrates features from both types of agents,and that combines and integrates features that are part of both types of agents. For example, we can build reactive systems with intelligent agents or the opposite (have cognitive systems whose agents have a reactive character to stimuli).
There are several architectures based on a three-layer model: reactive layer, cognitive layers and a layer of communication between agents.

1.3 Multi-agent Systems

1.3.1 Definition

The agent is the main component of multi-agent systems. According to Ferber [6] a multi-agent system (See Figure 1.3) is a system that is composed of the following elements:

- **An environment E :** in other words a space generally having a metric.
- **A set of objects O :** These objects are located, and that means that for any object it is possible, at a given moment, to associate a position in E. These objects are passive; That is, they can be perceived, created, destroyed and modified by agents.
- **A set A of agents :** which are particular objects, which represent the active entities of the system.

- **A set of relations R :** that unite objects (and therefore agents) between them.
- **A set of operations Op :** allowing the agents of A to perceive, produce, consume, transform and manipulate O objects.

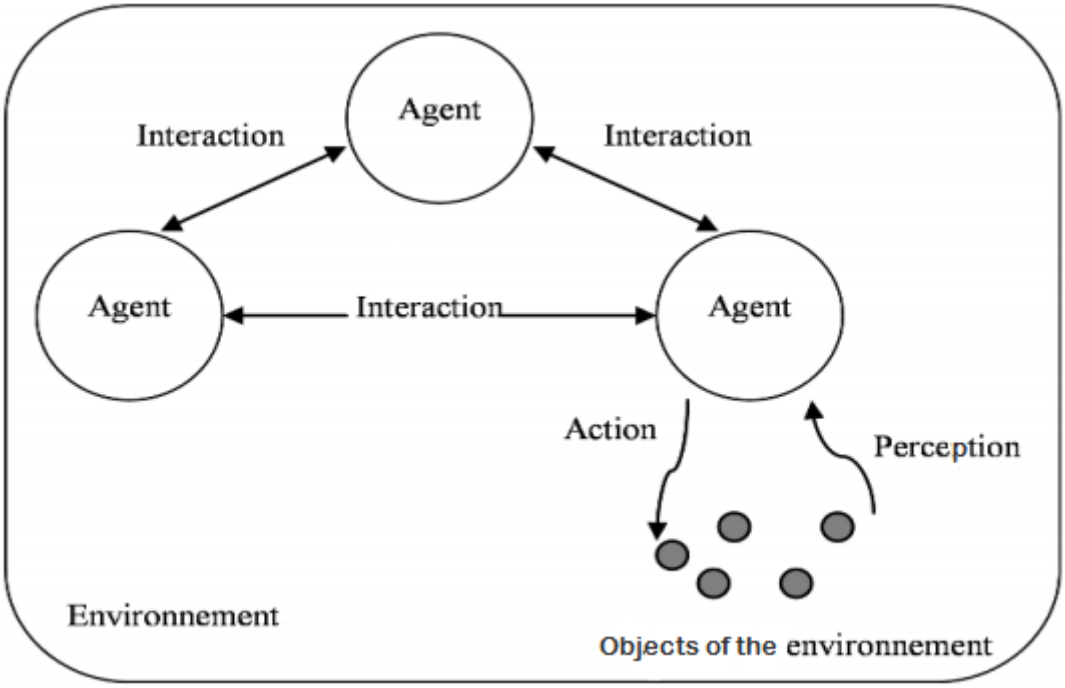

Figure 1.3: Representation of a multi-agent system by Ferber [1].

1.3.2 Multi-agent Systems Characteristics

A MAS has most of the following characteristics [15]:

- Each agent has incomplete information or capabilities for solving the problem and, thus, has a limited viewpoint.
- There is no global control system.
- Data is decentralized.
- Computation is asynchronous.

1.3.3 Multi-agent Systems Applications

The first MAS applications appeared in the mid-1980s and increasingly have been applied to various problems, including market simulation, monitoring, system diagnosis, and remedial actions [16] [17] [18].

- In [19], an approach is developed to prevent interconnected power systems from catastrophic failures. It uses a MAS-based defence system that allows agents to have adaptive decision criteria.

- In [20] [21], multi-agent based models are adopted in computational sociology to study the social life by modelling the interactions between the adaptive agents and the impact of those interactions on a collective organization

- In [22], an integrative and flexible method is proposed that uses agent-based modelling for assessment of market designs.
  Progress in AI, hardware, and sensor technologies have been achieved by the MAS community, resulting in agent technologies that are applied successfully to real-world industrial problems.

MASs are implemented in the real world, in numerous areas which are outlined below:

- **Airline Booking Systems :** This is one of the sound areas where multi-agent systems are increasingly being implemented [23]. The booking system involves human interaction where travel agents across a country book seats for a particular flight. However, the travel agents will need to go through some communication process in order to ensure which seats have been booked and which haven't. This communication can take place over the phone. Now, if we were to simulate the scenario sketched above using multi-agent systems, the agents would have to be reactive in the situation of an airline environment.
  - The travel agents will be simulated using agents.
  - The communication regarding seating information will take place using messages within the network.
  - Each agent will react in accordance to the feedback it receives from the airline booking system

- **Health Care :** This application is an emerging area of multi-agent systems. Routine procedures in hospitals and clinics involve the monitoring of blood pressure, temperature,

heart rate [24], etc. Currently on the wide scale, a patient will be tended by several nurses, who will from time to time, monitor all these vital health aspects. The related issues such as blood pressure and temperature may be discussed amongst nurses and further action may be taken, with direct communication taking place between the nurses. An implementation of the multi-agent System could involve having several agents that monitor the different areas of health of a particular patient. Communication will then take place via messages and the agents can be reactive in critical situations, by informing the related doctors and health care staff.

- **Information Extraction :** As we are all aware of, the web is being used for resource collection worldwide. One example happens to be online dictionaries, where we can search up the definitions of words to make concepts clearer. These are mainly implemented using multi-agent systems where agents throughout the world, holding different resource information are contacted, and all the different ideas and concepts relating to that particular word are presented to the user [25]. For example, the Wikipedia Encyclopedia.

1.3.4 Strengths and Limitations of MASs

The strengths and drawbacks of using multi-agent systems are outlined below:

1.3.4.1 Strengths

➢ *Parallel computation and robustness:* A system's operation could speed up with parallel computation as multiple agents are deployed, each with tasks that are different from each other.

➢ *Flexibility:* Independent tasks and time bounded requirements can be handled easily and efficiently.

➢ *Redundancy:* Fault tolerance can be boosted due to redundant implementation of agents and because the task is shared.

➢ *Granularity:* From a programmer's point of view, the granularity of MAS simulators results in relatively simple programs.

➢ *Scalability and Reusability:* Adding new agent is relatively easy to the systems whose capabilities and functionalities are expected to vary. Agents that are designed to be

independent and have unique characteristics can be reused in other systems which is similar to using a particular library in different frameworks.

➢ *Independency:* Since the repetitive tasks could be automated, MAS simplifies the human-machine interaction level by cooperation between autonomous agents rather than a centralized control.

1.3.4.2 Limitations

➢ Agent based simulations are relatively complex because every interaction between the agents and environment has to be defined each time an agent and environment variables are designed.

➢ A high computational requirement is one of the drawbacks when extensive computation is required with large number of agents involved and with a complex environment.

➢ The development time in a multi-agent system model could be high due to the fact that the components of MAS require a clear demarcation between themselves while designing. This becomes monotonous to the developer of the model.

➢ Insufficient mechanism for representation of organizational structure [26].

1.4 Interaction in a Multi-agent System

The interaction between the agents appears in several modes which are cooperation, coordination and negotiation.

1.4.1 Definition of interaction

A Multi-agent System is a smart system made up of a multitude of agents, each one of them is responsible for achieving its own goals in order to achieve the main goal of the system. The coherence of the system and its intelligence do not come from the intelligence of its agents, but of their interactions.

Jacques Ferber in [6] defines an interaction as : "dynamic linking of two or more agents through a set of reciprocal relationships. The interactions are not only the consequence of actions taken by several agents at the same time, but also the element necessary for the constitution social organizations"

In general, the interactions are implemented by a transfer information between agents or between the environment and agents; by perception or by **communication** (See section 1.4.5). The interaction between the agents appears in several modes, which are **cooperation**, **coordination** and **negotiation**. We have given an overview of these modes in the below sections.

1.4.2 Cooperation

Among the fundamental characteristics of a multi-agent system, we find the distribution of work among the different agents that constitute it, where each one of them is responsible for achieving its own goals, which are a sub-problem of the global problem.

Each agent has a set of skills that allow it to solve different problems, but there are situations where its capabilities and skills are not enough to perform certain tasks (or it does not have necessary means) so it will need the intervention of another agent of the system that will help it solve the problem, in other words, there is cooperation to make the system evolve towards its goals. [27] [28]

Cooperation is therefore to involve several agents to satisfy an individual or common goal.The cooperation situation appears when the actions of each agent satisfy at least one of the following

conditions :

➢ Agents have one goal in common and their actions tend to achieve that goal.

➢ Agents perform actions that achieve not only their own goals but also those of others.

1.4.3 Negotiation

Agents goals in a multi-agent system may be inconsistent and their demands may be contradictory. In a game situation, everyone seeks to be a winner, but achieving this goal for one of the players makes it impossible for the other players to achieve their goal. The resources that agents need may be rare and the use of a resource by one of the agents may prevent another agent from achieving its goal ... etc. These and other situations prevent the desirable progress of the system. so there must be a way that allows each agent to proceed its work: agents must negotiate the solution.

Smith defines the negotiation [29] : "By Negotiation, we mean a discussion in which the interested parties exchange information and come to an agreement. For our purposes negotiation has three important components :

(a) there is a two-way exchange of information,

(b) each party to the negotiation evaluates the information from its own perspective,

(c) final agreement is achieved by mutual selection".

To carry out the negotiation process, it is necessary to follow a protocol that facilitates convergence towards the solution. Negotiation is generally characterized by a minimal protocol of actions which is : propose, evaluate, accept or reject a solution.

1.4.4 Coordination

T. W. Malone defines coordination as "the set of additional activities that need to be performed in a multi-agent environment and that a single agent pursuing the same goals would not accomplish". [30]

The agents to be coordinated must therefore perform additional processing of the information. Coordination becomes necessary during different types of events [31]:

➢ The need for information or a result that only another agent can produce, such as a specialist for example.

➢ The limitation of the resources of the environment, which can generate conflicting situations for example.

➢ Cost optimization, sometimes linked to a limitation of resources, which allows several agents deciding to coordinate to save their efforts to produce the same result.

➢ Agents that have different goals but whose intermediate steps are dependent on objectives or intermediate steps of other agents.

In a system of multiple agents that are performing some activities in a shared environment an important aspect is the method of coordination adopted for the harmonious and coherent functioning of parts to obtain effective results. It is an important issue of research that has been widely investigated in several areas of research

1.4.5 Communication

Communication represents the basis of all the modes of interaction that we have seen before, either negotiation or coordination.

Communication is defined as a form of local action of an agent towards other agents. The questions addressed by a communication model can be summarized by the following interrogation: who communicates, what, to whom, when, why, and how?

• Why do agents communicate? The communication must allow the implementation of the interaction and consequently the cooperation and coordination of actions.

• When do agents communicate? Agents are often confronted with situations where they need to interact with other agents to achieve their local or global goals. The difficulty lies in identifying these situations.

• Who do agents communicate with? Communications can be selective on a small number of agents or broadcast to the set of agents.

- How do agents communicate? The implementation of the communication requires an understandable communication language common to all agents. It is necessary to identify the different types of communication and to define the means allowing not only the sending and receiving of data but also the transfer of knowledge with appropriate semantics to each type of message.

There are essentially two modes of communication between agents [32]:

1. signal communication,through the environment.
2. direct communication by sending messages.

1.5 Conclusion

Our work is situated at the intersection of the three domains: cloud computing, multi-agent systems and automated negotiation.
In this chapter, we have presented a general overview of the multi-agent Systems. The basic concept of this domain is the notion of agent, which represents an autonomous entity capable of perceiving, representing and acting on its environment. MASs help to meet the standards of software engineering in the development of systems, that is to say: modularity, reliability and reuse. This explains the increasing use of MASs in computer systems.
In the next chapters, we present the other two domains in which the work of this thesis is integrated.

Chapter 2

NEGOTIATION IN MULTI-AGENT SYSTEMS

2.1 Introduction

In systems composed of multiple autonomous agents, negotiation is a key form of interaction that enables groups of agents to arrive at a mutual agreement regarding some belief, goal or plan, for example. Particularly because the agents are autonomous and cannot be assumed to be benevolent, agents must influence others to convince them to act in certain ways, and negotiation is thus critical for managing such inter-agent dependencies. The process of negotiation may be of many different forms, such as auctions, protocols in the style of the contract net, and argumentation, but it is unclear just how sophisticated the agents or the protocols for interaction must be for successful negotiation in different contexts. All these issues were raised in the panel session on negotiation.
In this chapter, we present the negotiation activity in general as well as its components, types, approaches and protocols in multi-agent systems.

2.2 Definitions

- Bussmann and Müller in [33] proposed the following definition of negotiation: "negotiation is the communication process of a group of agents in order to reach a mutually accepted agreement on some matter."

- Durfee and Lesser in [34] gave the following definition: "the negotiation is a complex process of improving agreement on common viewpoints or plans through the structured exchange of

relevant information."

- For Wooldridge [35] :"Negotiation is seen as a method for coordination and conflict resolution."

- Negotiation for Bedou in [36] "is an iterative communication process during which there is attack and defense of certain points of view, solutions, beliefs, knowledge, proposals of alternatives, so as to reach one of the agreements, and this by revision of beliefs more obtaining new information, through argumentation, explanation, and where the agent clearly expresses the state of his beliefs and knowledge for a given subject."

- Rahwan et al. in [37] adapted the following definition :"Negotiation is a form of interaction in which a group of agents, with conflicting interests and a desire to cooperate, try to reach a mutually acceptable agreement on the division of scarce resources."

- Smith [29] defines negotiation: "By Negotiation, we mean a discussion in which the interested parties exchange information and come to an agreement.For our purposes, negotiation has four important components: **1)** it is a local process that does not involve centralized control, **2)** there is a two-way exchange of information, **3)** each party to the negotiation evaluates the information from its own perspective, and **4)** final agreement is achieved by mutual selection."

The different definitions share a common point which is the absence of an agreement on a given problem (all researchers agree on the purpose of the negotiation, which means to reach a satisfactory common agreement).So generally, we negotiate to solve disagreements, prevent a conflict, obtain a good, in short to satisfy a need, and this through an exchange process involving at least one other person.

2.3 Negotiation Components in MAS

Of all the work done in the negotiations, there are a number of common points on which we must now rely. Three elements predominate: a negotiation protocol, resources to negotiate and a reasoning model. Beer [38] identified three components that are the subject of negotiation research:

1. ***Negotiation protocols:*** the protocol defines the set of rules that govern the interaction. This covers the types of authorized participants, the trading statuses, the events causing

the change of trading status and the valid shares for a participant according to the current state of negotiation.

2. ***Negotiation objects:*** the definition of the issues over which an agreement must be reached. In many cases, for example in auctions, only one property of the object is negotiated, usually the price. Generally may be possible that numerous properties of the object are negotiated, for example quality, date of delivery, penalties and so on. Another issue concerns the type of operation that participants can perform on the agreement under negotiation. Very often the structure of the agreement is fixed and participants can only accept or reject it. In other situations the negotiation may be much more effective if the participants are able to change the characteristics of the agreement. For example participant can answer to a proposal with critics or counter-proposals. The agents do not negotiate a satisfactory agreement according to Negotiation protocol defined in advance. Such a protocol defines the possible actions on the objects of negotiation (offers and counter offers, possibility of failure, auction time, etc.) [39]

3. ***Reasoning and decision-making models of agents:*** concerns the design and development of the decision making mechanism of participants on the basis of the restrictions imposed by the negotiation protocol, on the nature of the negotiation object and in order to achieve their goals or to maximize some sort of utility. Regarding to the utility notion, usually agents negotiate on behalf of their owner.
   Each agent has its own decision-making model which can be more or less complex depending on the protocol, the nature of the negotiation objects and the possible operations during the process. The reasoning of the agent is strongly constrained by the protocol and the purpose of the negotiation and should help him to achieve his objectives [40]

Muller in [40] which was cited in [41], distinguishes between three elements in the negotiation (See Figure 2.1) :

1. *Languages*, which include communication primitives for negotiation, their semantics, and their uses in protocols.Primitives concern sending and receiving messages.The structure the negotiation object uses a description language of the object. The protocol specifies the possible sequences of actions and the conditions under which a request can be made,

2. *Negotiation processes* that may be either pre-established procedures or behaviour. The process concerns the proposal of solutions, the analysis and revision of preferable solutions,

3. *The individual decision* that can be made according to criteria of utility, strategy, preference systems, a comparison function(matching).

Figure 2.1: The different elements of negotiation and their structures [2].

2.4 Negotiation Types

In literature, we distinguish two types of negotiation: distributed negotiation , and Integral negotiation [42].

2.4.1 Distributed negotiation

Distributed negotiation is a process in which the negotiating agents are in a sum - to - zero logic, that is, a "win - lose" logic. As a result, each agent's goal is to maximize profits, and negotiation becomes a competition where the strongest wins over the weakest [43]. Each negotiation agent pursues individual goals and seeks to be the winner of the exchange. Everyone stands on their positions and negotiation is seen as a confrontation (showdown).So the deal is likely to be win-lose or lose-lose.

2.4.2 Integrative negotiation

Integral negotiation is a win-win process. Any win for one will be a win for the other. The context created by this partnership relationship is to define clearly the problem and explore all possible solutions to engage in an equitable solution between the two parties [44]. It generally

leads to an agreement in which both parties consider themselves winners. The agents take into account the interests of each in their decision: they pursue a common goal and the negotiation enables them to coordinate. This is a distributed search in a solution space.

2.5 Negotiation Approaches

Many research studies have been done to model the negotiation between software agents and between agents and humans. In the MAS community, there are often three approaches [45], [37]:

- Negotiation based on game theory [46] [47] [48],
- negotiation based on heuristics [49] [50] [51],
- and negotiation based on argumentation [52] [53] [54].

2.5.1 The Game Theory Approach

Game theory is a tool for studying interaction strategies between autonomous agents in automated negotiation. This approach allows agents to behave rationally when making their decisions and strategy choices.

2.5.2 Heuristic Approach

This approach does not always provide an optimal solution because it uses approximate strategies and it does not examine all the possible space of the results but it makes it possible to provide a solution closer to the optimal one. It is based on the testing and evaluation of the different results. This approach can not predict the exact behaviour of the system and those of its agents and it assumes that the agent has a complete knowledge of his desires and preferences

2.5.3 Argumentation-Based Approach

To implement such a system, it is necessary to add some arguments to the agents to support proposals and counter proposals. There are several types of potential arguments such as threats, rewards, reductions ...etc.Argumentation-Based negotiation increases the likelihood of an agreement, saves time and reduces the risk of failure. And for that it is necessary to build negotiating agents capable of arguing by specifying the mechanisms for the exchange of proposals and arguments, to implement the techniques in order to generate proposals and provide the

corresponding arguments, to provide the techniques for accepting proposals and the arguments associated or still technical to meet the proposals.

2.6 Negotiation Protocols

Negotiation protocol defines the rules of communication that negotiating agents must adopt and respect in their information exchanges [55]. It also defines:

- The types of possible participants (sellers, buyers, mediator ...).
- The states of negotiation (Offers, acceptance of a proposal, end of the negotiation).
- The events that end the transitions of states (No new offers, time gone).
- The valid actions of the participants (Acceptance, rejection, counter proposal ...)

For Beer et al. in [38], a negotiation protocol is "the set of rules that govern an interaction".

Many negotiation protocols have been proposed in multi-agent systems. We will introduce the best known.

