AGENT MEETING SCHEDULER

By

NURAINI ZAINAL ABIDIN

12419

Dissertation submitted in partial fulfillment of
the requirements for the
Bachelor of Technology (Hons)
Information Communication Technology

SEPTEMBER 2011

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Agent Meeting Scheduler

By

NURAINI BINTI ZAINAL ABIDIN

A project dissertation submitted to the
Information Communication Technology Programme
Universiti Teknologi PETRONAS
in partial fulfillment of the requirement for the
Bachelor of Technology (Hons)
Information Communication Technology

Approved by,

[illegible]

(Mohd. Hilmi bin Hasan)

Universiti Teknologi PETRONAS

TRONOH, PERAK

September 2011

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

[illegible]

NURAINI BINTI ZAINAL ABIDIN

Agent Meeting Scheduler

Nuraini Binti Zainal Abidin

Universiti Teknologi PETRONAS

This dissertation is purposed to record all the data gathered throughout author's study and research for this project. A deep study of agent algorithm is conducted based on current available agent meeting scheduler from combination of software agent and algorithm data structure knowledge. The current problem of typical meeting scheduler is it is time consuming and inefficient; and also a resource needs to be allocated to perform the meeting scheduling job. Agent meeting scheduler will be used to replace this typical meeting scheduler to make it more efficient in term of deciding meeting time. The study is meant to research and select suitable algorithm to be implemented in agent meeting scheduler. An agent meeting scheduler prototype then will be developed to prove that the selected algorithm is working properly. Qualitative research method is being used to gather necessary data on agent algorithm and this data will be used to select the suitable algorithm. Through the research conducted on available algorithm for agent meeting scheduler, genetic algorithm is selected to be used in this project. The agent meeting scheduler prototype then will be developed by using PHP language. PHP is selected for its interactivity and extensibility.

Corresponding Author:

Nuraini Binti Zainal Abidin
Bachelor of Technology (Hons) Information & Communication Technology,
Universiti Teknologi PETRONAS,
Email: nuraini.zainal@gmail.com
Mobile Phone: 019-390 9629

ACKNOWLEDGEMENT

First and foremost, praise be upon Allah s.w.t for his mercy giving me the strength in completing this project within the time period.

The writing of this dissertation has been one of the most significant academic challenges that I have been through in completion of my degree program.

Therefore I would like to express my gratitude to my supervisor, Mr. Mohd Hilmi bin Hasan for his endless support and guidance throughout this project despite having many other academic and professional commitment. Further thanks to all my friend and colleague for giving encouragement in completing the project. The encouragement gives me great strength in facing the obstacles in completing this dissertation.

This appreciation also goes to all lecturers for the ideas, assistance and support throughout the completion of this project. I would also like to thank my beloved family for their support.

Finally, thank you to all individuals that have contributed their ideas, knowledge, support and assistance in this project.

TABLE OF CONTENTS

Chapter 1 Introduction

1.1 Background of study	**1 – 2**
1.2 Problem Statement	**2**
1.3 Project Significant	**3**
1.4 Objectives	**3**
1.5 Scope of Study	**4**
1.6 Relevancy of Project	**4**
1.7 Feasibility analysis	
1.7.1 Technical Feasibility	**5**
1.7.2 Scope Feasibility	**5**
1.7.3 Time Feasibility	**6**

Chapter 2 Literature Review

2.1 Agent Meeting Scheduler	**7 – 8**
2.2 How agent meeting scheduler works	**8**
2.3 Variation of algorithm used	**8**
2.3.1 Intelligent Fuzzy Agent	**9**
2.3.2 A*-Algorithm Agent	**10 – 11**
2.3.3 N*-Algorithm Agent	**12 – 13**
2.3.4 Fuzzy Type-2 Ontology Agent	**14 – 15**

Chapter 3 Research Methodology

3.1 Research Methodology	**16 – 17**
3.2 Software Engineering Methodology	**18 – 23**
3.3 Project Activities, Key Milestone & Gantt chart	**24**
3.4 Tools Required	**25**