2.6.1 Contract Net Protocol

The Contract Net Protocol (CNP) has been proposed by R. G. Smith in 1980 [29]. It is a protocol based on the exchange of contracts, which connects an agent: the manager, with several other agents: the contractors.This is a negotiation from 1 to n agents that interact with each other. A contract can be developed between two participants: a contractor and the manager. The contractor is responsible for the execution of the task and the transmission of its results to the manager. The manager is responsible for managing the task and processing the results. That's why this model can be seen as a protocol for assigning tasks by contract network.The Contract Net is part of the FIPA (Foundation for Intelligent Physical Agents) specification (See Figure 2.2)

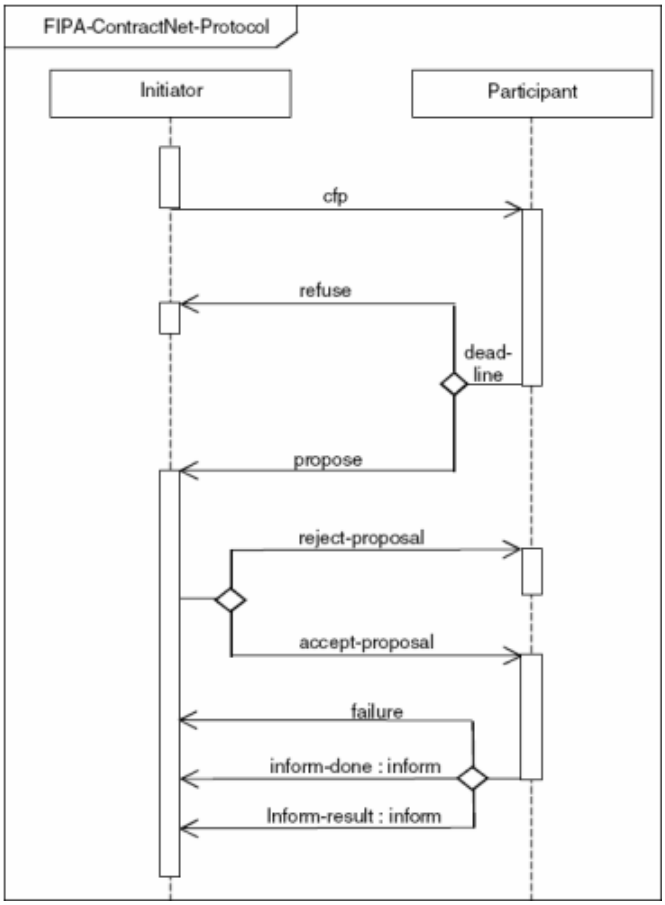

Figure 2.2: Interactions in the Contract Net protocol.

The protocol is as follows (See Figure 2.3):

1. Task announcement by the manager by sending a call for proposals (cfp)

2. Offer submission by one or more contractors.

3. Task Assignment to a contractor by the manager.

4. Transmission of the termination message to the manager by the contractor.

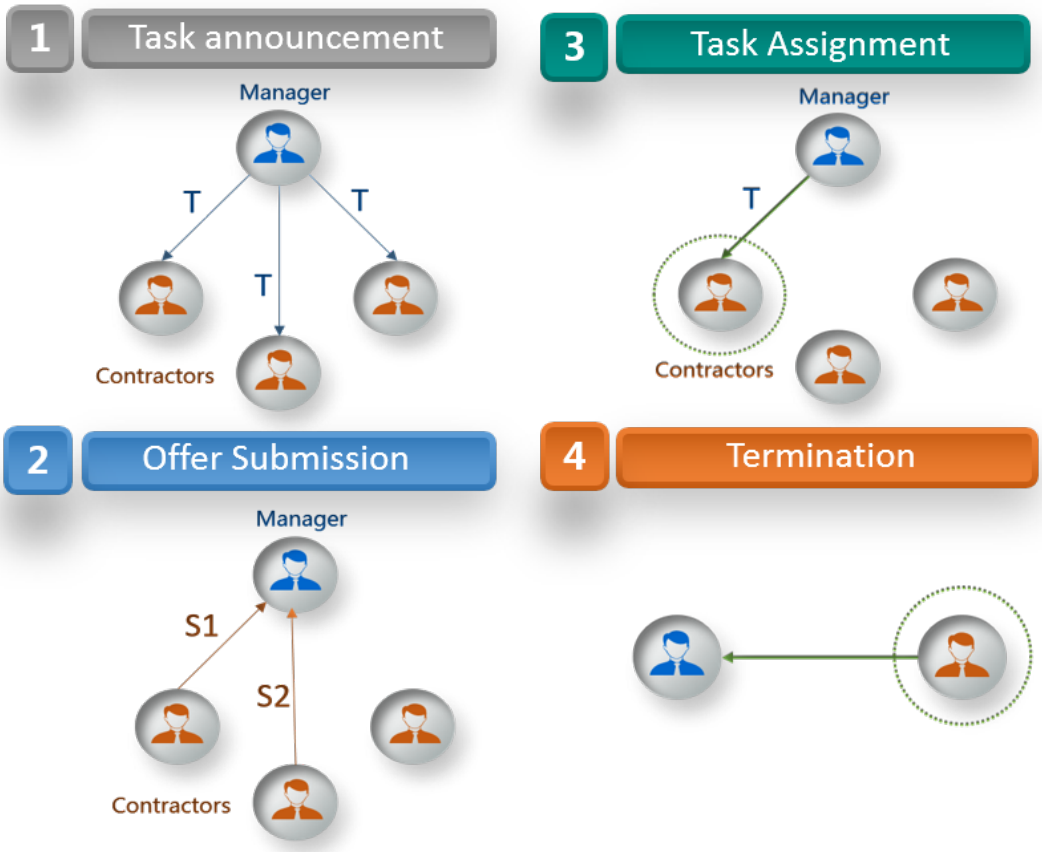

Figure 2.3: The Contract Net protocol.

Many negotiation protocols are based on the Contract Net .Originally,the purpose of this protocol was to propose a model for the distribution of tasks, based on calls for offers in public markets [56].

2.6.2 The Sian's protocol

This negotiation protocol has been designed within Multi-agent Learning Environment (MALE) ,by Sian in 1991 [57], it is a simple protocol where an agent proposes a hypothesis to other agents. Those agents provide their opinion on whether the hypothesis is correct or incorrect, and whether they want to modify it. Each agent votes on the correctness of the hypothesis. The agent that proposed the hypothesis gathers the votes and, using some heuristics, decides whether the hypothesis is accepted or not. If the hypothesis is accepted, it informs other agents of theacceptance. Figure 2.4 illustrates the Sian protocol.

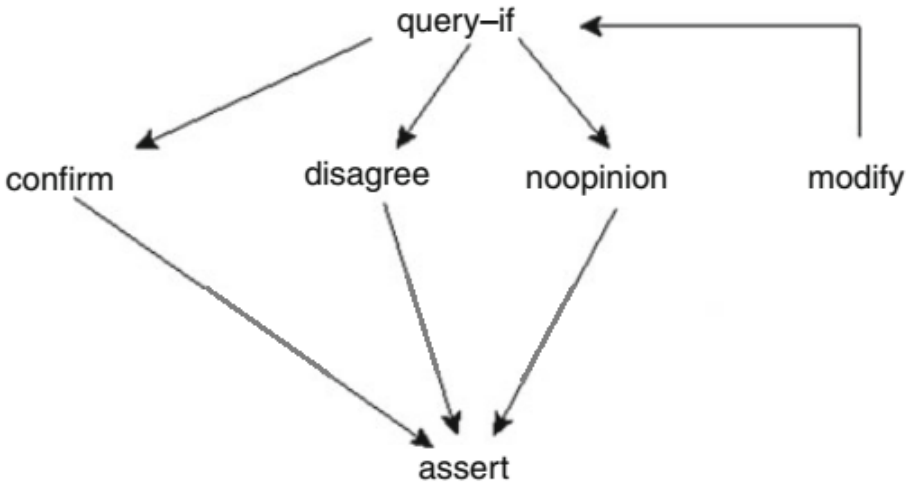

Figure 2.4: The Sian's protocol.

The first message is *a query-if message* containing the hypothesis to validate. The other agents can answer either by *a confirm message* denoting that the hypothesis is correct, by *a disagree message* denoting that the hypothesis is wrong, by *a noopinion message* denoting that they have no idea about this hypothesis, or by *a modify message* requesting to modify the hypothesis. Based on answers from other agents, the initiator decides whether the hypothesis is true. If it decides that the hypothesis is true, it informs the other agents via *an assert message* that the hypothesis is considered true.

2.6.3 Th Speech Act Negotiation Protocol

The Speech Act Negotiation Protocol (SANP) was defined by Chang and Woo in [58] [59].It proposes a negotiation between two agents: the attacker and the defender. The agent initiating the conversation is called the attacker: he will try to make his proposal accepted by his partner, called the defender.

The protocol is broken down into several phases:

1. *Starting situation* the agents establish a common model of the subject and consider whether a discussion is necessary.

2. *Issuing a proposal and receiving the notice* the attacker issues his proposal and the defender answers his agreement or refusal,

3. *Attack* if the defender rejects the proposal, the attacker will issue his arguments

4. *Tactical phase* the defender will defend his position and the attacker will try to counter it with arguments. This process continues until they identify their differences and try to reduce them,

5. *Problem Solving* each party proposes a compromise and tries to counter that of the other,

6. *Final result* the outcome of the negotiation may be the compromise of a party, a mutual compromise or third party arbitration request.

2.6.4 Auction Protocol

Auction mechanisms are particular interaction protocols that fix the rules of the interaction among one or more sellers and one or more buyers and fix the rules for determining resource allocation and prices on the basis of bids from the market participants. In auctions the seller, also called the auctioneer, want to sell an item at the highest possible value while buyers, also called bidders, want to buy the item at the lowest possible price. Therefore participants to an auction are self-interested agents that want to maximize their pay-off, but usually an auction is not a zero-sum game, that is games in which one player's losses are the opponent's gains. An auction's outcome is usually the definition of a contract between a seller and a buyer for the exchange of a certain product.
Many auction protocols are implemented in multi-agent systems. Among them, the English auction and the Dutch auction are likely the most widely implemented and used. The English auction is an ascending open cry auction where the price of the item for sale increases as long as at least one auction participant is willing to pay the price. It stops when no participant is willing to raise the price beyond the current bid.
The Dutch auction protocol is a descending open cry auction. Unlike the English auction, the price decreases as long as a participant does not inform it accepts to pay the price.

2.7 Conclusion

Negotiation plays a fundamental role in the development of agents and their collaborations. As a result, we have tried throughout this chapter to clarify the notion of negotiation in real life as in multi-agent systems by giving some definitions, furthermore we tried to give most of the principles that make it possible to understand the negotiation between the agents without forgetting the concepts on which it is based.

Chapter 3

CLOUD COMPUTING AND SaaS CONTRACTS

3.1 Introduction

Undeniably, internet technology has been growing exponentially since its inception. Currently, a new "trend" has appeared in the world ofInformation technology(IT), which is Cloud Computing [60] [61] [62].It is a steadily maturing large-scale model for providing on-demand IT resources (compute, storage, networks, platforms and applications) as a service over the Internet. With the evolution of virtualization, high-speed Internet access and especially the support of leader IT companies. Cloud Computing has become one of the fastest growing fields in the IT industry [63], [61], [64], [65]. This increasing attractiveness of Cloud results from its efficiency and flexibility, enabling customers to rapidly provision and access resources from anywhere and at any-time on a pay-per-use basis. Cloud Computing allows its users to avoid the installation and management efforts by externalizing their hardware and software resources to a large-scale environment promoting high availability and reduced costs.

In this chapter we present the basics of Cloud Computing, this chapter is divided into two main section. At first we present different definitions of cloud computing, and different concepts related to it. Then we explore the Software as a Service(SaaS)contract and its different clauses.

3.2 Cloud computing

3.2.1 Cloud Computing Definitions

Cloud computing is a large and dynamic field, so that today no one claims to have a clear and absolute definition. Indeed,several definitions of cloud computing have been published but most of them seem to focus only on certain aspects of this technology. Among these definitions we cite the followings :

- Foster et al. [66] defined Cloud computing as "a specialized distributed computing infrastructure with four main characteristics:

  1. It is enormously scalable,
  2. It is an intellectual entity that delivers different levels of services to clients,
  3. It is driven by economics by scale, and
  4. Its services can be configured dynamically."

- According to the official National Institute of Standards and Technology (NIST) definition [67]," Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction and users are billed according to their usage of services."

- According to Forrester Research [68], cloud Computing is : "A standardized IT capability (services, software or infrastructure) delivered via Internet technologies in a pay-per-use, self-service way."

- The European Network and Information Security Agency (ENISA) [69] has defined Cloud Computing as "on-demand service model for IT provision, often based on virtualization and distributed computing technologies."

- Also the J. Kaplan (2008) in [70] has defined the Cloud Computing as "a broad array of the web-based services aimed at the allowing users to obtain a wide range of functional capabilities on a 'pay-as-you-go' basis that previously required tremendous hardware/software investment and professional skills to acquire".

- While B. Martin (2008) in [70] propose the following definition of Cloud Computing " Cloud Computing really comes into focus only when you think about what IT always needs: a way to increase capacity or add capabilities on the fly without investing in new infrastructure, training new personnel, or licensing new software"

With cloud computing it becomes easier to access data with several devices. Especially for mobile devices this can be really useful since the only thing that is needed, is an Internet connection. In Figure 3.1 a diagram is shown of a cloud based solution.

Figure 3.1: Cloud computing diagram.

3.2.2 Cloud Service Models

Cloud Computing services are provided in many forms to customers according to their specific needs and these models are classified into three types of services: 1) Infrastructure as a Service (IaaS), 2) Platform as a Service (PaaS), and 3) Software as a Service (SaaS) [71] [72]. Other services have also been mentioned in the existing literature, such as Hardware as a Service (HaaS) [73]. However because of non-existence of exact and clear definition of these services, mainly three of the services are considered. These 3 services are the most important pillars of the Cloud through which the cloud solutions are provided to the customers.

3.2.2.1 The IaaS

Infrastructure as a service (IaaS) is the most basic service model since it offers a total infrastructure. This includes processing power and storage capacity. Sometimes firewalls and load balancers are offered but this functionality often belongs to the PaaS model. Providers with this service model have large amounts of storage and processing power which they supply on demand. The customers of these providers can install some operating system images as well as applications. By doing this, the customer has a lot of freedom to create his own environment. The downside of this is that basically only network architects can work with it since a complete network has to be set up. In this situation the provider often owns all the equipment and the operating systems and applications will run all in the cloud. This has as an advantage that when the hardware at the client's side breaks or gets stolen, it is easy to replace it. The client will just have to buy some new hardware to access the cloud and there he can find all the data. Mark Russinovich (Microsoft) explains the IaaS solution as servers on demand. An IaaS solution is the only service model that delivers completely platform independent resources to its users [74].

3.2.2.2 The PaaS

Platform as a service (Paas) is typically designed for software developers and provides them with platforms to design, develop and deploy applications. PaaS platforms are high-level integrated environments (OS, program-

ming languages, libraries, databases, web servers . . .) supporting the full software life-cycle. PaaS users have full control of the applications and the environment configuration settings, but no control of the underlying infrastructure that is maintained by the cloud provider. This aims at simplifying the software development process and allowing developers to focus on their applications' core features without worrying about complex low-level management operations. Examples of popular PaaS platforms include Google App Engine [75], Microsoft Azure Cloud Services [76] and Pivotal Cloud Foundry [77] .

3.2.2.3 The SaaS

Software as a service (SaaS), is the highest level of the Cloud stack that delivers complete applications to consumers through the internet. The SaaS provider is responsible for hosting, managing and controlling the application and its running environment (hardware infrastructure, software stack, access and security aspects,. . .). Details about the underlying infrastructure are transparent to SaaS users, who have simple access to the application's functionalities without the ability to control or customize its features. The SaaS delivery model has notably been popularized with SalesForce [78] and its Customer Relationship Management (CRM) application. Today the SaaS is more widespread with many new offerings such as social media platforms, e-mails, business accounting, collaboration, management applications and online-gaming.

Apart from being the services stack these layers indicates the roles and responsibilities of the users and providers. As the height of the layer increases, the managing responsibilities are shifted from users to the providers, and this is shown in Figure 3.2.
This figure demonstrates how the cloud provider has more operational responsibilities for PaaS and SaaS applications compared to IaaS.

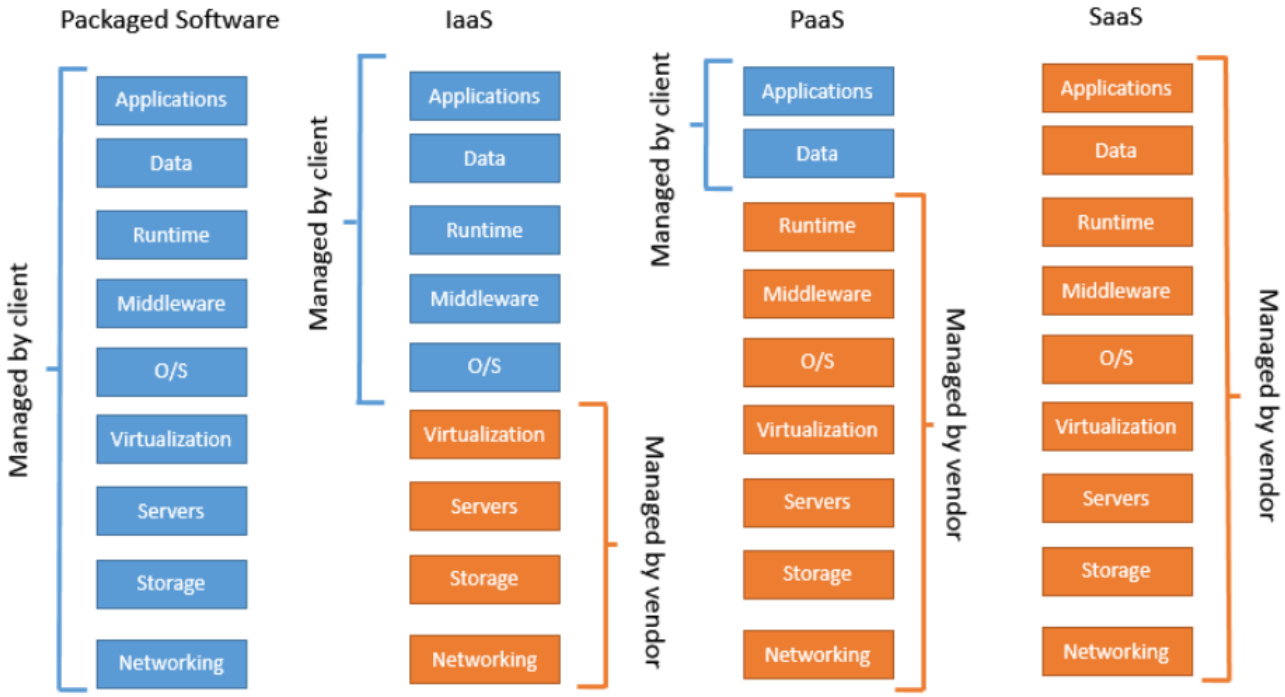

Figure 3.2: Cloud ownership in different service models.

3.2.3 Cloud Deployment Models

There are three main types of deployment models for cloud computing (See Figure 3.3): **1-***public*, **2-** *private*, and **3-** *hybrid*.

These cloud models share common characteristics but the main difference is because of the different groups of users for whom the Cloud is built.