TABLE OF CONTENTS

Chapter 4 Discussion

4.1 Findings

4.1.1 Current available agent meeting scheduler algorithm 26

4.1.2 Genetic Algorithm 26 – 29

4.1.3 Integration of Genetic Algorithm into the system 30 – 31

4.2 Data Analysis

4.2.1 System Flow 32 – 33

4.2.2 System Architecture 33 – 34

4.2.3 Database Design 35 – 36

4.3 Prototype 37 – 39

Chapter 5 Conclusion and Recommendation

5.1 Conclusion 40

5.2 Recommendation 40

References &Appendices 41 - 46

LIST OF FIGURES

Figure 1	Public Record	17
Figure 2	Waterfall Model	19
Figure 3	Genetic Algorithm Flow	27
Figure 4	Flowchart of genetic algorithm	29
Figure 5	Flow of agent meeting scheduler	32
Figure 6	System Architecture	33
Figure 7	Entity Relationship Diagram	36
Figure 8	Login page	37
Figure 9	Dashboard page	38
Figure 10	Schedule Meeting page	38
Figure 9	Meeting Preferences page	39

LIST OF TABLES

Table 1	Main Task	24
Figure 2	Sub Task	24

Chapter 1

Introduction

1.1 Background of Study

Meeting scheduling is a natural task in any organization. This task should be carried out carefully as meeting plays an important role in coordinating and updating work in an organization. Meeting can take place between two persons or more, and it could be in various types, depending on situation and purpose of the meeting. Meeting also may be conducted in meeting room or remotely by using video or call conference. Attendee of meeting from different country which have different time zone also make meeting scheduling more complex to be done.

Sometimes, people just need to schedule only one meeting; however, most of the time people need to schedule many meetings at the same time under several constraints [1]. While scheduling meeting, the person need to consider availability and preferences of other meeting participants. However, this is difficult as many people try to keep their calendar and preferences private.

There are several commercial meeting schedulers available in order to carry the task, such as Microsoft Outlook and Google Calendar but none of this product capable of autonomously communicating and negotiating with other using in order to schedule meeting in a distributed way by taking consideration of user's preferences and calendar availability (Zunino & Campo, 2009). This is where software agent comes in handy. An agent is a computer system that is capable of independent action based on behalf of its owner [2]. Agent makes your work easier as they do not need to be told what they are supposed to do all the time, but instead the agent will figure out for itself according to their pre-defined goal.

By using agent in meeting scheduler, meeting can be coordinated more easily without much user effort. Agent has the ability to interact and negotiate with another agent. With this social ability, agent is useful because the agent can get user information on user free time and their preferences without exposing it to other user. From the user free time and preferences, decision is made on most suitable time for a meeting. It may require several round of information exchange to locate the suitable meeting time but user involvement in this process is not necessary. This definitely will reduce time taken in deciding most appropriate meeting time in the organization.

1.2 Problem Statement

In a typical organization, employee spends much of their time in meetings. Usually, the process of getting mutual available time between all members of the meeting is time consuming. This process becomes more difficult as the number of meeting attendee increases. This is bad as it may affect the company's performance.

Currently most company use centralized, distributed calendar management to set up meeting for the employee. The method is inefficient because a person need to be in charge to ensure the availability of meeting participants. By doing this, company lost valuable resource's time. As for meeting participants, they need to reply for their availability to attend the meeting. If there are many participants which are unavailable for the meeting, new meeting date and time will be proposed and the steps will be repeated. This procedure is tedious as there are a lot of meeting needs to be scheduled in an organization and this routine is always going on.

There is a need to set up the meeting efficiently without needing to go through this monotonous procedure. This can be solved by using an agent. In this project, agent is used to decide the meeting time based on free time of meeting participant free time without overlapping with other meeting.