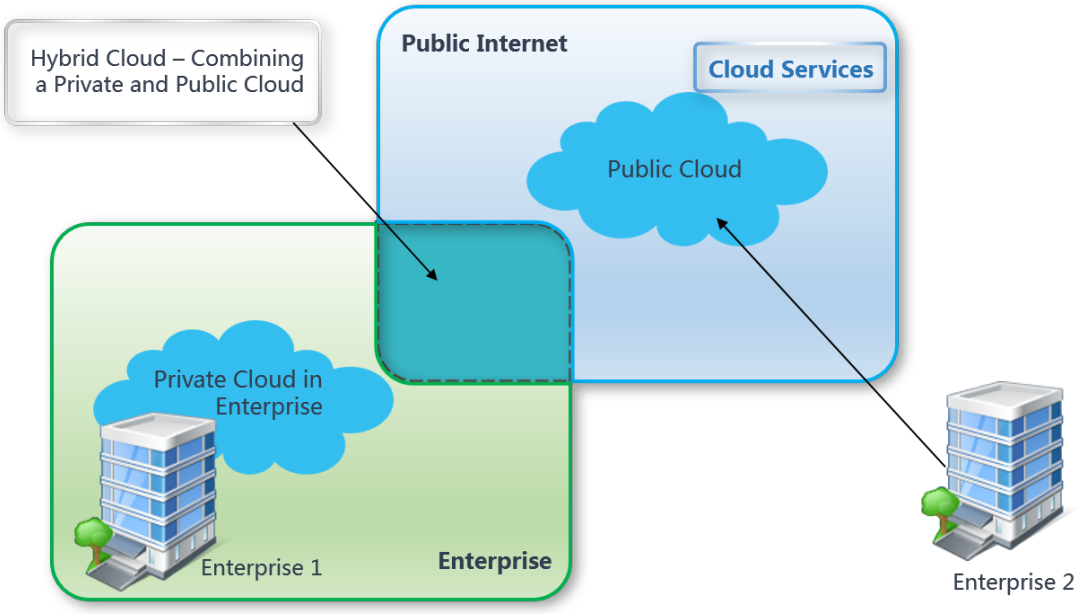

Figure 3.3: Cloud deployment models.

3.2.3.1 The Public Cloud

Public cloud is the most popular cloud computing deployment model at this moment. It provides a cloud service opened to the public. It can support a large number of requests with low cost. Public cloud providers [71] provide their clients with a variety of resources, including physical infrastructure, applications, software runtime environment and security, management, deployment and maintenance of these resources.When clients are using IT resources, they only need to pay resources and services they use. Though they share resources with other public cloud clients, they do not need to know about other clients or how the cloud platform is structured. Cloud provider is the only one who can control the physical facilities and make sure of the non-functional requirements like safety and reliability. Many big IT companies, like Amazon's AWS [79], Microsoft Windows Azure [76], Google's Google Apps [80] and Google App Engine [75], provide their own public cloud services.

3.2.3.2 The Private Cloud

The private Cloud, also known as enterprise Cloud, is another deployment model for cloud services. In this model the cloud resources are not shared by unknown third parties but are exclusive to a single organization only [81]. The cloud resources in this model may be located within the client organization premises or offsite. In this type of cloud the general public does not have access to the private cloud neither does the organization use the public cloud [82]. So private clouds can offer the provider and client greater control, security, and resilience than public cloud.

3.2.3.3 The Hybrid Cloud

The hybrid Cloud, as suggested by the name, is a combination of both public and private Cloud , this combination allows to keep the client's privacy and low the costs of building the cloud. By using hybrid cloud, enterprise can put the important or high security level parts of applications in their private infrastructure while others in the public clouds to decrease costs. Though hybrid cloud is not as popular as public cloud and private cloud, there are still some hybrid cloud products, like Amazon VPC [75].

3.2.4 Cloud Actors

There are five major actors identified by NIST [83] in the cloud reference architecture. Each actor performs a specific set of activities, and also the roles of these actors are interchangeable depending on the configuration of the particular cloud model.
Table 3.1 briefly lists the actors defined in the NIST cloud computing reference architecture.

Table 3.1: Actors in Cloud Computing

Actor	**Definition**
Cloud Costumer	A person or organization that maintains a business relationship with, and uses service from, Cloud Providers.
Cloud Provider	A person, organization, or entity responsible for making a service available to interested parties.
Cloud Auditor	A party that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation.
Cloud Broker	An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between Cloud Providers and Cloud Consumers.
Cloud Carrier	An intermediary that provides connectivity and transport of cloud services from Cloud Providers to Cloud Consumers.

3.2.5 Benefits of Cloud Computing

Cloud Computing with its different deployment and delivery models offers a number of benefits to businesses [84], [85], [86], [87], [88].
These benefits are:

a) Economies of scale resulting in low-costs of IT infrastructure, low maintenance costs and low IT administration costs.

b) It offers easier data monitoring.

c) The quality of E-Mail service, collaboration tools & storage services are better than any of the available paid on-premise services.

d) Quick & effective Communication with anytime anywhere access.

e) Improved performance as a result of having access to dynamic and scalable computing.

f) Security will be taken care of by the provider.

g) Universal access to computing resources.

3.3 SaaS contract

3.3.1 Cloud Software as a service

Before Cloud computing came into view, SaaS had been successfully implemented in the servers of SaaS vendors and it was delivered via Web [89]. But, when there was an increasing demand for SaaS each year [90], SaaS vendors need to find a solution to cope with these growing requests. An obvious solution for this problem is to host the SaaS in a Cloud computing infrastructure as it provides scalability to the SaaS that runs in a Cloud.

Based on the published definitions of Software as a Service , the best basic definition of SaaS is provided by Chong and Carraro [91], who's referred to as a software deployed as a hosted service and accessed over the Internet.

A SaaS model for serving customers in Cloud is shown in Figure 3.4. A customer sends requests for utilizing enterprise software services offered by a SaaS provider, who uses an application layer to satisfy the customer's request.

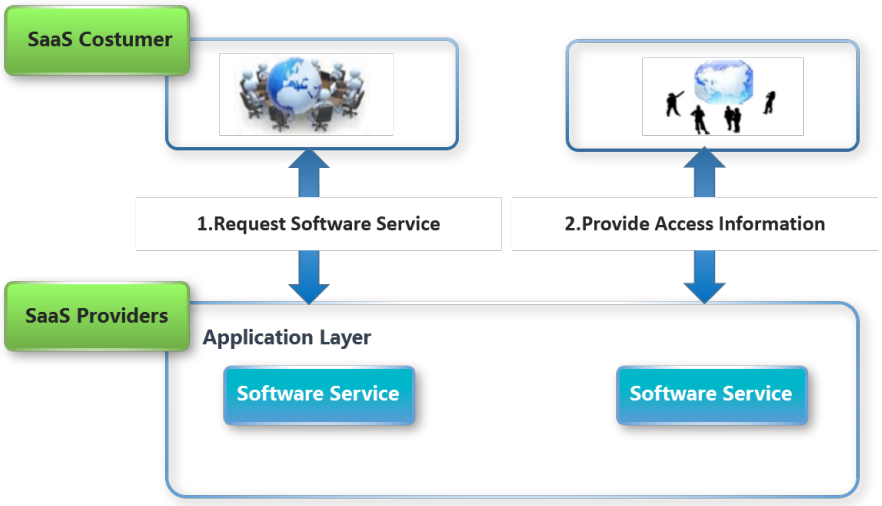

Figure 3.4: A system model of a SaaS layer structure.

3.3.2 The advantages and disadvantages of SaaS

3.3.2.1 Advantages

When we subscribe to a software service rather than purchase a software license, we get the following advantages:

- **Saving money**

  1. **Cost Reduction**
     We avoid overhead costs associated with the implementation of traditional software. Typical software implementation involves the purchase and maintenance of servers, the safe storage, installation and maintenance of software. This requires time and effort, IT, experienced staff and diverts employee efforts on a certain number of levels outside of your organization's core mission.
  2. **The Use**
     we pay a monthly or annual subscription. Compared to a traditional software license, this payment structure subscription is to our advantage, we can cancel or modify our subscription at any time without losing a significant initial investment.

- **Time optimization**
  By eliminating most of the typical runtime tasks associated with licensed software and having the software already installed running on the SaaS provider's data center, the deployment time tends to be much shorter with SaaS application.

- **Focus Budgets on Technology and Competitive Advantage rather than the infrastructure**
  When we subscribe to a hosted web-application, we free our organization of high-cost technical support and time management, including:

  – Purchase and support the server infrastructure needed to install and maintain the software internally.

  – Provide the necessary equipment redundancy and hosting to ensure security, reliability and scalability.

- **Immediate access to the latest innovations**
  With traditional licensed software, we usually have to wait for the next release to benefit from the latest innovations or move our organization to a new browser or operating system. Given the cost and complexity of switching to a new version, it may not even be practical to upgrade each time a new version is available.

  On the other hand, with a SaaS subscription, we benefit from innovations on an ongoing basis. As soon as a new feature or enhancement appears in the app, we can start using it.

- **Join a community of interest**
  The purchase of a traditional software license is much of an individual affair. However, when we subscribe to a SaaS application, we become a member of a community whose demands are the focus of concern.
  SaaS changes the relationship between software vendors and customers. In a service environment, a convergence of interest between the customer and the seller that is more intimate than the one expressed

in the world of conventional applications.

Intimate results because :

- SaaS providers continually monitor how their customers use the app.
- Customers can easily compare themselves to their peers.

3.3.2.2 Disadvantages

Despite these advantages, there are disadvantages that cause business problems in the SaaS approach:

- Requiring a constant internet connection, can be slow in case of slow internet connections, limited features offering, security might not meet the organization standards, danger of loss of business in case of data loss or cloud vendor filing for bankruptcy [86], [92].
- The problem of respecting the confidentiality of company data is very often put forward as a brake on the adoption of SaaS services. This is probably the most delicate aspect of SaaS resistance in the sense that in the end it is reduced to the trust that the company's decision-makers have in the provider. The disclosure of information to unauthorized third parties may have an accidental origin, as in the case of Google [93] in early March 2009 : Documents stored on Google Apps were visible to unauthorized users.

3.3.3 SaaS Contract Definition

A SaaS contract is an agreement between a service provider and a customer of a SaaS solution .It is used to define the terms of service, licensing requirements, and other legal terms to clients of Software as a Service companies.It provides an on-line reference to both the providers of the software and its users.

3.3.4 Common Clauses of SaaS Contracts

SaaS Contracts make it easy to present all legal documentation at once to service users. the content of the contract differs from a service provider to another.Depending on their preferences SaaS service providers can include any other documentations or conditions .Common parts of a SaaS contracts are the following :

1. **Scope of the agreement**

   Includes the parties of the contract as well as a global definition of the contract.

2. **Contract Period**

   the contract Period or The term of contract is the length(duration)of the agreement between the service customer and provider.

3. **Pricing**

   This clause is concerned with the The amount which is to be paid by the service customer for continued use of the software.

4. **Confidentiality and security**

   Each party shall treat as confidential all Confidential Information of the other party, shall not use such Confidential Information except to exercise its rights and perform its obligations under this SaaS Agreement, shall not disclose such Confidential Information to any third party, and shall effect and maintain

adequate security measures to safeguard the other party's Confidential Information from unauthorised access or use.

5. **Monitoring and control, audit and inspection**

   Monitoring and **Control** present the structure, policies, and procedures used to ensure that the objectives of a contract are accomplished and providers meet their responsibilities, while **audit** is the process of checking that compliance obligations have been met, including that the required **inspections** have been done.

6. **Dispute resolution**

   Each party must, to the extent possible, continue to perform its obligations under the Agreement even if there is a dispute.

7. **Default termination and early exit**

   Includes termination rights , consequences of termination or expiry on both parties of the agreement, as well as causes for service provider to restrict or suspend the customer's access to the SaaS Service.
   If a Party fails to perform or observe any material term or condition of the contract and the failure continues, the other Party may terminate the contract.

8. **Service Level Agreement (SLA)**

   SlA is the service contract component that provides objective and measurable assessments of key elements of the service in the provider's standard form contract, including:

   - Service availability :
     means the number of minutes in a year that the key components of the SaaS Services are operational as a percentage of the total number of minutes in such year
   - Service response time :
     the performance of the service is related to the response time .

9. **Warranties and Indemnities**

   The requirement of one party to pay for any necessary defence costs and damage awards in the event of a third-party claim.

10. **Intellectual Property**

    includes copyright and all rights existing anywhere in the world conferred under statute, common law or equity relating to inventions (including patents), registered and unregistered trade marks and designs, circuit layouts, data and databases.

11. **Penalties**

    If the service which was delivered does not achieve service objectives or is below the performance measurement, some penalties will follow.

12. **Limitation of liability**

    Describes in specific terms the level of liability of the SaaS service provider and client.

3.4 Conclusion

In this chapter,and in the first part we discussed the cloud computing paradigm that allows companies to have infrastructure and software directly online on the Internet. Then we showed the three cloud services IaaS, PaaS and SaaS. These three services can be deployed in three different topologies: Public Cloud, Private Cloud and Hybrid Cloud. After that we presented the different actors in the cloud environment and why cloud computing is beneficial for business.In the second part we studied SaaS , a software deployment model in which applications are accessible through the Internet as well as its pros and cons.Then we gave a global description of SaaS contracts,their definition and some common clauses that we can find in all SaaS contracts which can ensure that expected requirement and efficiencies are actually achieved .

Chapter 4

PROPOSING A NEGOTIATION PROTOCOL FOR SaaS CONTRACTS

4.1 Introduction and Motivation

In a multi-agent system, agents interact in order to accomplish tasks or to achieve goals. The interaction takes place, usually in a common environment where the agents have different areas of influence, including various parts of the environment on which they can act. These areas can be disjointed but in most cases, they overlap.By interacting in such a shared environment, agents must coordinate their actions and have mechanisms for resolving conflicts.

The preferred mechanism for conflict resolution and coordination , inspired by the human model, is negotiation .
In the case of intelligent agents and in multi-agent systems, negotiation is a basic component of the interaction, especially because agents are autonomous [94].

In this chapter, we give a detailed description of our contribution, which consists in proposing a protocol that allows the negotiation of the SaaS contracts between two parties; Cloud Service Customer (CSC) and Constraint Satisfaction Problem (CSP)s .For that, we start by introducing the problem that we are trying to solve as well as the objectives aimed at by this study .Then we present some related work and discuss a little the contribution of the new solution, after that we present the general functioning of our negotiation process.

In the second part of this chapter, we discuss the relationship of our solution with Contract Net Protocol model. In a third place, we introduce the global process of the proposed negotiation.Then in the fourth part, we begin by presenting the proposed protocol in a global way. After that we try to give a detailed description of the different internal evaluation functions of the proposals, and counter proposals, as well as the mechanism used to generate a proposal or a counter proposal. Then in the fifth section of this chapter, we present the architecture of a negotiating agent that we adopted in our work.And finally, we will end this chapter with a

conclusion.

4.2 Problem Statement and Objective

Negotiation plays a fundamental role in Cooperation Activities by enabling people to resolve conflicts that could jeopardize cooperative behaviour.

Typically in Cloud computing, and specifically in Software as a Service(SaaS), which is One of the most important models of delivery in the Cloud ,a software is offered as a service by cloud service providers who also define their service contract, costumers, on the other hand search for services,and in order to Select the most appropriate one , negotiation that enables them to impose their requirements on Cloud providers is needed.

Our objective is to propose a negotiation protocol for SaaS Contacts that allows the two parties;the(SaaS)cloud service customer and the cloud service providers to enter a negotiation concerning multiple issues of the contract related to the service in question ,in an automated way , and for that we propose to include intelligent agents in the functional architecture of the protocol .The proposed protocol is based on multi-agent systems (MAS). The choice of multi-agent systems can be justified by the fact that these systems have several good properties that make them more appropriate to our working context, among these properties we can mention [95]:

The distribution geographically and logically of entities, data or heterogeneous information;

The complexity of the global problem to be solved, which can only be manipulated by heuristic strategies using local data or knowledge;

Dynamic environment requiring, for problem solving, reactive and adaptive entities;

Openness It is not possible to give a complete specification of the problem to be solved nor to define a global utility function.

As a result, multi-agent systems seem to be particularly well suited to modelling the issues (distribution, openness, flexibility and collaboration of services) underpinned by the negotiation of SaaS contracts .That's why the object of our study will focus on the study of MAS solutions for the modelling of SaaS contracts negotiation.

4.3 Related Work

In this part, we will review the negotiation frameworks that have been proposed in areas related to our work , to discuss their limitations and our contribution.

In [96], the authors proposed a broker-based negotiation to deal with the various demands of cloud customers. Typically, the broker will select suitable providers and negotiate with multiple providers on behalf of customers.

In [97], the author proposed a complex Cloud negotiation mechanism that uses three types of agents: consumer agents , provider agents and broker agents ,the proposed mechanism uses negotiation to establish an SLA (cf.chapter 3) between cloud participants. The negotiation happens between consumer agents and broker, and also between provider agents and broker agents.

In [98],the authors designed a cloud SLA negotiation mechanism to support a flexible establishment of SLAs for both providers and customers. The novelty of the SLA negotiation mechanism is that it can support multi-issue negotiation.

In [99], the authors suggested using automated and intelligent negotiation solutions for reaching an SLA for an open competitive computational grid.

In [100], the authors proposed an automated negotiation mechanism for grid resource negotiation. These negotiation mechanisms are designed for price negotiation, but fail to consider other SLA issues such as service time slot and response time.

In [101] [102], the authors considered a cloud services market in which a client creates an SLA proposal for each provider, then providers give their offers. Afterwards, the client selects the provider whose SLA offer suits best its requirements.

In [103], the authors proposed a broker-based framework for Software as a Service (SaaS) . The SaaS Broker selects a suitable SaaS provider on behalf of the service consumer. Then, it negotiates the SLA terms with that provider based on the quality requirements of the service consumer.

In [104], the authors considered a concurrent one-to many negotiation mechanism for Grid resource co-allocation, with a consumer agent negotiating for multiple resources in multiple Grid resource markets.

4.3.1 Contribution compared to Related work

There is not much research for cloud computing in this area of negotiating contracts (SaaS contracts). and for that reason we chose to compare our proposed model with related work (subsection 4.3) , by defining some fields of negotiation which related works highly missed.
The negotiation for cloud computing has to show adaptability in these manners : multi-issue negotiation, automated negotiation,and negotiating with the multiple providers.

- Multi-issue negotiation : The cloud service customer and the cloud service provider needs to negotiate over the multiple issues with the capability of giving preferences for each issue.
- Automated negotiation : The adaptability which allows the cloud service providers and customers to perform the negotiation process by using given intelligent agents .
- negotiating with the multiple providers : The cloud service customer should be able to negotiate with the multiple cloud service providers and then compare the outcomes with each one in order to choose the best offer.

In further aspects, we can deduce other limitations of related works that were listed previously:

In [96],authors suggest that cloud customers always choose using brokers to negotiate for them ,The issue with using brokers is that brokers will work on ending the negotiation with agreement, so they can obtain a commission from the deal.

In [97], the proposed mechanism requires the presence of the broker agents in order for the negotiation to happen, these types of agents work as inter-mediators between the other two types of agents,and as we mentioned above using a broker has its own issues.

In [98], the work only supports 2 issues which are time slot (Duration) and price negotiations. Also, in their work providers and customers have to end the negotiation with an agreement.providers and customers have to end the negotiation with an agreement.

In [99],the authors proposed approach is based on the negotiation in grid which is totally different from the one in cloud computing, furthermore their approach lacks a negotiation strategy that maximizes provider's profit since the negotiation is between users that want to use the same resource instead of providers competing for a customer (and that's the case in cloud computing).

In [100], the proposed mechanism is designed for price negotiation, but fail to consider other SLA issues or SaaS contract issue in overall.

In [101] [102], the proposed solution supports one issue which is pricing, and focused only on the service level agreement rather than the main contract.

In [103], the proposed framework doesn't allow a direct communication between the service consumer and provider, everything has to go through the broker, furthermore this framework is only concerned with service level agreement.

In [104], the authors proposed mechanism is in the field of grid computing which makes the negotiation process totally different from cloud computing. Moreover, their mechanism does not explicitly consider multi-issue negotiations.

Given the requirements above and the limitations, we propose an agent-based, multi-issue negotiation protocol for the SaaS Contract, this proposed work will allow the Cloud Service customer to negotiate directly with multiple Cloud Service Providers and to choose the best deal among the ones offered by those providers -if found-, giving them the ability to end the negotiation without an accord on both sides .The negotiation process will be performed through intelligent agents in a multi- agent system.and unlike the approaches above our negotiation will be concerned with the SaaS contract issues, including ones from the SLA which is a document attached to the contract.