1.3 Project Significant

This project main contribution is to prove that agent meeting scheduler can be implemented by using genetic algorithm. Moreover, meeting scheduling process can be eased by using agent based meeting scheduler. This will solve usual problem in meeting scheduling where the process takes longer time and may need to be repeated if the meeting time is not suitable.

Benefits of project:

- Reduce time taken for end user to coordinate meeting time.
- Save company resources to arrange suitable meeting time.
- Make meeting scheduling process more efficient.
- Proves that agent meeting scheduler can be implemented by using genetic algorithm, which is currently unavailable.

1.4 Objectives

The objectives of this project are:

- To research on possible agent algorithm that can be used to develop the project
- To select a suitable algorithm to be implemented with the project
- To design an agent meeting scheduler which have these features:

  a) Set up a meeting based on participant's free time

  b) Inform meeting participant of upcoming meeting date and time

  c) Inform meeting participant of recently scheduled meeting date and time

- To prove that the system can be implemented by using the selected algorithm

1.5 Scopes of Study

Prototype of distributed meeting scheduler will be developed which:

- Select suitable meeting date and time given date, duration and participants
- Schedule meeting up to 5 meeting participants
- User will update their availability by using the system
- User will not cancel the mutual agreed meeting time decided by agent
- User meeting availability is decided based on user free time
- The system is implemented by using genetic algorithm
- No comparison with other working algorithm will be made

1.6 Relevancy of the Project

This project is relevant to be conducted by the author because she is a student of Information Communication Technology (ICT) programme and this project is an IT based project. Since the project Agent Meeting Scheduler is related to the author's course of study hence this project is relevant to be conducted.

The relevancy points of this project are:

- This project require author to apply knowledge from software agent and algorithm data structure course which the author studied previously.
- Author has ample time to complete the project, which is 1 semester to conduct research and another 1 semester to develop the prototype.
- The author can use UTP IRC e-resources to obtain research materials.
- The prototype can be developed by using the author's laptop.
- No extra or special hardware is required since the project is software based.
- The project can be carried out flexibly everywhere and anywhere, no specific time and place is needed.

1.7 Feasibility Analysis

1.7.1 Technical Feasibility

- Familiarity with application language (PHP): More familiar
- Familiarity with technology (Agent-based Meeting Scheduler): Less familiar

1.7.2 Scope Feasibility

- This project development study focuses on the aspects of algorithm that can be used to implement agent meeting scheduler.
- The study will be conducted and covered within the topics of algorithm that had been applied in developing agent meeting scheduler and suitable algorithm that can be used to develop agent meeting scheduler.
- Based on the selected algorithm, an agent meeting scheduler will be developed to schedule meeting up to 5 users.
- Since the objective is to prove that agent meeting scheduler can be implemented by using genetic algorithm, there are no comparison with other working algorithm required.
- Since the scope state that user will accept the decided meeting time without cancelling it, the author will only focus on deciding suitable meeting time without taking considering the consequences after the decision is made.

1.7.3 Time Feasibility

- The project is divided into 2 phases – 'Final Year Project 1' and 'Final Project 2'. The planning, analysis and design phases will be done in 'Final Year Project 1', and the implementation and testing phases will be carried out in 'Final Year Project 2'. The research and studies part will be done during 'Final Year Project 1' while the prototype must be delivered by end of 'Final Year Project 2'.
- There is ample time to carry out the project because the author conducts research during 'Final Year Project 1' phase, where the duration is one semester (14 weeks). One semester provides enough time for the author to gather important data to be studies for this project. Output of 'Final Year Project 1' is the agent meeting scheduler system design.
- During the 'Final Year Project 2', author will continue the project by developing the agent meeting scheduler based on the system design from 'Final Year Project 1'. The author had been provided 1 whole semester (14 weeks) in order to carry out this task. Since the system architecture and design was done during previous stage, the development process will be easier and can be done within the time limit.
- Since this task does not require any purchase of hardware or tools, so there are no blocking time for author to complete the project. Hence, this project is feasible to be conducted during these 2 semesters.

Chapter 2

Literature Review

The literature review is focused and described on current available agent meeting scheduling technology. Another focus lies on algorithm that the developer implements in their agent meeting scheduler. The algorithm used will be explained in details on the component involved and how it works to coordinate meeting time. These algorithms will be referred as guidance for the author to implement her selected algorithm and to design the system architecture of the system. With these main contents as the backbone of the report, it would be the strong points in the author Final Year Project 2 before starting project development which the author will implement selected algorithm for agent meeting scheduler.

2.1 Agent Meeting Scheduler

Meeting scheduler is not a new technology in this century. As time passes, new technology of meeting scheduler is growing to make it more user-friendly and more effective. According to Zunino & Campo (2009), the main goal of designing meeting scheduler is to make it automated as far as possible, while keeping user's agenda and preferences private. This can be done by using agent in the meeting scheduler. Agent is a software component that can act autonomously to achieve its directed goal. In meeting scheduler, agent will act as a representative of its owner to carry out meeting coordination task. Shakshuki & Koo (2006) said that the agent has the abilities to negotiate with other users, provide free time slots, and display the calendar to users [3]. Thus, using agent in distributed meeting scheduler make the conventional meeting scheduling easier.

There are a lot of researches had been conducted on developing this agent based meeting scheduler, such as Jenning & Jackson (1995)'s Agent Based Meeting Scheduling: A Design and Implementation [4], which gives details on implementing agent in meeting scheduler. Lamsweerde, Darimont & Massonet (1995) have highlighted problem and lesson learnt in goal directed elaboration of requirement for meeting scheduler [5] while Garrido & Sycara (1996) had published conference proceeding on topic Multi-Agent Meeting Scheduling [6]. This proves that the community is aware of agent meeting scheduler and they are trying to improve it from times to times.

2.2 How agent meeting scheduler work

When a user wants to schedule a meeting, the user needs to input meeting details to his agent. Important details required for the meeting is meeting title, type, date and list of attendees. Then the agent, which act as host agent of the meeting will start to communicate and negotiate with other agent of the meeting attendee, which is attendee agent to check for their free time. This ensures privacy of meeting attendee as their agent will only provide free time slot and their meeting preferences to the host agent. From the data obtained from attendee agent, host agent will try to coordinate most suitable meeting time according to meeting attendee's available time. Then the host agent will notify all meeting attendee about the decided meeting time. By using this agent approach, user might save their time by only specifying their intended meeting day and let the agent decide the meeting time.

2.3 Variation of algorithm used

Currently, there are several agent based meeting scheduler that has been implemented by using variety algorithm. Algorithm is a procedure used to solve a mathematical or computational problem or to address a data processing issue. In other words, an algorithm is instruction list designed to reach a particular goal. This algorithm will differentiate on how the agent will carry their task, compared to other meeting scheduler.

2.3.1. Intelligent Fuzzy Agent

One of available agent algorithm for meeting scheduler implementation is intelligent fuzzy agent developed by Chang-Shing Lee & Chen-Yu Pan (2003) [7]. The Intelligent Fuzzy Agent (IFA) contains three sub-agents which are Meeting Negotiation Agent (MNA), Fuzzy Inference Agent (FIA) and Genetic Learning Agent (GLA) to assist the meeting host in holding the meeting. These three components are designed to carry out suitable meeting time selection.

When a user request to schedule a meeting, MNA receive the invitee's name, meeting time slot and meeting subject and this details will be sent to MSDSS and FIA. Upon receiving invitee's name, MSDSS retrieve their schedules from Group Calendar Data Base (GCDB) and common free time or removable working time will be computed for all invitee and then respond the computing result to MNA. Next, FIA will deduce the sufficient meeting time with the attending possibility of all invitee based on information provided by MNA and Personalized Knowledge Base (PKB).