4.4 Proposed model

4.4.1 General running of negotiation

The negotiation in our work is defined as a process which has three main components :

- **The negotiation parties**
- **The negotiation object**
- **The negotiation protocol**

The parties of the negotiation in our work are the Cloud Service Customer(CSC) and Cloud Service Providers(CSPs).

- Cloud Service Customer is the one that uses cloud computing services.
- Cloud Service Providers are those who provide cloud services to consumers.

The object of negotiation is Cloud service contract and to be specific the Software as a Service (SaaS) provided by the service provider.The attributes of this negotiation object are the set of parameters of this service indicated in the SaaS contract. Clearly, an attribute (a clause of the SaaS contract) may be negotiable or non-negotiable. This corresponds to the possibility of modifying or not the value of the attribute in a proposal. We can cite as an example of negotiable parameters(attributes):Price,Period...etc.As for non-negotiable parameters, there are for example: Monitoring and control, audit and inspection,liability...etc.These parameters and others are explained on the previous chapter.

The negotiation protocol can be described as follows ;agent A (the representative of a service customer) with the request for proposal "cfp" is the initiator of the negotiation. And according to the proposal that it receives from the other party; each agent B (the representative of each service provider), the agent A can answer with a rejection, an acceptance or a counter-proposal. thus by iteration on the proposals, both agents try to change values according to their needs. The negotiation process therefore leads to successive changes in the parameters, in conflict.

To modify the parameter in conflict, one can already imagine several scenarios. For example it would be possible to carry out a policy of looseness of the constraints (to be less demanding), in the realization of a particular effort (to be more efficient), or the refusal of admittance (stick on one's positions).

in the next section, we discuss the relationship between the Contract Net Protocol(CNP), and the proposed protocol.

4.4.2 Applicability of CNP to the proposed negotiation protocol

The negotiation protocols that we presented on the second chapter of this study allowed us to have an overview of the details of these used protocols.
the proposed negotiating protocol is based on the Contract Net Protocol(CNP). this protocol may be suitable for solving our problem because it is a distributed negotiation protocol that gives agents more control, unlike other protocols that are centralized and require a mediator whose role is to disseminate information between agents. This protocol is both based on the approaches of game theory and heuristics.
The strategies clearly depend on the target application and are all the more detailed as they are coupled to the application. a fine and application-specific strategy strongly determines the outcome of the negotiation and the speed of convergence to the point of agreement.
We can distinguish simple strategies that involve linear or quadratic types of functions (or any type of hard-wired function that are generally suitable only for negotiation on a single given criterion, and the more complex strategies of Jennings [105] [106] [107] that implement functions of evaluation(multi-criteria weighted additive

function) and of proposal (concession, give and take , add-drop)that are more evolved.
In recent models, agents can be both cooperative and competitive and exchanges are formalized. So for the decisions of the agents, (acceptance / refusal) of the contracts,the strategy that implements a multi-criteria weighted additive function and is sufficiently general and may be suitable for our negotiation protocol.

4.4.3 Global process of negotiation

in the literature and when using agents in a context of e-business negotiation, we can consider three possible cases:

- The use of agents upstream of the negotiation: which manifests itself at the level of research and the collection of information about the various potential partners on the virtual marketplace, more correctly the user activates and configures a broker agent and sends it to the virtual market place during this time other companies / brokers have sent representative agents on the marketplace, agents will try to search and collect information on potential partners before triggering the negotiation process
- The use of the agents during the negotiation: in this case the agent will move to the negotiation process after choosing his potential partner, so we need a negotiation model
- The use of agents downstream of the negotiation: the conclusion of an agreement after a negotiation may turn out later to be incomplete and lacks perfection as both parties hoped .Therefore, there may be false interpretations of the contract findings, supporting manipulations post-finding .In this perspective, the use of intelligent agents downstream of the negotiation can be extremely useful.They can ensure some kind of monitoring over the findings of the negotiation and its applications to ensure the necessary follow-up. In addition, in the perspective of the absence of any problem likely to call into question the findings of the negotiation, the intelligent agent can also be used to collect essential information for the smooth running of subsequent negotiations with the same potential partner. For this, the intelligent agent will try to ensure some kind of monitoring over on the partner in question ,and as soon as a need arises, he will leave to offer his services. Finally, just as in the case of the use of intelligent agents in upstream of negotiation, their downstream uses of negotiation is done in the pure prospect of research and information gathering.

As part of our work, we will focus on the use of intelligent agents in both the upstream of the negotiation and during negotiation. Our work will be be composed of three phases: discovery, selection and negotiation , in which the first two phases are part of the upstream of negotiation and the third one is the main negotiation itself (Figure 4.1).

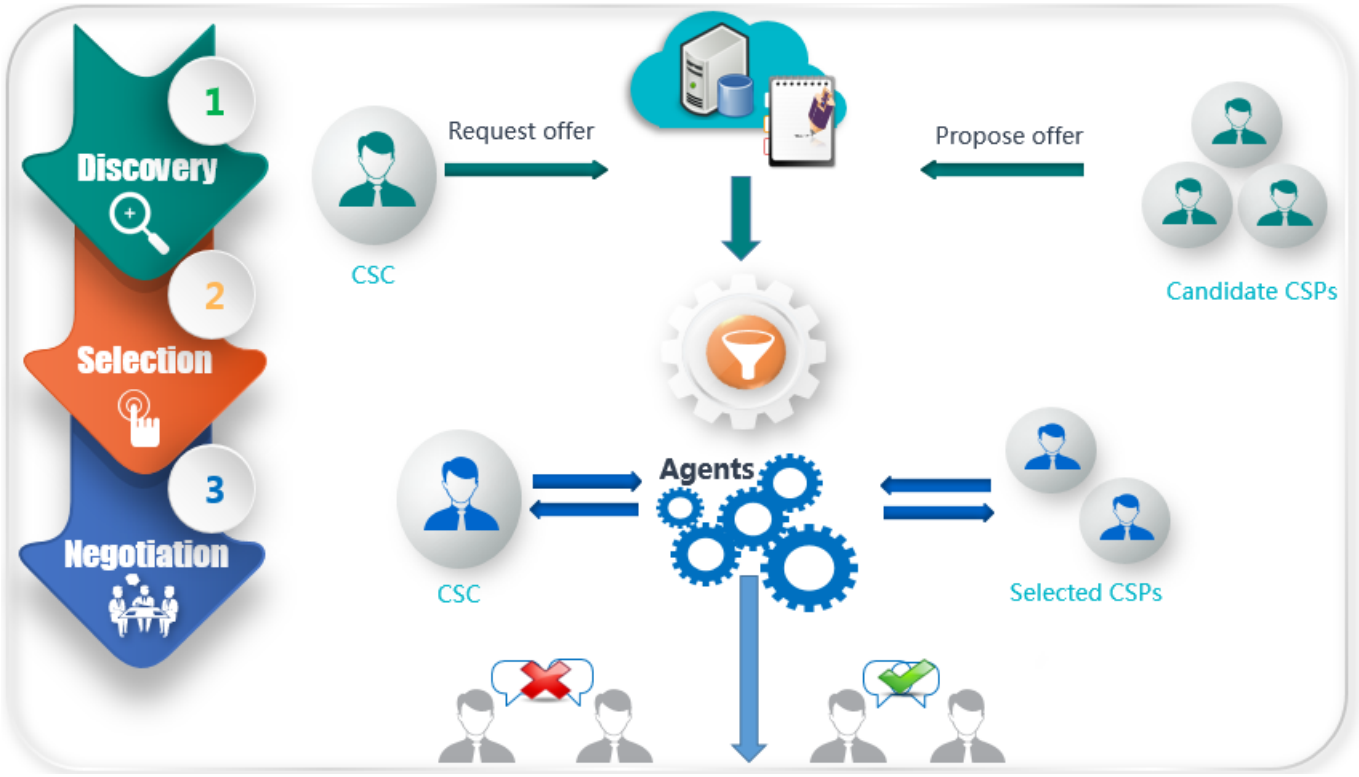

Figure 4.1: General Architecture of the framework's phases

Phase 1 : Discovery

This is an important phase , because it will provide the input to the next one , the output of this phase is the set of CSPs offers after the call for proposals sent by the CSC, both inputs of the call for proposal and the offers are entered by the users.So each user whether it is the Cloud Service Customer or the Cloud Service Providers, will be able to update their offers and requests via a user interface.

Phase 2 : Selection

This phase will use the output of the one that Precedes it as an input for the CSC's selection function , the function will go through all the offers proposed by the set of CSPs candidates, one by one , and then filter them in order to select the best matched candidates. The selection function will include the detailed standards of the demanded SaaS service.The output of this phase will be the CSPs that been selected and with whom the CSC will be negotiating separately.

The discovery and selection phases work together, and we can describe their flow of action as follows : given an initiator agent A, that presents the Cloud Service customer,and another agent B representing a Cloud Service Provider. The agent A starts the discovery phase by sending a call for proposal(cfp) that contains values of the SaaS contract
Agent B , and after it evaluates the call for proposal, using an internal evaluation function "proposal evaluation",responds either with a *rejection* or a *proposal*.

In the case where the agent B responds with a *rejection*,agent A excludes the CSP represented by agent B from entering the selection phase.

In the case where the agent B responds with a *proposal*, it enters the selection phase , then agent A evaluates the proposal and makes a decision based on an internal function"Providers Selection",according to the decision agent A either excludes the CSP from being selected ,or accept the primary offer and awards the CSP presented by agent B , in other words selects it to start the negotiation with.
The discovery and selection process we proposed is illustrated in the sequence diagram below(Figure 4.2)

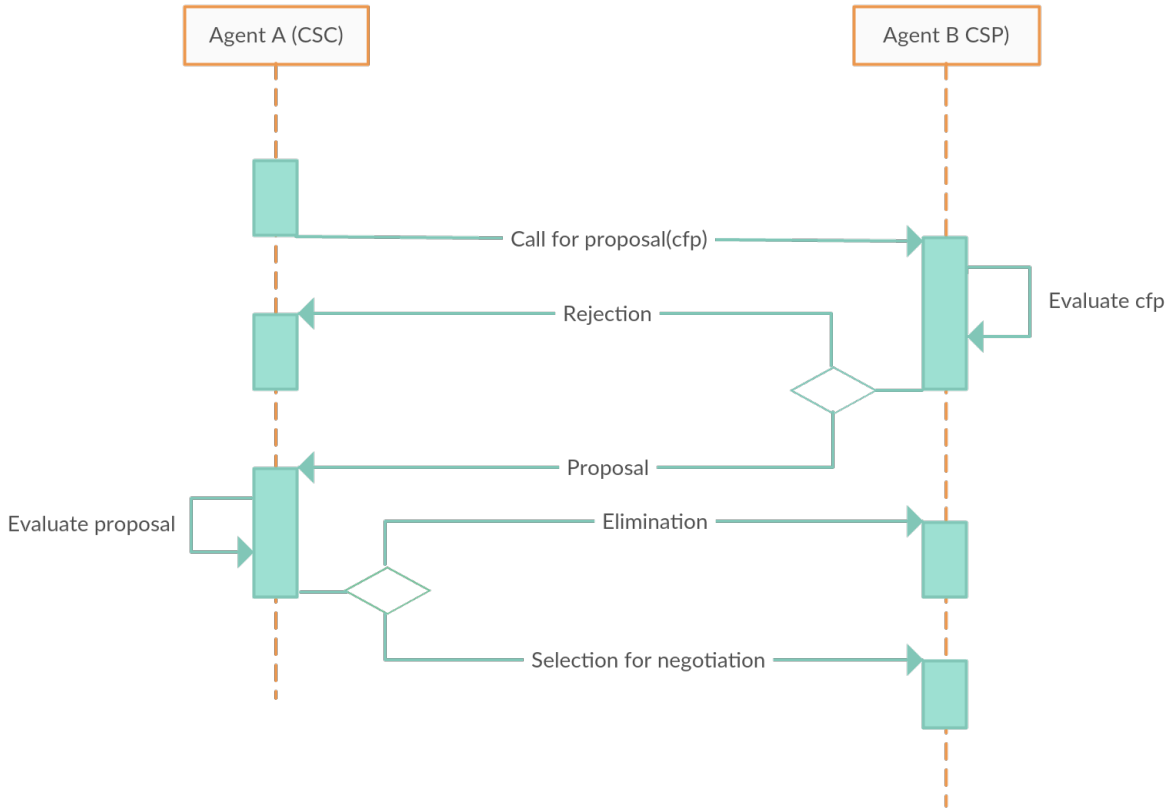

Figure 4.2: The proposed discovery and selection sequence diagram

our work takes into consideration the use of agents upstream of the negotiation ; presented by the discovery and selection phases
The following algorithm 4.1, illustrates the discovery and selection of CSPs as a pre-negotiation process.

Given an initiator agent A representing the Cloud Service Costumer(CSC), an agent B representing candidate cloud service provider (CSPs) among many others, a set of negotiable parameters of Saas contract.

1. The initiator agent A sends a call for proposal (cfp) to the agent B.
2. The agent B evaluates the call for proposal using an internal function “evaluate proposal”.
3. The previous evaluation allows agent B to make a decision, either it responds by a *proposal* or a *rejection* (refuse).
4. **If** response(B) = rejection **Then**The agent A eliminates the agent B from being selected for negotiation.

   Else

   If response(B) = proposal **Then** The agent A evaluates the offer to select agents for negotiation using an internal function “ select for negotiation ” and either ,selects the agent B to start negotiation with, or eliminate agent B from being selected to start negotiation with .

   EndIf

EndIf

Listing 4.1: The proposed Discovery and selection phases

Phase 3 : Negotiation

In this stage, the Cloud Service customer will negotiate separately with each selected Cloud Service Provider(see, Figure 4.3). Then, the outcomes of each session of the negotiation will be compared with each other. So basically the customer is is interested by the service provided by each provider but with some modification in the parameters of the service that means changes in the proposed offer, therefore, and at this level, the negotiators CSC and CSP will use agents that present them on the cloud computing marketplace, the two agents will try to come to an agreement on the parameters of the SaaS contract in question and this by following the CNP standard. The output of this phase is the final output of the system , so the best outcome from the customer’s perspective will be picked up, which will be the agreed value for each parameter in the SaaS contract.

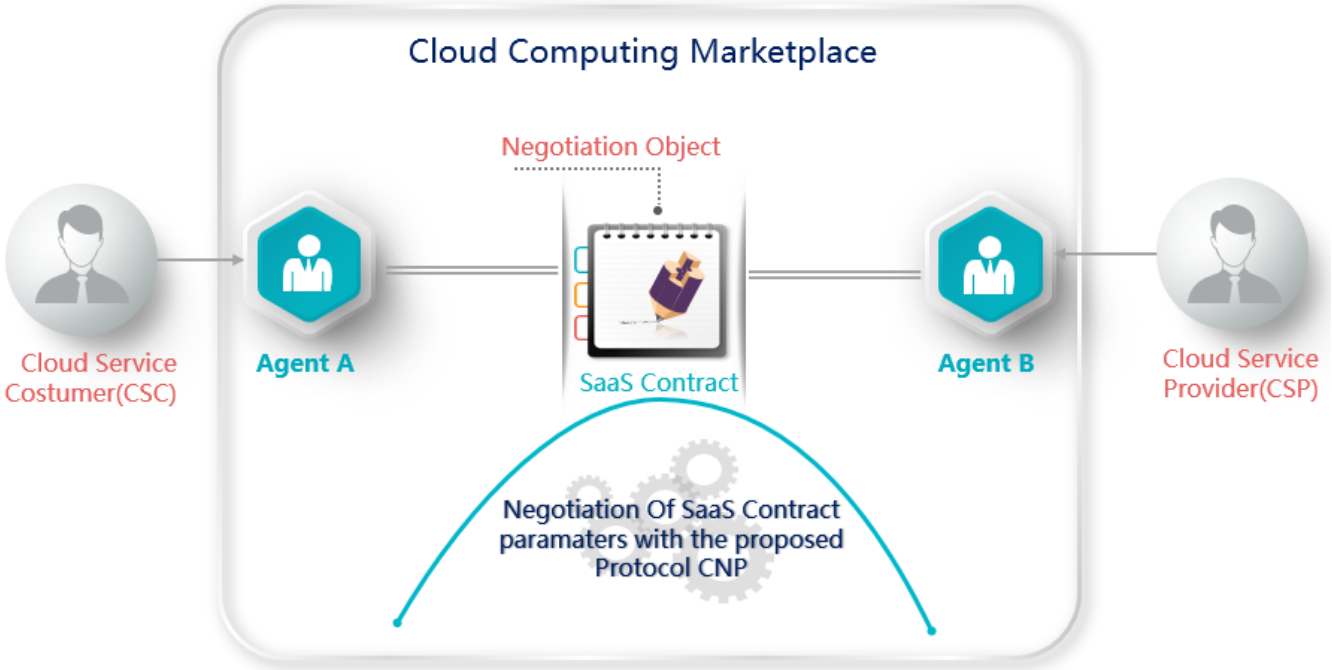

Figure 4.3: Agent-based architecture for SaaS contract's negotiation

We describe the negotiation phase in the following manner : given an initiator agent A, that presents the Cloud Service customer, and that is willing to enter negotiations with another agent B representing a Cloud Service Provider,and which it is interested in its provided service but does not agree with some provided service parameters.The agent A starts the negotiation phase by sending the fist proposal that contains values of the SaaS contract parameters, which it wishes to verify.
Agent B , and after it evaluates the received proposal, using an internal evaluation function, responds either with an *acceptance*, a *rejection* or a *counter-proposal*.

In the case where the agent B responds with an *acceptance*, the negotiation process ends with success.

In the case where the agent B responds with an *rejection*, the negotiation process ends with failure.

In the case where the agent B responds with an *counter-proposal*, agent B evaluates it, using a counter-proposal evaluation function.Based on this evaluation, agent B sends a new proposal or (counter-proposal) to Agent A.
In this manner, the process will continue until stopping condition (criteria) is reached, This condition refer to the number of counter-proposals sent and received by the two agents in the negotiation process.
The negotiation process we proposed is illustrated in sequence diagram below (Figure 4.4)

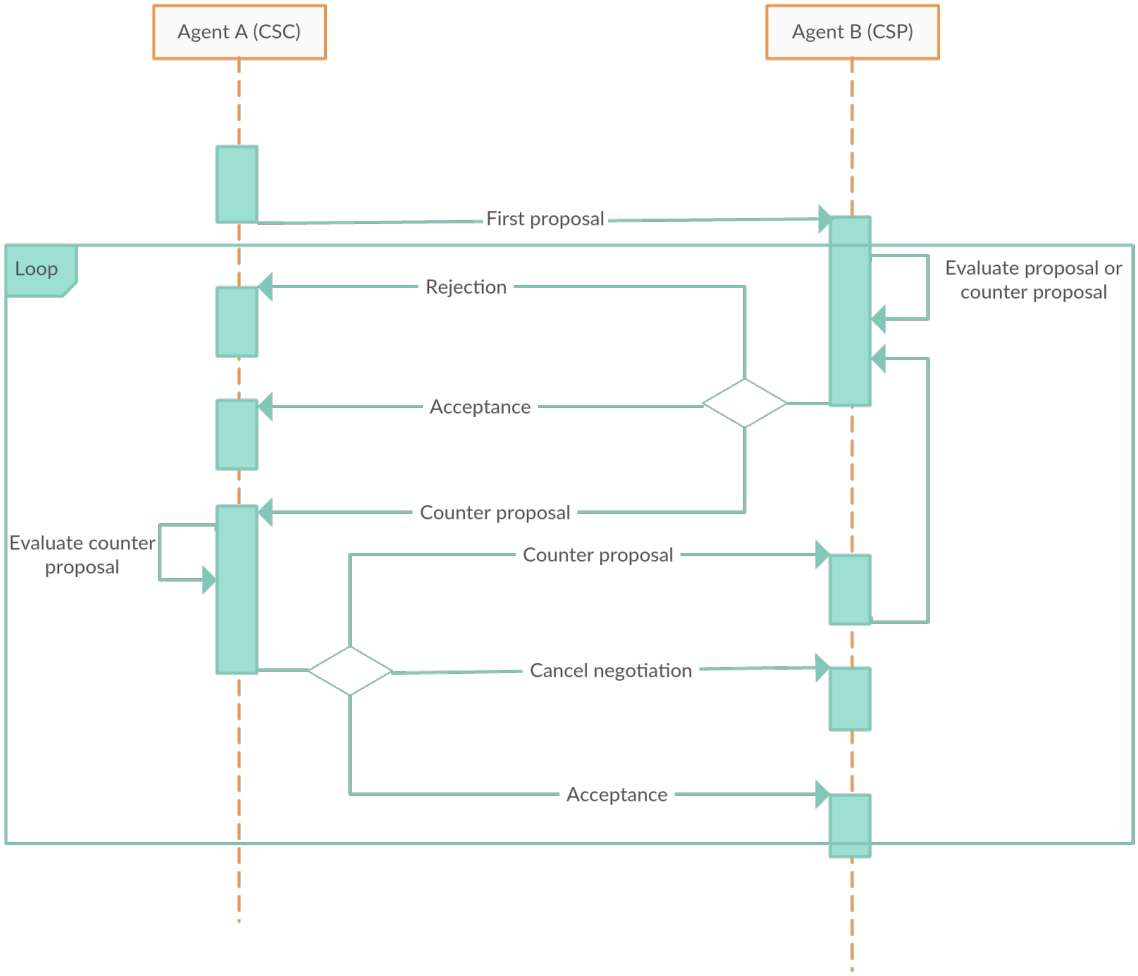

Figure 4.4: The proposed negotiation sequence diagram

The following algorithm 4.2 illustrates the proposed negotiation protocol

Given an initiator agent A , representing the Cloud Service Costumer (CSC), and an agent B representing the selected cloud service provider(CSP) for negotiation, a set of negotiable parameters(issues) of the SaaS contract, a counter X of counter-proposals.

1. Initiator agent A,sends a proposal p_A to agent B, to start the negotiation.
2. Repeat

   a. The agent B evaluates the proposal (or counter-proposal) using an internal function "evaluate proposal".

   b. The previous evaluation allows agent B, to make a decision, either it responds by: a *rejection*,an *acceptance* or a *counter-proposal*.

   c. **If** response(B) = rejection **Then**Agent A ends the negotiation process with failure.

   Else

   If response(B) = Acceptance **Then** End of the negotiation process with success,

   Else

   If response(B) = Counter-Proposal **Then** Increment the counter X,

   If X = limited number of counter-proposals **Then** End the negotiation process with failure.

   Else Agent A evaluates the counter-proposal using an internal function "evaluate counter-proposal", sends a new proposal (Or counter-proposal) and goes to (**a**).

   EndIf

   EndIf

   EndIf

   EndIf

   Until(X = limited number of counter-proposals)OR (End of the negotiation process with failure)OR (End of the negotiation process with success);

Listing 4.2: The proposed negotiation protocol

4.4.4 Identifying the SaaS contract negotiable parameters

SaaS Contracts make it easy to present all legal documentation at once to service users. the content of the contract differs from a service provider to another.Depending on their preferences, SaaS service providers can include any other documentations or conditions.

A customer's room for negotiation of the SaaS contract depends heavily on the provider; smaller providers will be much more open to wholesale changes than the larger providers.

The negotiable parameters(issues)of the SaaS contract are listed in the table 4.1 below.

Table 4.1: Negotiable parameters of a SaaS contract

Parameters	**Description**
Price	The SaaS service charges.
Contract period	The duration of the SaaS contract.
Service availability	Uptime of service in a specific time.
Response time	The service response time which is a measure of its performance.
Intellectual property	The owner of copyright and all rights regarding the service using output.

Note:

The Service Availability and Response Time are parameters of the SLA of the SaaS contract, and we chose them as negotiable parameters for the overall SaaS contract

4.5 Internal evaluation functions

Negotiation is a form of decision-making, where two or more parties, together, explore possible solutions to reach an agreement. In general, negotiation can be classified according to the number of parties involved and the number of negotiated attributes (parameters). in terms of the number of participating parties, the negotiation scenarios can be one-to-one, one-to-many or many-to-many. in terms of the number of negotiated attributes, the negotiation can relate to only one attribute or to multiple attributes.

4.5.1 Constraint Satisfaction Problem (CSP) for negotiation

A constraint-satisfaction problem (often shortened to CSP) has two ingredients. The first is a set of variables, each associated with a set of possible values (called its domain). The other is a set of constraints — a fancy word for rules — that describe relationships among the variables [108].

When we select a value for each variable, we have what's known as an assignment or a state. Solving a CSP means finding an assignment that satisfies all the constraints. A CSP may have any number of solution states

(including zero).

Even if the name is new, the idea of a CSP is probably familiar. For instance, many brain teasers — like Sudoku or crosswords or logic puzzles — are really just constraint-satisfaction problems.

In our work we try to combine the multi-attribute utility theory (cf next section)and the constraints in the values domain to specify a more complex preferences. Using the Multi-Attribute Utility Theory (MAUT) with constraints allows us to express the constraints and preferences for example: the total price of objects to be bought has to be less than 30000DA and preferably less than 2500DA.

4.5.2 Evaluation of proposals and counter proposals

The agents in the proposed protocol use the Multi-Attribute Utility Theory and constraint-based reasoning for the evaluation, the selection, and the generation of proposals and counter proposals.

4.5.2.1 The Multi-Attribute Utility Theory(MAUT)

Multi-Attribute Utility Theory is an evaluation scheme which is very popular by consumer organisations for evaluating products.

The purpose for using utility theory in decision making is to create a mathematical model to aid the process. It gives the decision maker the ability to quantify the desirability of certain alternatives. [109]

It is been used as standard decision making tool in USA and many European countries. As it is multi criteria decision making method and its simplicity and easiness in formulation of model.

Based on the MAUT, the overall evaluation $u(x)$ of an object x is defined as the weighted sum of its evaluation with regard to the relative values dimensions

Combined evaluation is defined by the following function:

$$u(x) = \sum_{n=1}^{n} w_i u_i(x)$$

Here, $u(x)$ is the evaluation of the object on the i-th dimension of value d_i and w_i is the weight that determines the impact of the i-th dimension of value on the overall evaluation ,n is the number of different dimensions of values, and $\sum_{n=1}^{n} w_i = 1$.

Example : To illustrate how to apply the MAUT , the following example is used : A faculty director wants to evaluate his students. In this context, the alternatives to this decision problem are the students and the preferred attributes by the decision maker are the students grades in mathematics (M), physics (P) and language (L). The preferred attributes (M) and (P) are expressed on the definition space $\Omega_{Mp} = [0, 20]$ and the attribute (L) is expressed on the discrete definition space $\Omega_L = \{A, B, C, D, E\}$. The decision maker wishes to compare three students S_a,S_b et S_c whose grades are shown in the following table 4.2.

Table 4.2: Example :Students grades

	Mathematics (M)	Physics (P)	Language (L)
Student S_a	16	16	B
Student S_b	18	18	C
Student S_c	16	15	B

The director uses the same utility function $u_{Mp} : [0, 20] \longrightarrow [0, 1]$ for the criteria (M) and (P). He is perfectly satisfied when a student's grade is worth $1_{Mp} = 20$. On the other hand, if a student's grade is less than or equal to $0_{Mp} = 8$, the director is totally dissatisfied. By definition, $u_{Mp}(8) = 0$ and $u_{Mp}(20) = 1$. Furthermore, he strongly prefers a mark of 16 to 12, whereas he prefers only moderately a mark of 12 (respectively 20) to 8 (respectively 16). Regarding the criterion (L), the director is perfectly satisfied if the student has obtained the mark $1_L = A$ and totally dissatisfied if he has obtained the mark $0_L = E$. By definition, $u_L(E) = 0$ and $u_L(A) = 1$. In addition, the director's level of satisfaction varies linearly between A and E. The utility functions function u_{Mp} and function u_L , (cf. Figure 4.5) model the decision maker's preferences.

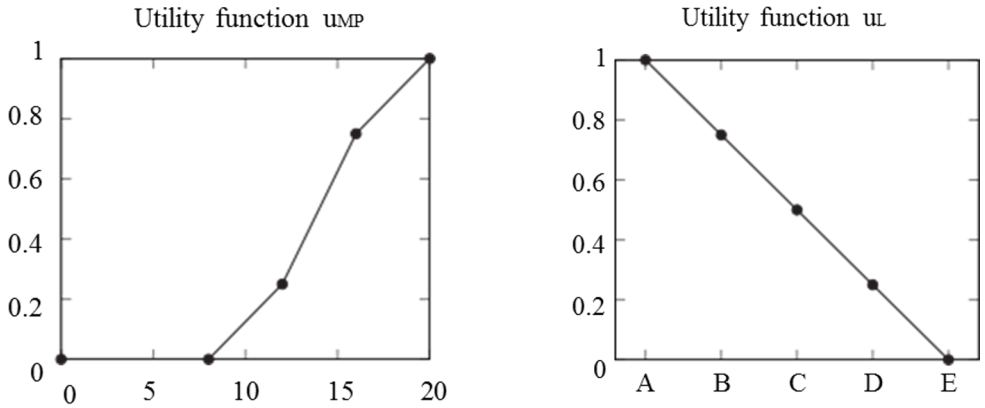

Figure 4.5: Example: Utility functions u_{Mp} and u_L

Using u_{Mp} and u_L, the value of the different criteria can be calculated:

Table 4.3: Example : Utilities values

	Mathematics (M)	Physics (P)	Language (L)
Student S_a	0.75	0.75	0.75
Student S_b	0.875	0.875	0.5
Student S_c	0.75	0.625	0.75

To complete the MAUT model, it is necessary to determine the utility associated with each student.For

that, the director uses a weighted sum S_w whose weight vector is $w = (w_M, w_P, w_L) = (0.3, 0.3, 0.4)$.This choice reflects the director's desire to give equal importance to mathematics and physics in his assessment.In addition, the weight given to languages(40% of the weighting) is high because the decision-maker prefers students with a balanced profile rather than high-performing students only in science.The use of S_w provides the following results :

Table 4.4: Example : Utilities calculated using a weighted sum

	Student S_a	Student S_b	Student S_c
Utility	0.75	0.725	0.7125

Thus, in the sense of this MAUT model built from u_{Mp},u_L and S_w, the director's preferences with regard to his students are : $S_a \succsim S_b \succsim S_c$.

After presenting the principle of MAUT , we will, in what follows, show how we applied this theory for the evaluation of a proposals or a counter proposal

4.5.2.2 Applicability for our problem

We suppose that the agents of our architecture use the MAUT and constraint-based reasoning for the evaluation, the selection and the generation of proposals and counter proposals.The attributes of MAUT correspond to the negotiable parameters(clauses) of the SaaS contract.
When agent A receives a proposal p_B of another agent B, and for B to be considered by agent A it has to satisfy all the constraints that consist of : the proposed value of each negotiable parameter, x_n,has to belong to a domain D_{xn} as it was specified by the agent A (this domain is a private information).Therefore, the evaluation of a proposals,composed of a set n of parameters $\{ x_1, x_2, \ldots x_n \}$ is defined by the following function:

$$U(x_1, x_2, \ldots x_n) = \sum_{i=1}^{n} w_i u_i(x_i); \sum_{i=1}^{n} w_i = 1$$

Such that x_i is the i-th parameter of negotiation, u_i is the utility function of the i-th parameter and w_i is the weight of the i-th parameter's value.

Evaluation of a proposal /counter proposal

In our work, We consider a proposal as a set of five parameters (attributes) denoted by: P(Price, Period, Availability, ResponseTime, IntellectualProperty).
Because we are using MAUT for the evaluation of proposals and counter proposals, each agent and depending on its preferences, assigns a weight W_i to each attribute x_i(this weight is a matrix that presents how important each attribute for these agents) , so that $\sum_{i=1}^{n} w_i = 1$ /n=5, which means that the weights of all attributes must sum up to 1 In the same way, each agent performs an evaluation U on an attribute x_i, denoted by $U(x_i)$ according to its

private information. For example, when it receives a proposal where the attribute Period is 9, it performs the following evaluation:

if *Val(Period)<9* **then**
  $U(x_2) \longleftarrow 3$
else
  $U(x_2) \longleftarrow 5$
end

Here, x_2 denotes the parameter ***Period***.

So, at the time of receiving a proposal(counter proposal),an agent performs a computation of its utility(denoted by U_p), according to the received proposal, in the form of a weighted sum of evaluations given to attributes of a proposal(counter proposal).Next, it compares it with its own Absolute Utility denoted by U_A), where:

$$U_A\text{-}U_P \leq 25$$

Therefore, a proposal (counter proposal)could be accepted:

- In the *discovery* phase: if providers accept the request of the customer , they send their first proposal.
- In the *selection* phase: if the customer accepts the first proposal of the providers,it selects those providers to enter the next phase with them(negotiation)with it.
- In the *negotiation* phase: if the customer or the providers accept the proposal or the counter proposal they either answer by another proposal , counter proposal or if they find an intersection of preferences they end the negotiation with an agreement.

Note:

The absolute utilityU_A is computed only once and it is part of the agent's private information.On the other hand,the proposal utility $P(U_P)$ is computed each time a proposal(counter proposal) is recieved.

4.5.3 Generation of proposals(counter proposals)

Proposals generation is the main process of decision making which governs the progress of the negotiation as well as its results.It contains the search for possible solutions based on individual preferences, which motivate the the negotiation participants to reach an accord that belongs to the intersection of their preferences [110].

In the context of our work, in order to generate a proposal(counter proposal),the negotiation agent uses the constraint-based reasoning to reach a possible solution taking into account the criteria it received just before the proposal(counter proposal) is expressed.

4.6 Negotiation agent architecture

we assume that the negotiating agents are homogeneous agents in other words they contain the same type of components.
We motivate this choice by the need of the simplification of different interactions and the need for the unification of the negotiation mode undertaken individually by agents.

Our agents are cognitive, they are made of the internal part responsible for the decision making. As for the party responsible for the communication with the user, it is performed by a user interface component. The interaction with the other agents, among other things the other negotiation agent(in the case of provider agent,the interaction is with a single consumer agent), is carried out by receiving(perception) and transmitting(action) components. We represented the negotiation by a "Processing" component which is concerned with both pre-negotiation(discovery and selection) and negotiation. It is the essential part on which our work is based. The architecture is shown schematically in figure 4.6.

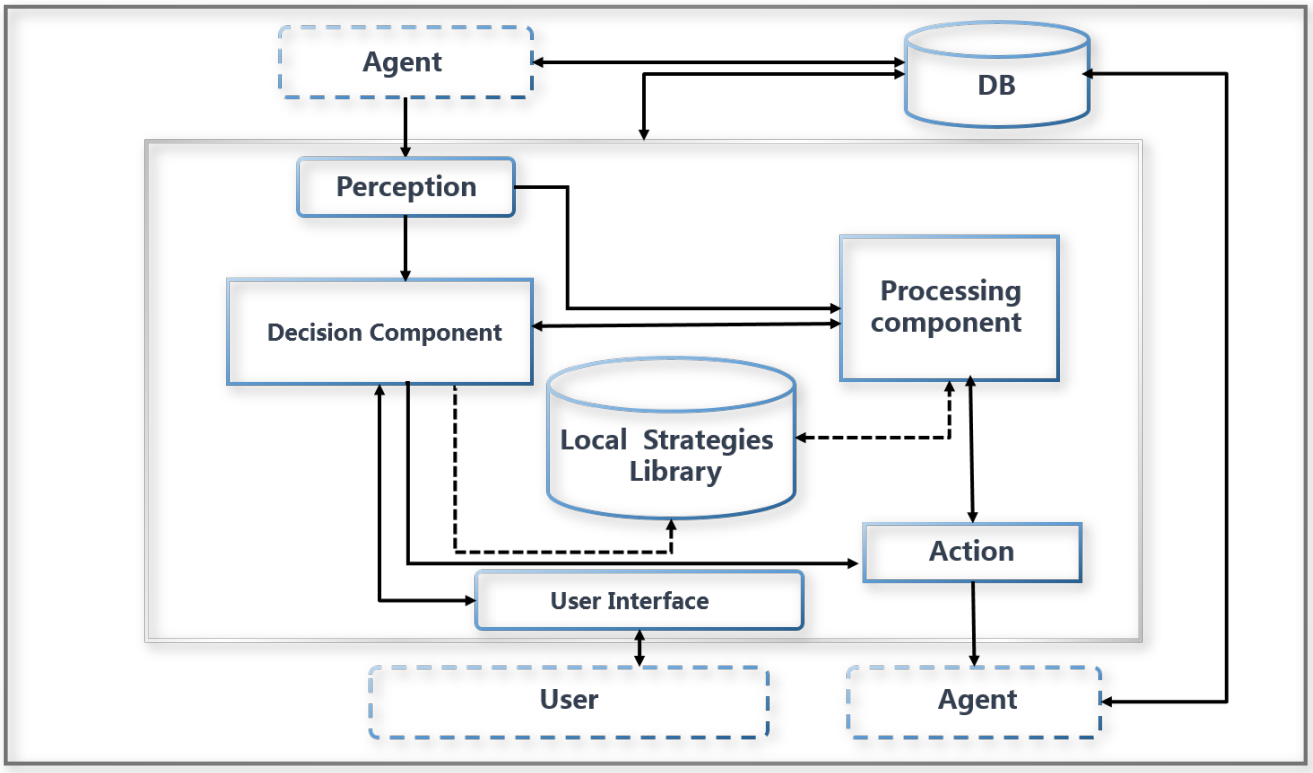

Figure 4.6: Negotiation agent architecture

4.6.1 The Perception/Action components

The Perception/Action components process receptions and messages. If it is conflicting information (proposals, counter-proposal or critics), the message is then sent directly to the negotiation component that will process it. Otherwise, it will be sent to the decision component.

4.6.2 The Decision component

The decision component is the most important of the components next to the processing component. It determines the behaviour of the agent towards the received messages and also indicates the appearance of the conflict.

4.6.3 The Processing component

The processing component is based essentially on reasoning. the reasoning mode is identical for all the agents since they are homogeneous. it also contains the methods that describe the behaviour of the agent, including the achievement of goals, the choice of strategies, etc. The role of the processing component is to resolve the conflict by changing the decision component in the case where the carries conflicting information.
Therefore, this component expresses the autonomy of the agent by his ability to make preferences on the choice of solutions (goals) and how to achieve them (strategies).

4.6.4 The User Interface component

The agents presented in the architecture are supposed to represent a cloud service customer and cloud service providers. Users work cooperatively to solve common problems. To do this, each user provides the necessary information to the representative agent (negotiation agent). The operation is performed by the interface component. In the opposite case, the delegated agent periodically informs his delegate about the state of negotiation.

4.6.5 The Local Strategies Library

Functionally, we used this component to provide a local strategy to the processing component. This component presents a set of strategies functions used by our model. Each strategy belonging to the component is instantiated by its designation and a list of arguments.Among the arguments, we can consider: the utility function and the results provided by the decision component.

4.7 Conclusion

In this chapter we have presented the problematic of automated negotiation as well as the interest of such a policy in the SaaS contract initiative. After discussing the applicability of CNP negotiation protocol to our work context and the general running of the negotiation process, we have detailed the proposed protocol by specifying the evaluation functions used and the reasoning used for the generation of proposals and counter proposals.
Then, we have presented the architecture of a negotiation agent while explaining its different components.
The following chapter presents the aspects of implementation our model. We used the JADE platform for this purpose.

Chapter 5

IMPLEMENTATION OF THE PROPOSED MODEL

5.1 Introduction

In the previous chapter of this master's thesis, we proposed our multi-issue negotiation model for SaaS contracts on the cloud, based on CNP negotiation protocol and multi-agent systems.
In this chapter, we will first present our work environment , we will give an overview of the programming language as well as the software environment we used for the implementation and simulation of our model. Then, we will demonstrate the process of weights, constraints and preferences assignments for each agent parameters values and the absolute utility computing. Next, we will present a series of simulations to highlight our proposition. And, finally we will end this chapter with a conclusion.