Then, parameter of the fuzzy variable which represents the behavior of the invitee will be stored in PKB for FIA. FIA also will receive details of the meeting including invitee's name and their priority, meeting time length, meeting event importance and meeting time preference with working priority. After that FIA will retrieves the PKB to get the meeting time preference of all invitee for computing attending possibility to be sent to MSDSS. Afterwards, meeting host will receives adequate meeting time slots from MSDSS and decides the final meeting time. Result of final meeting time then will be passed back to MSDSS and it will refer to GCDB to get the personal profile of attendee and announce the meeting information.

2.3.2. A*-Algorithm Agent

Next available algorithm of agent meeting scheduler was used by Sugumaran, Narayanasamy & Easwarakumar [8]. Each organization (node) is assigned a set of agents, which one agent per person. There is one agent called scheduling agent (SA) while the other agents are user agent (UA). The SA schedule the meeting on behalf its user and on behalf if there are global meeting conducted, SA will act on behalf of the organization.

There are 2 data structures in A*-Algorithm which are open-heap and node-set. Open heap have nodes that have been generated but not yet expanded. Every node of open heap is mainly used to keep the node's heuristic value and its node-id. The open-heap is a min/max heap in which min/max heuristic value node is at the root. This ensures that open-heap will always get expanded before all other nodes. As for node-set, it contains node that have been generated. It is mainly used to check whether a node is already in the search tree or not. Purpose of having node-set is to reduce the complexity of the algorithm in search of nodes.

There are 3 phase of negotiating protocol implemented in this agent meeting scheduler. First phase is proposal to the executive. In this phase, upon receiving meeting schedule request; the agent will consult its' user calendar with user's preferences to get user availability. If user has any free time slot, the proposal will be sent to the executive otherwise will apply A*-Algorithm to find suitable meeting time request. The proposal consists of a set of date and duration of a particular meeting. Normally the agent uses 3 proposals at a time. When the executive receives the proposal, it will be processed and the executive then will send the bid to the host agent. Based on the bid, host agent will decide to go for next phase, negotiate or submit new proposal.

Second phase is proposal to the invitees. Host agent will send the proposal to the invitees based on the result received from the executive bid. Upon receiving the proposal invitee will decide based on the priority on the meeting, instruction from the host agent and its opinion and they invitee may apply A*-Algorithm for the proposal. Next the invitee will send their bid to host agent. Based on the bid, host agent will decide to go for next phase, negotiate or submit new proposal with the available information obtained from the executive and the attendees.

Last phase of negotiation is confirmation/cancellation to executives/invitees. In this phase, host agent will send confirmation/cancellation message to all executive and invitees. If any of them finds that the time slot is not available for the meeting, they will apply A*-Algorithm to find any user is free in that particular group for the promised slot. If the promised slot is possible to replace the meeting time slot, agent will send message to all meeting participants. Otherwise, user whom are unavailable for the meeting send Cancellation message to host. If host receive at least one Cancel message, it will negotiate with all executive and invitee whom sent the Cancel message or else it will unblock the time slot for the meeting and send new proposal with available details obtained from user so far.

2.3.4. Fuzzy Type-2 Ontology Agent

Recent agent based meeting scheduler was developed by by CS Lee, MH Wang, MH Wu, CY Hsu, YC Lin, SJ Yen on 2010 [10]. This meeting scheduler is implementing fuzzy type 2 ontology algorithm to its agent. There are 3 important components in this meeting scheduler which are type-2 fuzzy set, type-2 fuzzy personal ontology and fuzzy type-2 meeting scheduling ontology. According to the developer, fuzzy type-2 was selected because it provides better performance than fuzzy type-1 in term of understanding its user. Meanwhile, fuzzy type-2 ontology was used to set a better meeting time by considering user personal preferences.

This meeting scheduler is built by several important components. The decision support multi-agent is formed by scheduling agent, meeting negotiation agent, fuzzy interference agent and sentence extraction agent. This is the core component of the system which will handle the important task in deciding meeting time. In this system, every meeting participant will have their own personal agent and type-2 fuzzy personal ontology. Personal agent will represent their owner in communicating and negotiating with other agent in order to decide meeting time. Meanwhile, type-2 fuzzy personal ontology is used to record user meeting preferences and behaviour.