5.2 Programming language and development environment

5.2.1 The Programming language: Java

The Java language is an object-oriented programming language created by James Gosling and Patrick Naughton employees of Sun Microsystems with the support of Bill Joy (co-founder of Sun Microsystems 1982), officially presented on May 23, 1995 at SunWorld. Created in its beginnings to run especially on heterogeneous environments (mainly embedded), its success is mainly due to its integration with large internet browsers public [111]. It allows an object-oriented (like SmallTalk and, to a lesser extent, C ++), modular (ADA language) programming and uses a syntax very similar to that of the C language. [112]
In addition to its object orientation, the Java language has the advantage of being modular (we can write generic portions of code, which means that is usable by several applications), rigorous (most errors occur at compile time). and not at runtime) and portable (the same compiled program can run on different environments). On the other hand, Java applications have the defect of being slower at run time than applications programmed in

C for example. [112]

This software, is written in Java, and the justification is that:

- Agents developed under jade are fully written in Java. This language is therefore imposed as a consequence of the later choice,
- Java is a cross-platform language that allows designers, using the principle of "write once, run every where", to write code that can work in any environment (whatever the operating system is),
- Java has a rich class library including Graphic User Interface (GUI) management (window, dialog box),
-

5.2.2 Development environment : Eclipse

Eclipse [113] is an extensible, universal and versatile integrated development environment(See Figure 5.1, potentially enabling the creation of development projects that implement any programming language. The application is written in Java (with help of the SWT graphic library, IBM), and this language, through specific libraries, is also used to write extensions. Its specificity comes from the fact of its architecture, all the functionalities of this software workshop are developed as plugin.

There are two main reasons for considering Eclipse as a foundation for client applications:

- The quality of Eclipse : the Eclipse development environment has imposed itself by its reliability and quality of its graphical interfaces. The massive use of the Eclipse development environment by the Java developer community and its use as a base for commercial products (WebSphere Studio, SAP NetWeaverStudio ...) have made it possible to test the Eclipse framework....
- The open source availability of the Eclipse framework.

In our work we used the version 2019-03 (4.11.0).

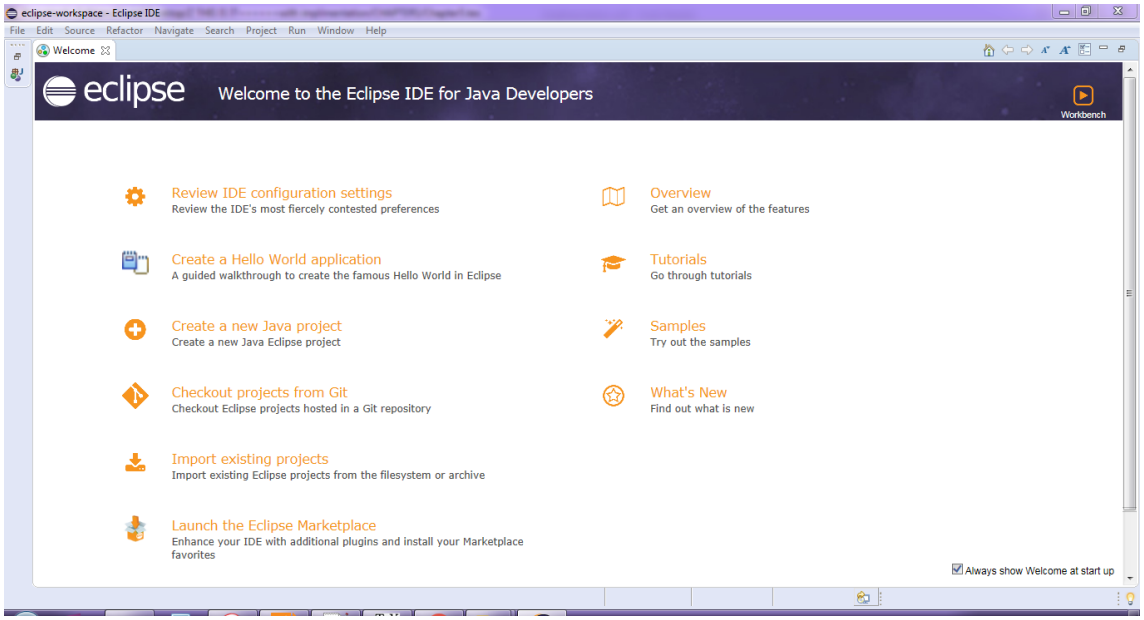

Figure 5.1: The Eclipse development environment.

5.2.3 MAS observation and visualization tool: the JADE platform

The first software developments, that eventually became the JADE platform, were started by Telecom Italia (formerly CSELT) in late 1998, motivated by the need to validate the early FIPA specifications. [114]

JADE is a software platform that provides basic middleware-layer functionalities which are independent of the specific application and which simplify the realization of distributed applications that exploit the software agent abstraction [115]. A significant merit of JADE is that it implements this abstraction over a well-known object-oriented language, Java, providing a simple and friendly API. The following simple design choices were influenced by the agent abstraction. [114]
JADE has three basic components :

- An execution environment in which agents can "live";
- A class library that programmers can use to develop their agents;
- And a suite of graphical tools for managing and administering active agents.

In addition, JADE can be distributed across multiple hosts and consists of three basic agents:

- AMS (Agent Management System) that manages the agent life-cycle, maintains a list of all agents, and controls access and usage the communication channel between agents;
- DF (Directory Facilitator) which records agent descriptions and the services they offer;
- and ACC (Agent Communication Channel) that manages agent-to-agent communication.

The choice of this platform is justified by the following points: [116]

- JADE simplifies the implementation of an MAS through a FIPA compliant Middleware which is a class library that users can use and extend as well as a set of graphical tools that allow debugging and administration of MAS to be designed;
- JADE provides a transparent communication through the exchange of messages in the standard FIPA-ACL language;
- JADE decreases the programming effort because it implements two agents: Directory Facilitator (DF) and Agent Management System (AMS) whose features are useful to our application.

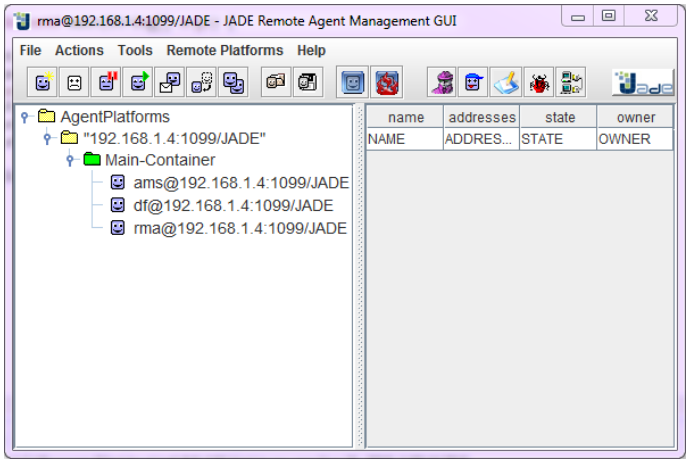

Figure 5.2: Java agent development environment.

5.3 Weights, constraints and preferences on parameters values of each agent

According to the previous sections , we have two negotiating agents , one that represents the Cloud Service Customer(CSC), which we name **customer agent**, and another that represents each one of the Cloud Service Providers(CSPs), which we name **provider agent**. Both agents use the constraint based reasoning. For that, each agent has **constraints** and **preferences** on values of which the negotiable parameters of the SaaS contract need to take. In what follows, we will demonstrate the constraints and preferences of each agent mentioned below.

5.3.1 Constraints and Preferences

5.3.1.1 The customer agent

We suppose that the customer agent has the following constraints (See Table 5.1) on the values that the five negotiable parameters of the SaaS contract need to take.

Table 5.1: Constraints and prefrences of the customer agent

Parameters	**Constraint**	**Preference**
Price	$\leq$ 400 $	200 $
Contract period	$\geq$ 20 month	24 month
Service availability	$\geq$ 80 %	99 %
Response time	$\leq$ 40 ms	20 ms
Intellectual property	/	True

We will see later, that the negotiating agent is based on its constraints in order to make decisions.

5.3.1.2 The provider agent

We suppose that the customer agent has the following constraints (See Table 5.2) on the values that the five negotiable parameters need to take.

in order to evaluate the SaaS contract parameters it is necessary to build a scale that represents the characteristics of parameter levels.
A scale of 0 (the worst) to 10 (the best) serves as an evaluation measurement.

Table 5.2: Constraints and prefrences of the provider agent

Parameters	**Constraint**	**Preference**
Price	$\geq$ 300 $	350 $
Contract period	$\leq$ 20 month	15 month
Service availability	$\leq$ 80%	70%
Response time	$\leq$ 40 ms	20 ms
Intellectual property	/	False

5.3.2 Evaluation scales of negotiable Parameters

In this part, we assign for each negotiating agent and for each negotiable parameter a scale that allows to evaluate(select)it.

And for that, we will present a scale interpretation of each one of the negotiable parameters, in a form of a pseudo algorithm which shows how both agents assign an evaluations of 10 to 0 to each parameter values.

5.3.2.1 Scales for the customer agent

a.The Price evaluation

```
if (Price <= 400) then
    if (Price = 200) then
        U(Price)← 9;
    else
        if (Price < 200) then
            U(Price)← 10;
        else
            if (Price > 200)and(Price < 400) then
                U(Price)← 5;
            else
                if (Price >= 400) then
                    U(Price)← 0;
                end
            end
        end
    end
end
```

b.The Contract Period evaluation

```
if (Contract Period >= 20) then
    if (Contract Period = 20) then
        U(Contract Period)← 5;
    else
        if (Contract Period = 24) then
            U(Contract Period)← 10;
        else
            U(Contract Period)← 0;
        end
    end
end
```

c.The Service Availability evaluation

```
if (Service Availability >= 80) then
    if (Service Availability = 80) then
        U(Service Availability)← 5;
    else
        if (Service Availability = 99) then
            U(Service Availability)← 10;
        else
            U(Service Availability)← 0;
        end
    end
end
```

d.The Response Time evaluation

```
if (Response Time <= 40) then
    if (Response Time = 20) then
        U(Service Availability)← 8.5;
    else
        if (Response Time < 20) then
            U(Service Availability)← 10;
        else
            if (Response Time > 20) and (Response Time < 40) then
                U(Service Availability)← 5;
            else
                if (Response Time > 40) then
                    U(Response Time)← 0;
                else

                end
            end
        end
    end
end
```

e.The Intellectual Property evaluation

```
if (Intellectual property = True) then
    U(Intellectual property)← 10;
else
    U(Intellectual property)← 0;
end
```

5.3.2.2 Scales for the provider agent

a.The Price evaluation

```
if (Price >= 300) then
    if (Price = 350) then
        U(Price)← 10;
    else
        if (Price = 300) then
            U(Price)← 5;
        else
            U(Price)← 0;
        end
    end
end
```

b.The Contract Period evaluation

```
if (Contract Period <= 20) then
    if (Contract Period = 15) then
        U(Contract Period)← 10;
    else
        if (Contract Period = 20) then
            U(Contract Period)← 5;
        else
            U(Contract Period)← 0;
        end
    end
end
```

c.The Service Availability evaluation

```
if (Service Availability <= 80) then
    if (Service Availability = 70) then
        U(Service Availability)← 10;
    else
        if (Service Availability = 80) then
            U(Service Availability)← 5;
        else
            U(Service Availability)← 0;
        end
    end
end
```

d.The Response Time evaluation

```
if (Response Time <= 40) then
    if (Response Time = 20) then
        U(Service Availability)← 8.5;
    else
        if (Response Time < 20) then
            U(Service Availability)← 10;
        else
            if (Response Time > 20) and (Response Time < 40) then
                U(Service Availability)← 5;
            else
                if (Response Time > 40) then
                    U(Response Time)← 0;
                else
                end
            end
        end
    end
end
```

e.The Intellectual Property evaluation

```
if (Intellectual property = True) then
    U(Intellectual property)← 10;
else
    U(Intellectual property)← 0;
end
```

5.3.3 Negotiable parameters weights

Since the proposed protocol uses the multi attribute utility theory (MAUT)for the evaluation of proposals and counter proposals, each agent, according to its interests, assigns each parameter a weight w_i that demonstrates the importance of one parameter over another.

The Table 5.3 represents negotiable parameters weights for each agent.

Note:

For each agent , $\sum_{i=1}^{n} w_i = 1$ /n= the number of negotiable parameters, in our case n = 5.

Table 5.3: Negotiable parameters weights for each negotiating agent

Parameter	Weights	
	Customer Agent	Provider Agent
Price	0.3	0.3
Contract Period	0.15	0.2
Service Availability	0.25	0.15
Response Time	0.1	0.1
Intellectual Property	0.2	0.25

5.4 Calculating absolute utility U_A for each agent

In our protocol, we supposed that each agent has its constraints and preferences on the values of negotiable parameters. Each negotiating agent, computes for once its absolute utility that we noted by U_A.Thereafter, that utility is been based on for decision making regarding a received proposal(or a counter proposal). So before starting to apply the proposed protocol, we will first calculate the U_A for each agent(Customer agent and provider agent).

a.Calculating U_A for the customer agent

Since the values associated with the negotiable parameters are the preferable values then the evaluation of each parameter is 10 (see the scales above).Thereby,
U_A = 0.3*10+.15*10+.25*10+0.1*10+0.2*10 = 10

b.Calculating U_A for the provider agent

In the same way, Since the values associated with the negotiable parameters are the preferable values then the evaluation of each parameter is 10 .Thereby,
U_A = 0.3*10+.15*10+.25*10+0.1*10+0.2*10 = 10

Throughout the negotiation course proposals(offers) and counter proposals will be transmitted, and in order to make a decision, they need to be evaluated.
In order to evaluate proposals and counter proposals, Proposal utility U_P needs to be calculated. A proposal can take a form of a tuple of five attributes P= (Price,Contract Period, Service Availability, Response Time, Intellectual Property). In case a provider agent(provider or customer) sends an initial proposal(offer) with these values : P= (350,20,80,15,true)

U_{pi} =5*0.3+ 5*0.15+5*0.25+10*0.1+10*0.2

$\Rightarrow U_{pi} = 6.5$

5.5 Results and Simulations

we will run the simulation that is going to follow the three phases that we mentioned in our proposed work which are: discovery, selection, and negotiation.

Also, through the simulation we will introduce the different interfaces of our software which will mainly concern the customer User Interface(Customer UI) and the Providers User Interface(Providers User Interface (UI)) .

5.5.1 Discovery Phase

5.5.1.1 Customer side

Before the discovery phase takes place the costumer has to inform that it is ready to start the simulation(See Figure 5.3).

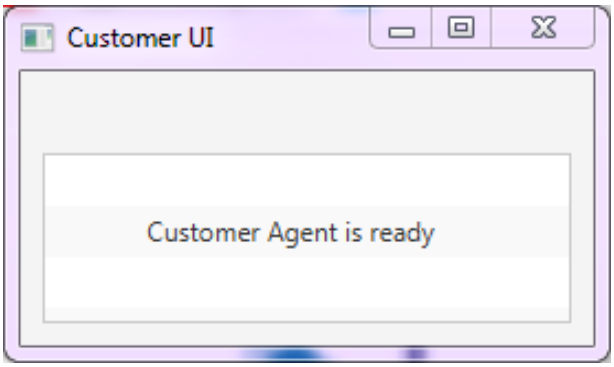

Figure 5.3: Customer agent is ready interface.

After the customer agent gets ready, the discovery phase starts.

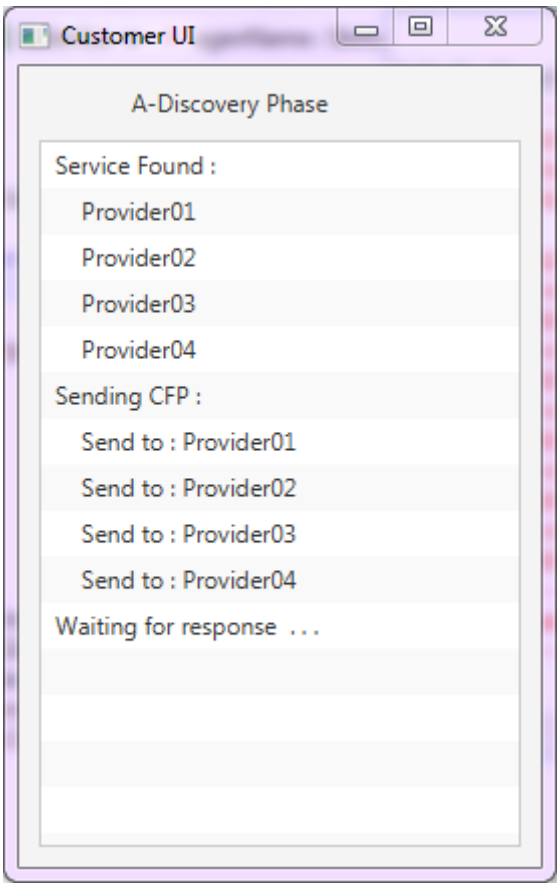

Figure 5.4: Discovery phase from the customer agent side

Figure 5.4 shows the discovery phase, where the customer agent searches for service providers, lists them if found, then it sends a Call For Proposal (CFP) for each one of provider agents and then waits to receive a response from them.
the simulation shows that the customer agent discovered 4 providers and sent CFPs to all of them.

5.5.1.2 Provider side

In order for the provider agents to be discovered, they first need to inform that it they are ready, then, they have to register the service they offer. After they receive the CFP sent by the customer agent each one of them decides whether to send an offer or not.

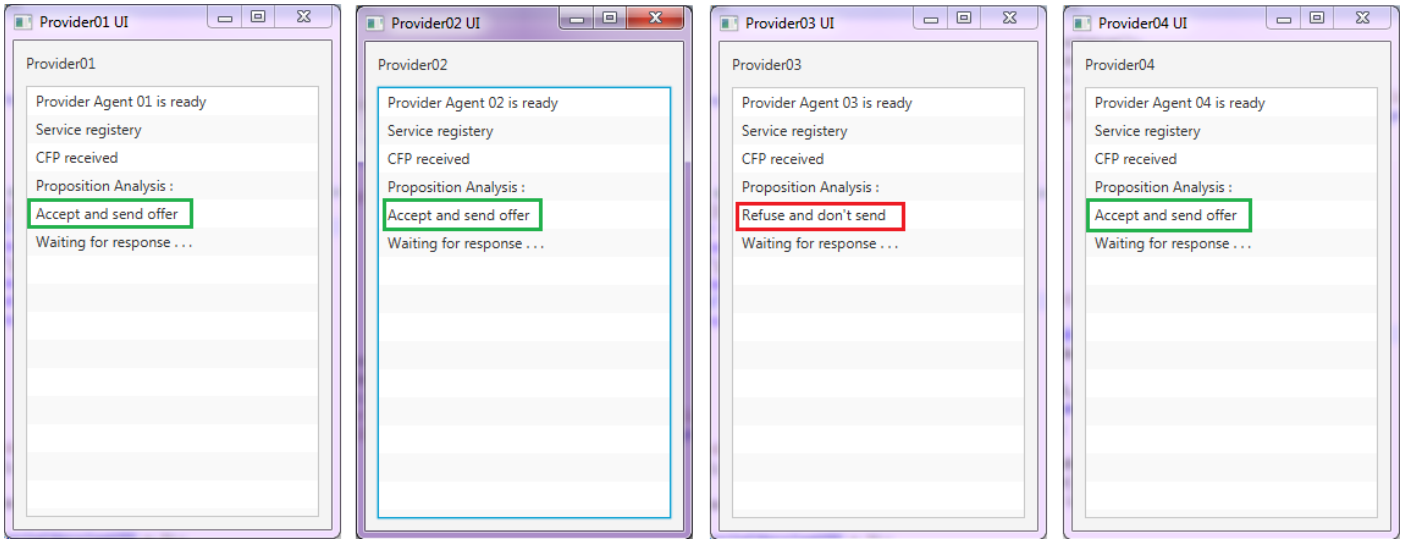

Figure 5.5: Discovery phase from the provider agents side.

Figure 5.5 represents the four provider agents that have been discovered by the customer agent, Provider01, Provider02, Provider03, and Provider04.

In this simulation and after all the provider agents receive the CFP, they Evaluate it.

Provider01, Provider02, Provider04 decided to accept the CFP and send their offer to the customer agent , while Provider03 decided to refuse it and not send an offer.

After they send their responses they wait for the response of the customer agent.

5.5.2 Selection Phase

In this phase the customer agent and if it receives offers from the provider agents , it evaluates them and selects the providers it intends to start negotiation with.

Figure 5.6 shows the received offers from the three providers and their evaluation.

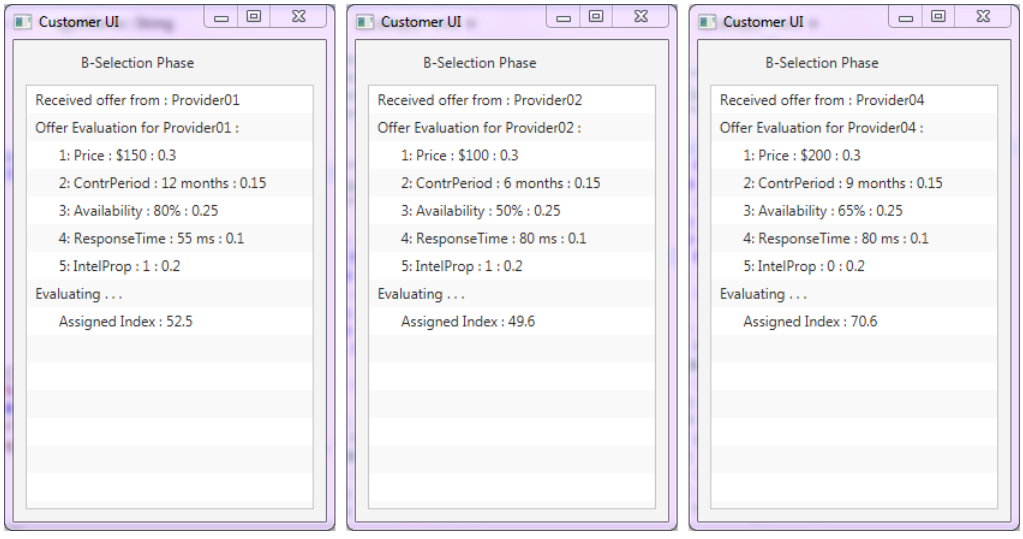

Figure 5.6: Received offers and their evaluation.

After assigning an evaluation value(Index), the customer agent selects according to the result the providers of which it will start the negotiation with.

In the case of this simulation(See Figure 5.7) Provider01 and Provider04 were selected, whereas Provider02 wasn't.

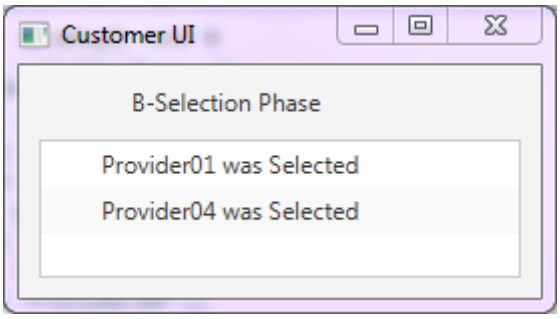

Figure 5.7: Selected provider agents.

5.5.3 Negotiation Phase

After selecting the providers, the customer agent and providers will start negotiating.

5.5.3.1 Provider side

In Figure 5.8, Provider agents are informed that they have been selected to carry on negotiations with the customer agent

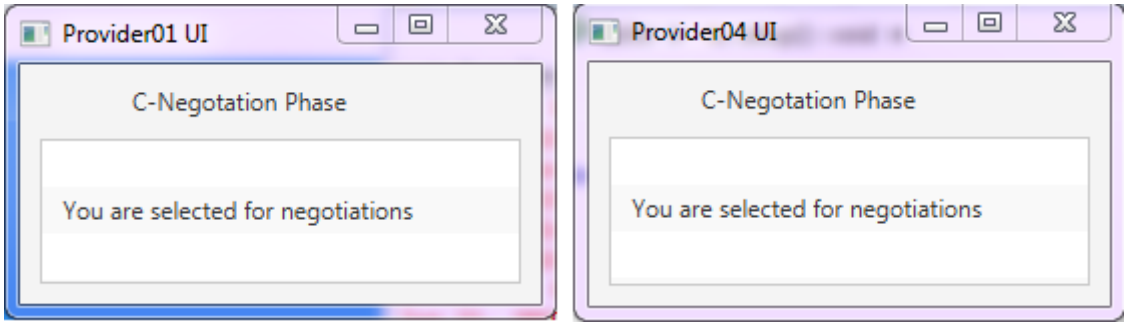

Figure 5.8: Selected providers for negotiations.

At the same time, negotiations start by receiving the counter proposals from the customer. Then each provider agent evaluates it, assign them an index.After that they generate and send counter proposal to the customer agent and wait for its response as shown in Figure 4.4

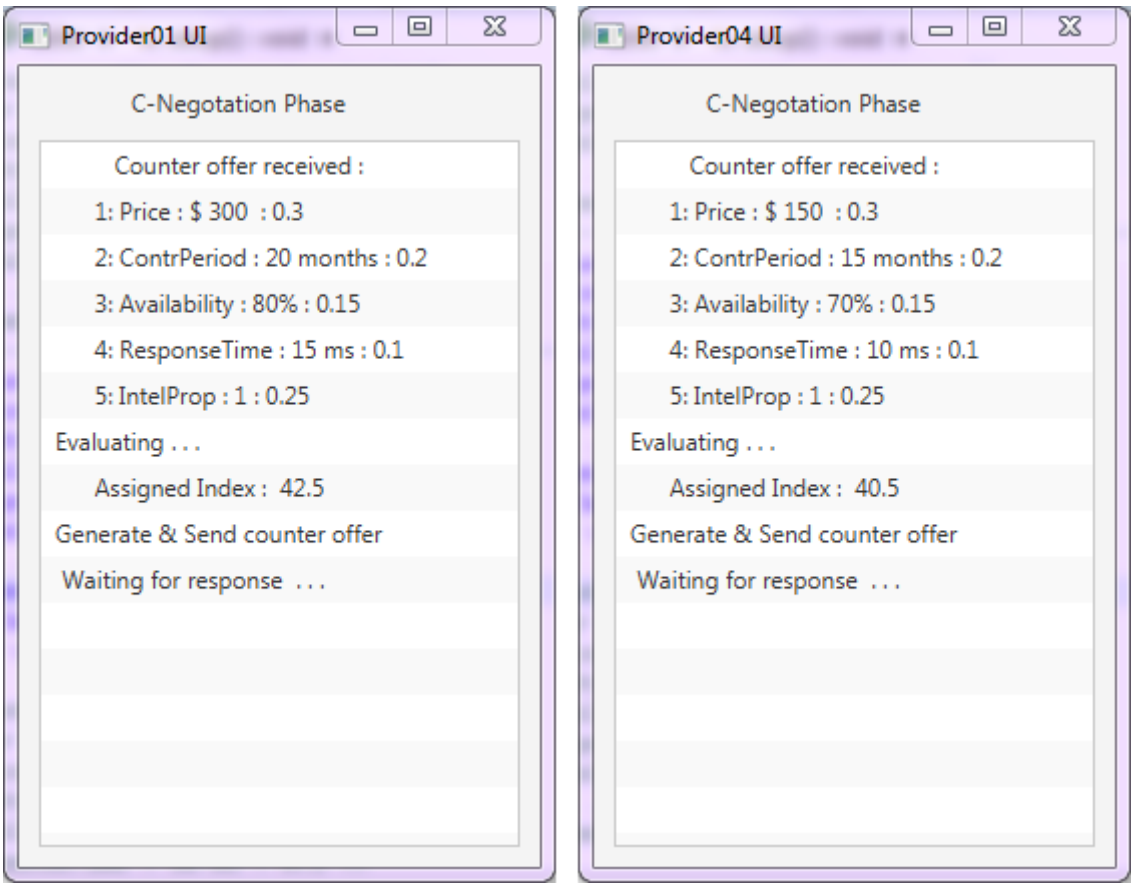

Figure 5.9: negotiations from the povider agents side.

The same process repeats until the negotiations are completed and each agent receives the result of negotiations.

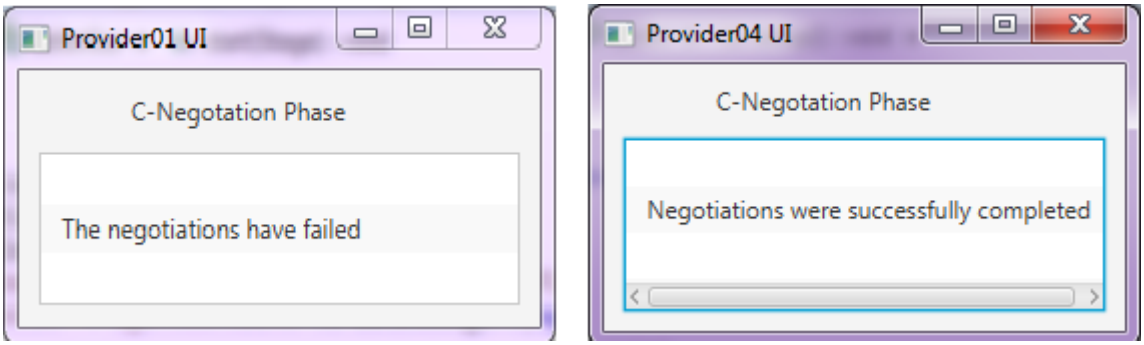

Figure 5.10: Received result of negotiations.

5.5.3.2 Customer side

After selecting Provider01 and Provider04, the customer begins negotiating with both of them , and starts by sending them counter offer(See Figure 5.11).

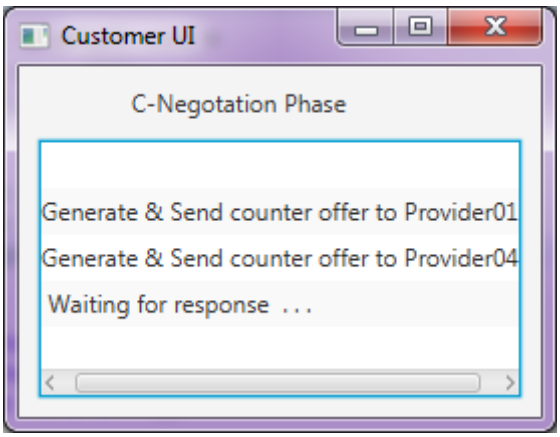

Figure 5.11: The begining of negotiations.

After receiving the counter offer from both providers the customer agent and as shown in Figure 5.12 evaluates the counter offers and assigns them an index.

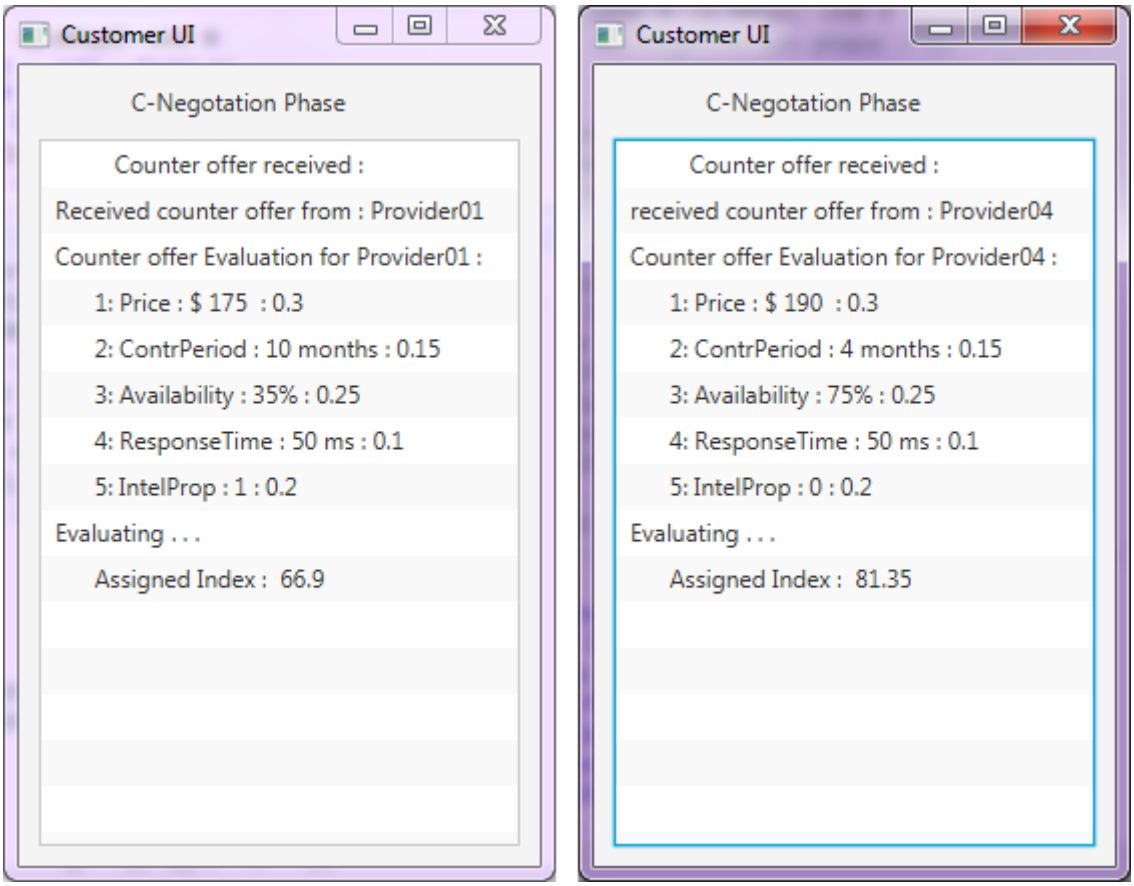

Figure 5.12: The evaluation of received counter offers.

After evaluating the counter offers the customer agent again generates counter offer for both provider agents Provider01 and Provider04 (See Figure 5.11).

the negotiation process continues in the same way until the negotiations are done and one of the providers is chosen.

in this simulation, the customer agent and after negotiating with both provider agents chooses Provider04 to sign the SaaS contract with, and informs both providers of the negotiation result (See Figure 5.13).

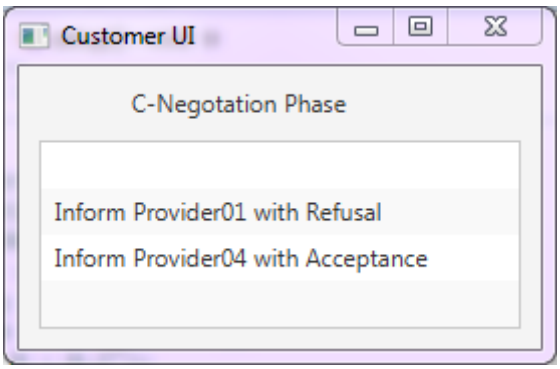

Figure 5.13: The end of negotiations.

5.6 Conclusion

In this chapter, we have tried to implement all the ideas that characterize the model we have proposed. we used Eclipse as a development environment . And to implement our system, we chose the JADE platform. This platform simplifies the development of multi-agent systems while providing a complete set of services and agents. The results obtained show that the use of our model makes it possible to give better outcome for the overall negotiation and high performance for the course of negotiation between the cloud service client and service providers.

GENERAL CONCLUSION

With more and more businesses turning to the Internet, many of them look up to cloud computing as a technology for solutions to some of their problems. But negotiation of cloud contracts(particularly SaaS contracts) has not been taking a speed development in the fast world of internet . Although there were some works which emphasized the need for negotiation in the Cloud Computing, none of them provided a practical solution that considers the negotiation that needs to support multi-issue(attribute) negotiation with the possibility of giving preferences for each issue and that the negotiation needs to be automated, meaning it has to be run by an intelligent agent following a specific protocol,and that is the best way to improve negotiation speed. Our thesis aims to serve this objective.

Throughout this master's thesis, we presented the different technologies needed to propose a formal approach for automatic negotiation between agents on a SaaS contract in the cloud computing market. So we were interested in intelligent agent technology and its use in the automatic negotiation field.
Our proposed model is composed of two types of agent:

- The customer agent which represents the cloud service customer(CSC), and that follows the CNP negotiation protocol and both constraint satisfaction problem and multi attribute utility theory in order to discover the cloud service providers that offer the desired service, next, select among them(if possible) the ones to start negotiation with, then, start the negotiation and finally choose one provider to sign the SaaS contract with.
- The provider agent which represents each one of the cloud service provider(CSPs)and uses the same elements as the customer agent in order to response to the call for proposal (cfp) sent by the customer agent , and if been chosen, to start negotiation with .

In the future, we aim to detail the functionalities of both types of agents that make up our architecture. Also, we aim to evaluate our proposed method more extensively through some real system case studies that represent possible situations which may arise during the realization of a possible SaaS contract between the cloud service providers and the service customer.Furthermore, we plan to include a re-negotiation phase which allows both parties to update the agreed terms in case of the contract violation .
This work has been accepted for oral presentation at the International Conference on Advanced Intelligent Systems for Sustainable Development (AI2SD'2019), that will be held at the Faculty of Sciences Semlalia, Cadi Ayyad University Marrakech , Morocco, July 08–11, 2019.(See Appendix)

APPENDIX

Call For Papers

Acceptance Letter

The International Conference on Advanced Intelligent Systems for Sustainable Development
- AI2SD'2019 -

July 08-11 2019, Marrakech, Morocco

Notification of Acceptance

Dear Authors,

Based on the reviewer's reports, we are pleased to inform you that your paper

ID : 289

TITLE : towards a saas contracts negotiation model based on multi agent system

AUTHORS : khellaf douaae, kenatef hana and hioual ouassila

Has been accepted for **ORAL PRESENTATION** at the International Conference on Advanced Intelligent Systems for Sustainable Development (**AI2SD'2019**), that will be held at the Faculty of Sciences Semlalia, Cadi Ayyad University Marrakech , Morocco, ***July 08–11, 2019***.

On behalf of the **AI2SD'2019** Organizing Committee, you are kindly invited to participate in this conference, which I am sure, will certainly enhance the conference's importance and success.

Thank you very much for your contribution, and see you in Marrakech.

Prof. Mostafa Ezziyyani
General Chair
AI2SD'2019

Bibliography

[1] Adina Magda Florea. Multi-agent systems lecture 12, 2002.

[2] F. Zavoral, J. Jung J., and C. Badica. *Intelligent Distributed Computing VII: Proceedings of the 7th International Symposium on Intelligent Distributed Computing - IDC 2013, Prague*. Czech.., August 2013.

[3] Stuart J. Russell. Rationality and intelligence. *Artificial intelligence*, 94(1-2):57–77, 1997.

[4] Yoav Shoham. Agent-oriented programming. *Artificial intelligence*, 60(1):51–92, 1993.

[5] Don Gilbert, Manny Aparicio, Betty Atkinson, Steve Brady, Joe Ciccarino, Benjamin Grosof, Pat OConnor, Damian Osisek, Steve Pritko, Rick Spagna, et al. Ibm intelligent agent strategy. *IBM Corporation*, 1995.

[6] Ferber Jacques. Les systèmes multi-agents, vers une intelligence collective. *InterEditions, Paris*, 322, 1995.

[7] Yoann Kubera, Philippe Mathieu, and Sébastien Picault. *Everything can be Agent!* 2010.

[8] Bruno N. Di Stefano and Anna T. Lawniczak. Cognitive agents: Functionality & performance requirements and a proposed software architecture. In *2009 IEEE Toronto International Conference Science and Technology for Humanity (TIC-STH)*, pages 509–514. IEEE, 2009.