This meeting scheduler executes several steps to coordinate suitable meeting time for its participants. First, the host will ask his personal agent to address a meeting request. The host agent will do so by sending meeting related information such as meeting subject, time and place to the main component of the system, which is decision supported multi-agent. The scheduling agent then will notifies all attendee's personal agents including the host agent to retrieve the preferences for the meeting and schedules for the future activities from their own type-2 fuzzy personal ontology.

Then, the scheduling agent will organized all retrieved meeting-related information to construct the type-2 fuzzy meeting scheduling ontology for all potential attendees. This meeting scheduling ontology was built with help of domain experts too. One of decision support multi-agent component, scheduling agent will retrieves the built type-2 fuzzy meeting scheduling ontology to get the schedule and preferences of the attendees. If there are no conflict with the scheduled meeting for all potential attendees, the meeting schedule agent will report the result to host's personal agent. However, if there are conflicts between attendee free time, negotiations between agents will take place.

The negotiation will be handled by meeting negotiation agent. If the negotiation process is successful, the meeting negotiation agent will send the result back to the host's personal agent. Otherwise, it will pass the meeting-related information to the fuzzy inference agent to infer the attendance possibility for the attendees. The inferred result then will be transferred to sentence extraction agent to transform the result into a semantic sentence. The result in form of this semantic sentence then will be submitted to the host. Finally, host will announce the final meeting-related information such as the confirmed attendees, pre-defined meeting time and places to the final attendees.

Chapter 3

Methodology

3.1 Research Methodology

This final year project aims to create agent based distributed meeting scheduler. Since there are several similar systems currently available, this project will approach to design the project by implementing different algorithm from current system. In order to achieve this, research is extremely important in determining suitable algorithm to be used in designing the system.

Qualitative research method was chosen for this project. This type of research helps to gain insight into value system and how these attributes have an important role in decision making and conducting research. For this project, qualitative research of public record approach was being practiced in order to achieve project goals.

Public records can be collected from external or internal source. They are all part of public domain information and were created with the purpose of helping and furthering research. External records are government office records, census and vital statistics reports, newspaper archives and other suitable public domain that can be accessed easily whereas internal records are specific to particular companies or organization. This includes journal and proceeding paper and also any published material of the particular companies

The materials used for this project research mostly are from journal which can be accessed by using Universiti Teknologi PETRONAS (UTP) e-resources service. Google Scholar also was used in finding suitable material for the topic. It is important to choose quality material for reference in this research to ensure its reliability. The journal then will be evaluated whether it is suitable to be used as reference for the topic or not. The information gained then will be analyzed and then decision is made in fulfilling this project goal.

Research was done to obtain information on multi agent distributed meeting scheduler algorithm. The finding then will be listed as current available algorithm implemented in distributed meeting scheduler. Algorithm that is not being listed, which means it is not yet being implemented in distributed meeting scheduler then will be selected to be implemented in this project.

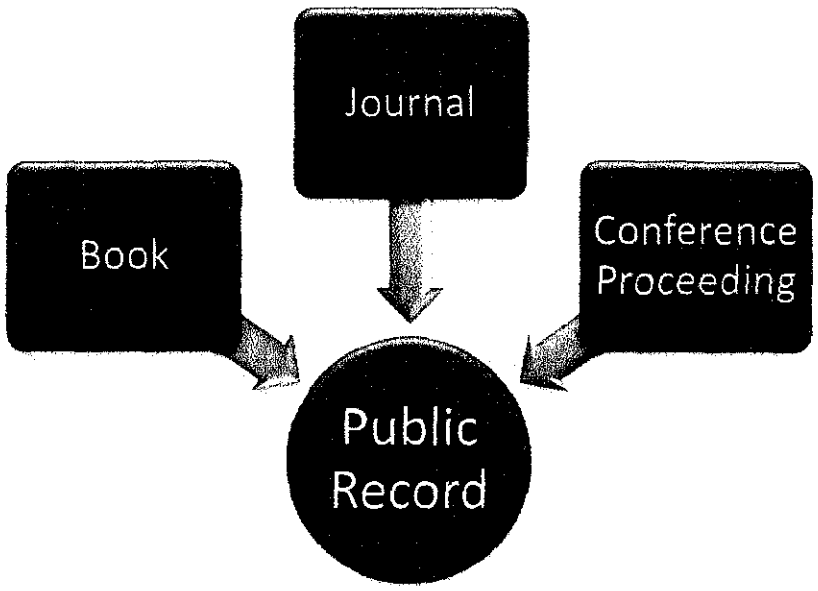


Figure 1: Public Record

3.2 Software Engineering Methodology

System development life cycle (SDLC) is a conceptual model used in project management that describes the phases involved in an information system development project. This covers from the initial phase from capturing user requirement until the very end of process which is system delivery.

It is an important process used by a system analyst to develop an information system including requirement, validation, training and stakeholder ownership. Any SDLC should output a system which high in quality and meets or exceeds customer expectation, complete within time and cost estimates, work efficiently and effectively and also inexpensive to maintain.

Various SDLC model have been created such as waterfall model (the original SDLC method), rapid application development (RAD), joint application development (JAD), the fountain model and the spiral model. Nowadays several models are customized and combined into a hybrid methodology. Regardless model type chosen, documentation is crucial and is usually done in parallel with the development process. The oldest and most commonly used is the waterfall model. Waterfall model is a sequential stage which the output of each stage will become the input for the next.

Choosing correct SDLC is important to keep the progress of project development in phase. SDLC will determine whether the project development can change phase easier when some error or changes occur in the project. This will determine whether the project can run effectively or not. Choosing different SDLC will draw different planning in system development. Almost all SDLC is similar by having the same phases, but the sequence or phase occurrence and ordering is arranged slightly different with each other.

For this project, waterfall model was chosen for its simplicity. Below is the diagram of waterfall model:

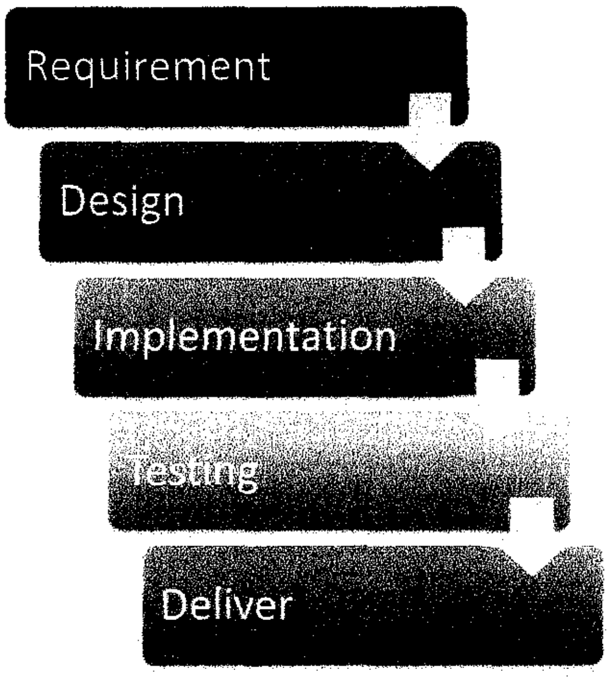


Figure 2: Waterfall Model

The model starts with system requirement stage, followed by design, implementation, testing and finally delivery of the system. The details of every phase will be described below:

1) **Requirement**

The first phase in software development is gathering the requirement. All requirement of the system that need to be developed will be stated clearly during this phase. Requirements are a set of functionalities and constraints that the end user expects from the system. The requirement is gathered and then will be analyzed for the validity and possibility of development. Final output of this phase is Requirement Specification which will provide guideline for the next phase of the model.