[9] B. Chaib-Draa and P. Levesque. Hierarchical models and communication in multi-agent environments. In *Proceedings of the Sixth European Workshop on Modelling Autonomous Agents and Multi-Agent Worlds (MAAMAW-94)*, pages 119–134, 1994.

[10] B. Chaib-Draa, A. Ken, J. Williams, C. Hall, and R. Kent. Distributed artificial intelligence: An overview. *Encyclopedia Of Computer Science And Technology*, 31:215–243, 1994.

[11] Brahim Chaib-draa. Interaction between agents in routine, familiar and unfamiliar situation. *International journal of cooperative information systems*, 5(01):1–25, 1996.

[12] Brahim Chaib-draa and Pascal Levesque. Hierarchical model and communication by signs, signals and symbols in multiagent environments. *Journal of Experimental and Theoretical Artificial Intelligence*, 8(1):7–20, 1996.

[13] Innes A. Ferguson. Touringmachines: An architecture for dynamic, rational, mobile agents. Technical report, University of Cambridge, Computer Laboratory, 1992.

[14] Micha.el P. Georgeff and Amy L. Lansky. Reactive reasoning and planning. In *AAAI'87 Proceedings of the sixth National conference on Artificial intelligence - Volume 2*, pages 677–682, 1987.

[15] Katia P. Sycara. Multiagent systems. *AI magazine*, 19(2):79–92, 1998.

[16] Stephen D. J. McArthur, Euan M. Davidson, Victoria M. Catterson, Aris L. Dimeas, Nikos D. Hatziargyriou, Ferdinanda Ponci, and Toshihisa Funabashi. Multi-agent systems for power engineering applications—part i: Concepts, approaches, and technical challenges. *IEEE Transactions on Power systems*, 22(4):1743–1752, 2007.

[17] S. D. J. Mcarthur, E. M. Davidson, V. M. Catterson, A. L. Dimeas, N. D. Hatziargyriou, F. Ponci, and T. Funabashi. Multi-agent systems for power engineering applications part ii: Technologies, standards, and tools for building multi-agent systems. *IEEE Transactions on Power systems*, 22(4):1753–1759, 2007.

[18] Victoria M. Catterson, Euan M. Davidson, and Stephen D. J. McArthur. Practical applications of multi-agent systems in electric power systems. *European Transactions on Electrical Power*, 22(2):235–252, 2012.

[19] Juhwan Jung, Chen Ching Liu, Steven L. Tanimoto, and Vijay Vittal. Adaptation in load shedding under vulnerable operating conditions. *IEEE Transactions on Power Systems*, 17(4):1199–1205, 2002.

[20] Eric Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the national academy of sciences*, 99(suppl 3):7280–7287, 2002.

[21] Klaus G. Troitzsch and Nigel Gilbert. Simulation for the social scientist, 2005.

[22] Nan-Peng Yu, Chen-Ching Liu, and James Price. Evaluation of market rules using a multi-agent system method. *IEEE Transactions on Power Systems*, 25(1):470–479, 2010.

[23] Floyd Garvey and Suresh Sankaranarayanan. Intelligent Agent based Flight Search and Booking System. *International Journal of Advanced Research in Artificial Intelligence*, 1(4), 2012.

[24] Vg Koutkias, I. Chouvarda, and N. Maglaveras. Multi-agent system architecture for heart failure management in a home care environment. In *Computers in Cardiology, 2003*, pages 383–386, 2003.

[25] Frederico L. G. Freitas and Guilherme Bittencourt. *Cognitive Multi-agent Systems for Integrated Information Retrieval and Extraction over the Web*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

[26] Franco Zambonelli, Nicholas R. Jennings, and Michael Wooldridge. Organisational abstractions for the analysis and design of multi-agent systems. In *International Workshop on Agent-Oriented Software Engineering*, pages 235–251. Springer, 2000.

[27] M. D. Sadek. Attitudes mentales et interaction rationnelle: vers une théorie formelle de la communication. *Unpublished thèse, Université Rennes*, 1, 1991.

[28] Brahim Chaib-Draa, Imed Jarras, and Bernard Moulin. Systèmes multi-agents: principes généraux et applications. *Edition Hermès*, 242:1030–1044, 2001.

[29] Reid G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on computers*, (12):1104–1113, 1980.

[30] Thomas W. Malone et al. What is coordination theory? 1988.

[31] Alexandre Grouls. Agents et systèmes multi-agents: vers une synthèse de ces concepts. 2013.

[32] J. L. KONING and S. Pesty. Modèles de communication, in principes et architectures des systèmes multi-agents. *édité par Jean-Pierre Briot et Yves Demazeau, Collection IC2, Hermes Science Publications, Paris*, 2001.

[33] Stefan Bussmann and Jorg Muller. A negotiation framework for co-operating agents. *Proceedings of CKBS-SIG*, pages 1–17, 1992.

[34] Edmund H. Durfee and Victor R. Lesser. Negotiating task decomposition and allocation using partial global planning. *Distributed Artificial Intelligence (Vol. 2)*, pages 229–243, 1989.

[35] Michael Wooldridge, Nicholas R. Jennings, and Katia P. Sycara. A Roadmap of Agent Research and Development. *Autonomous Agents and Multi-Agent Systems*, 1(1):7–38, 1998.

[36] Isabelle Bedou. Modélisation de la coopération entre Systèmes d'Information : une approche cognitive basée sur la négociation. *INFORSID. Congrès*, pages 427–442, 1997.

[37] Iyad Rahwan, Liz Sonenberg, Nicholas R. Jennings, and Peter McBurney. Stratum: A methodology for designing heuristic agent negotiation strategies. *Applied Artificial Intelligence*, 21(10):41, 2007.

[38] Martin Beer, Mark D'inverno, Michael Luck, Nick Jennings, Chris Preist, and Michael Schroeder. Negotiation in multi-agent systems. *The Knowledge Engineering Review*, 14(3):285–289, 1999.

[39] Mihnea Bratu. *Coordination de négociations pour la sous-traitance dans le cadre d'alliance inter-organisationnelles*. PhD thesis, Ecole Nationale Supérieure des Mines de Saint-Etienne, 2007.

[40] H. Jürgen Müller. Negotiation principles, foundations of distributed artificial intelligence. *GMP O'Hare, and NR Jennings, New York: John Wiley & Sons*, 1996.

[41] Marjorie Le Bars. *Un simulateur multi-agent pour l'aide à la décision d'un collectif: application à la gestion d'une ressource limitée agro-environnementale*. PhD thesis, Université Paris Dauphine-Paris IX, 2003.

[42] Chun-Che Huang, Wen-Yau Liang, Yu-Hsin Lai, and Yin-Chen Lin. The agent-based negotiation process for b2c e-commerce. *Expert Systems with Applications*, 37(1):348–359, 2010.

[43] Chun Ching Lee and C. Ou-Yang. A neural networks approach for forecasting the supplier's bid prices in supplier selection negotiation process. *Expert Systems with Applications*, 36(2):2961–2970, 2009.

[44] Kannan Mohan, Peng Xu, and Balasubramaniam Ramesh. Supporting dynamic group decision and negotiation processes: a traceability augmented peer-to-peer network approach. *Information & management*, 43(5):650–662, 2006.

[45] N. R. Jennings, S. Parsons, C. Sierra, and et Faratin P. Automated negotiation. *In Proceedings of 5th International Conference on the Practical Application of Intelligent Agents and Multi-Agent Systems (PAAM-2000), Manchester, UK*, pages 23–30, 2000.

[46] Singiresu S. Rao. Game theory approach for multiobjective structural optimization. *Computers & Structures*, 25(1):119–127, 1987.

[47] Shiva S. Makki, Luther Tweeten, and James Gleckler. Agricultural trade negotiations as a strategic game. *Agricultural Economics*, 10(1):71–80, 1994.

[48] Maryam Esmaeili, Mir-Bahador Aryanezhad, and Panlop Zeephongsekul. A game theory approach in seller–buyer supply chain. *European Journal of Operational Research*, 195(2):442–448, 2009.

[49] Jeffrey E. Teich, Hannele Wallenius, Markku Kuula, and Stanley Zionts. A decision support approach for negotiation with an application to agricultural income policy negotiations. *European Journal of Operational Research*, 81(1):76–87, 1995.

[50] Peyman Faratin, Carles Sierra, and Nick R. Jennings. Negotiation decision functions for autonomous agents. *Robotics and Autonomous Systems*, 24(3-4):159–182, 1998.

[51] Martin Bichler and Arie Segev. Methodologies for the Design of Negotiation Protocols on E-markets. *Computer Networks*, 37(2):137–152, 2001.

[52] Daniel R. Crary. International negotiation: Analysis, approaches, issues: Victor A. Kremenyuk, Ed. San Francisco: Jossey-Bass Publishers, 1991, 486 pp, 1992.

[53] Abdelhakim Hafid, Gregor von Bochmann, and Rachida Dssouli. A quality of service negotiation approach with future reservations (NAFUR): a detailed study. *Computer Networks and ISDN Systems*, 30(8):777–794, 1998.

[54] Ulrich Pordesch. Negotiating security among end users: concept and test in a simulation study for the public health service. *Computer networks and ISDN systems*, 30(16-18):1597–1605, 1998.

[55] Harris Sondak and Max H. Bazerman. Matching and negotiation processes in quasi-markets. *Organizational Behavior and Human Decision Processes*, 44(2):261–280, 1989.

[56] Randall Davis and Reid G. Smith. Negotiation as a metaphor for distributed problem solving. *Artificial intelligence*, 20(1):63–109, 1983.

[57] Sati Singh Sian. Adaptation based on cooperative learning in multi-agent systems. *Decentralized AI*, 2:257–272, 1991.

[58] Man Kit Chang and Carson C. Woo. SANP: A Communication Level Protocol for Negotiations. *ACM SIGOIS Bulletin*, 13(3):8, 1992.

[59] Man Kit Chang and Carson C. Woo. A speech-act-based negotiation protocol: design, implementation, and test use. *ACM Transactions on Information Systems*, 12(4):360–382, 1994.

[60] Armando Fox, Rean Griffith, Anthony Joseph, Randy Katz, Andrew Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, and Ion Stoica. Above the clouds: A berkeley view of cloud computing. *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS*, 28(13):2009, 2009.

[61] Qi Zhang, Lu Cheng, and Raouf Boutaba. Cloud computing: state-of-the-art and research challenges. *Journal of internet services and applications*, 1(1):7–18, 2010.

[62] Marios D. Dikaiakos, Dimitrios Katsaros, Pankaj Mehra, George Pallis, and Athena Vakali. Cloud Computing: Distributed Internet Computing for IT and Scientific Research. *IEEE Internet Computing*, 13(5):10–13, 2009.

[63] Frank Gens. Worldwide and regional public it cloud services forecast, 2018–2021, 2019. https://www.idc.com/getdoc.jsp?containerId=US43625818.Accessed 15 May2019.

[64] Cisco. Cisco global cloud index: Forecast and methodology, 2016–2021 white paper, 2016. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.html . Accessed: 15 May 2019.

[65] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud Computing And Emerging IT Platforms: Vision, Hype, And Reality For Delivering Computing As The 5th Utility. *Future Generation Computer Systems*, 25(6):599–616, 2009.

[66] Ian Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu. Cloud computing and grid computing 360-degree compared. *arXiv preprint arXiv:0901.0131*, 2008.

[67] James Staten, Simon Yates, and Lauren E. Nelson. Final version of nist cloud computing definition published, 2011. https://www.nist.gov/news-events/news/2011/10/final-version-nist-cloud-computing-definition-published . Accessed: 14 May 2019.

[68] James Staten. Best practices: Infrastructure-as-a-service (iaas), 2009. https://www.forrester.com/report/Best+Practices+InfrastructureAsAService+IaaS/-/E-RES48378 . Accessed: 18May 2019.

[69] Daniele Catteddu. Cloud computing: Benefits, risks and recommendations for information security. *Iberic Web Application Security Conference*, pages 17–17, 2009.

[70] Jeremy Geelan et al. Twenty-one experts define cloud computing. *Cloud Computing Journal*, 4:1–5, 2009.

[71] Peter M. Mell and Timothy Grance. The nist definition of cloud computing. *Special Publication (NIST SP) - 800-145*, 23(6):50–50, 2011.

[72] Luis M. Vaquero, Luis Rodero-Merino, Juan Caceres, and Maik Lindner. A break in the clouds: towards a cloud definition. *acm special interest group on data communication*, 39(1):50–55, 2008.

[73] Francesco Maria Aymerich, Gianni Fenu, and Simone Surcis. An approach to a cloud computing network. In *2008 First International Conference on the Applications of Digital Information and Web Technologies (ICADIWT)*, pages 113–118, 2008.

[74] Mark Russinovich. Introduction to windows azure: the cloud operating system. https://channel9.msdn.com/Events/BUILD/BUILD2011/SAC-852F . Accessed 14 May 2019.

[75] Google Inc. Google app engine, 2019. https://aws.amazon.com/fr/vpc/ . Accessed: 17 May 2019.

[76] Microsoft azure cloud services., 2019. http://azure.microsoft.com/fr-fr/services/cloud-services/. Accessed: 17 May 2019.

[77] Inc. Pivotal Software. Pivotal cloud foundry, 2019. https://pivotal.io/platform . Accessed: 14 May 2019.

[78] Salesforce, 2019. https://www.salesforce.com/fr/ . Accessed: 18 May 2019.

[79] Inc. Amazon Web Services. Amazon virtual private cloud, 2019. https://aws.amazon.com/fr/. . Accessed: 17 May 2019.

[80] Google Inc. G suite: Collaboration and productivity apps for business, 2019. https://gsuite.google.com/ . Accessed: 17 May 2019.

[81] Sun Microsystem. Open source & cloud computing: On-demand. *Innovative IT On a Massive Scale*, 2012.

[82] Faith Shimba. Cloud computing: Strategies for cloud computing adoption. 2010.

[83] Fang Liu, Jin Tong, Jian Mao, Robert Bohn, John Messina, Lee Badger, and Dawn Leaf. Nist cloud computing reference architecture. *NIST special publication*, 500(2011):1–28, 2011.

[84] S. Voona and R. Venkantaratna. Cloud computing for banks. *Infosys Technologies Ltd*, 2009.

[85] Rajkumar Buyya, Chee Shin Yeo, and Srikumar Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In *2008 10th IEEE international conference on high performance computing and communications*, pages 5–13. Ieee, 2008.

[86] Michael Miller. *Cloud computing: Web-based applications that change the way you work and collaborate online*. Que publishing, 2008.

[87] Daniele Catteddu. Cloud computing: benefits, risks and recommendations for information security. In *Iberic Web Application Security Conference*, pages 17–17. Springer, 2009.

[88] Traian Andrei and Raj Jain. Cloud computing challenges and related security issues. *A Survey Paper. DOI= http://www. cse. wustl. edu/˜ jain/cse571-09/ftp/cloud. pdf*, 2009.

[89] Yefim V. Natis. Introducing saas-enabled application platforms: Features, roles and futures. *Gartner Inc*, 2007.

[90] K. Selçuk Candan, Wen-Syan Li, Thomas Phan, and Minqi Zhou. At the frontiers of information and software as services. In *New Frontiers in Information and Software as Services*, pages 283–300. Springer, 2011.

[91] Frederick Chong and Gianpaolo Carraro. Architecture strategies for catching the long tail. *MSDN Library, Microsoft Corporation*, pages 9–10, 2006.

[92] Keith Jeffery and B. Neidecker-Lutz. The Future of Cloud Computing: Opportunities for European Cloud Computing Beyond 2010. *Analysis*, page 66, 2009.

[93] Kelly Fiveash. Google docs suffers serious security lapse, 2009. https://www.theregister.co.uk/2009/03/09/google_docs_serious_security_breach/ . Accessed: 18 May 2019.

[94] Politechnica university of bucharest. agents intelligents : Cours web interactif, 2002(accessed February 10, 2019). http://turing.cs.pub.ro/auf2/html/chapters/ . Accessed: 20 April 2019.

[95] Djamel Benmerzoug, Zizette Boufaida, and Mahmoud Boufaida. From the analysis of cooperation within organizational environments to the design of cooperative information systems: an agent-based approach. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 495–506. Springer, 2004.

[96] Linlin Wu, Saurabh Kumar Garg, Rajkumar Buyya, Chao Chen, and Steve Versteeg. Automated sla negotiation framework for cloud computing. In *2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, pages 235–244. IEEE, 2013.

[97] Kwang Mong Sim. Towards complex negotiation for cloud economy. In *International Conference on Grid and Pervasive Computing*, pages 395–406. Springer, 2010.

[98] Seokho Son and Sung Chan Jun. Negotiation-based flexible sla establishment with sla-driven resource allocation in cloud computing. In *2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, pages 168–171. IEEE, 2013.

[99] Gheorghe Cosmin Silaghi, Liviu Dan ŞErban, and Cristian Marius Litan. A time-constrained sla negotiation strategy in competitive computational grids. *Future Generation Computer Systems*, 28(8):1303–1315, 2012.

[100] Kwang Mong Sim. Grid resource negotiation: survey and new directions. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(3):245–257, 2010.

[101] Mario Macías, J. Oriol Fitó, and Jordi Guitart. Rule-based sla management for revenue maximisation in cloud computing markets. In *2010 International Conference on Network and Service Management*, pages 354–357. IEEE, 2010.

[102] Mario Macias and Jordi Guitart. Using resource-level information into nonadditive negotiation models for cloud market environments. In *2010 IEEE Network Operations and Management Symposium-NOMS 2010*, pages 325–332. IEEE, 2010.

[103] Elarbi Badidi. A broker-based framework for integrated sla-aware saas provisioning. *arXiv preprint arXiv:1605.02432*, 2016.

[104] Kwang Mong Sim and B. Shi. Concurrent negotiation and coordination for controlling grid resource co-allocation. *IEEE Trans. Systems, Man and Cybernetics*, (Part B):753–766, 2010.

[105] Peyman Faratin. *Automated service negotiation between autonomous computational agents*. PhD thesis, Queen Mary University of London, 2000.

[106] P. Faratin, C. Sierra, N. Jennings, and P. Buckle. Designing Flexible Automated Negotiators: Concessions trade-offs and issue changes. Technical report, 1999.

[107] C. Sierra. Faratin. p., and nr jennings. a service-oriented negociation model between autonomous agents. In *8th European Workshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW-97) International Conference on Computational Linguistic (COLIG)*, pages 15–35, 1997.

[108] Matthew Butterick. Constraint-satisfaction problems (and how to solve them), accessed February 12, 2019. `https://docs.racket-lang.org/csp/index.html` . Accessed: 8 May 2019.

[109] Shivraj Sarjerao Patil, D. N. Mudgal, and S. B. Patil. Multi-attribute utility theory for contractor prequalification. *SSRG International Journal of Civil Engineering (SSRG-IJCE)*, 2016.

[110] Ryszard Kowalczyk and Van Bui. On constraint-based reasoning in e-negotiation agents. In *International Workshop on Agent-Mediated Electronic Commerce*, pages 31–46. Springer, 2000.

[111] Christophe Levointurier. *De la nécessité d'une vision holistique du code pour l'analyse statique et la correction automatique des Applications Web.* PhD thesis, Université Rennes 1, 2011.

[112] L. Vercouter G.Picard. *"Initiation à la programmation orientée-objet avec le langage Java" Pôle Informatique 2013-2014.*

[113] Sylvain Lefebvre. *Services de répartition de charge pour le Cloud: application au traitement de données multimédia.* PhD thesis, Paris, CNAM, 2013.

[114] Fabio Luigi Bellifemine, Giovanni Caire, and Dominic Greenwood. *Developing multi-agent systems with JADE*, volume 7. John Wiley & Sons, 2007.

[115] Michael Wooldridge and Nicholas R. Jennings. Intelligent agents: Theory and practice. *The knowledge engineering review*, 10(2):115–152, 1995.

[116] Mohammed Amine Guermoudi. *Conception et implémentation d'une systéme multi-agent pour le test de primalité nombre premier.* PhD thesis.