During this phase, the author had set the system requirement based on the objective and scope of the project. System requirement is important and should be conductor carefully as this will lead the direction of the project. This requirement also serves as constraint in developing the system.

Proposed system requirement for this agent meeting scheduler project:

- To develop a prototype that will be able to schedule meeting time based on user free time given date of meeting
- The system shall be able to support up to 5 users
- The system also shall be able to inform user on agreed meeting time and date
- Dedicated GUI shall be developed to manage the meeting scheduling activity

2) Design

The second phase of waterfall model software development is design. Design serves as a blueprint for a developer to implement it in the software. Requirement specification from the first phase is studied; and the overall system structure and its outlay will be defined in this phase. The technology, architecture, database design and data structure design of the software will also be defined during this phase. From this information, developer can plan how they want to construct this software and the logic involved within it. Software analysis and design is a very important phase in software development process. Any fault in this phase could cost high price in order to solve it.

Generally, the author will need to develop GUI where the meeting time will be scheduled and implement genetic algorithm in the logic of getting the suitable meeting time.

The system prototype will have these features:

- Implement genetic algorithm in coordinating most preferred meeting time given date, duration and participants
- Inform user of upcoming and recently scheduled meeting date and time
- User may update their preferred meeting time whether morning or evening session

3) Implementation

The next step after design phase is software implementation. Based on algorithm planned or flowchart designed, the actual coding of the software is carried out during in this phase. This is where the concept and idea will be turned into a product. Code generation will encounter no problem if design of the software is done in detailed manner. A proper execution of previous stage will ensures a smooth implementation of this stage.

The project is web based, therefore the author will be using server client model for the system architecture. The system will be developed by using PHP language. The language is suitable because it work great with HTML, offer interactivity and easy to learn. Besides that, PHP is very powerful and there are extensive available resources can be found online for author to learn more about the language. The author decided to use local server for the project, which can be implemented by XAMPP for Windows tool. The data for the system will be stored by using MySQL.

There are 3 main functions that need to be designed for this system which are function to get the meeting time and date by checking meeting participant's availability; and another 2 functions to update meeting preferences which are user availability and preferred meeting time. The author is focusing on meeting scheduling function since it is significant for the author's project and other functions are depending on it. Genetic algorithm can only be implemented if this function is completed.

4) Testing

After completing the development stage, the system prototype needs to be tested. This is to ensure the prototype meet its requirement and to check for any flaws or bug within it. If any bug or flaws found during this phase, the author has to revert back to design mode, which is to code properly for every function required for the system.

Before the system will be tested by the user, developer must test the software in order to find any hidden bug and ensure all the function is working properly. Then, the system will be tested by using usability test and UAT.

Usability test will require participation of moderator, observer and participant. Moderator will give instruction to user on how to use the system and observer will observe user reaction and feedback in using the system. This test are meant to test whether the system achieve its goal effectively. Based on this test, recommendation can be provided to developer on how to make their product more effective and efficient. This test emphasis more on user interface of the system.

Meanwhile, UAT is conducted to gives the end user the confidence that the software being delivered to them meets their requirement. This test also helps to detect bug related to usability of the software. UAT is carried out in environment that closely resembles the real environment which it is intended to be used. In the author case, the UAT will be conducted by using desktop or laptop as it is typical in any organization.

There are several steps to conduct User Acceptance Testing:

- User Acceptance Test (UAT) Planning
- Designing UA Test Cases
- Selecting a Team that would execute the (UAT) Test Cases
- Executing Test Cases
- Documenting the Defects found during UAT
- Resolving the issues/Bug Fixing
- Sign Off

Test case will be prepared in order to test the software. If entire scenario in the test case is passed successfully, then the software is ready to be delivered.

5) Deliver

This is the last phase of waterfall model software development. A proper execution of all the preceding stage ensures a software meet its requirement and satisfies end user. However, at this stage developer needs to provide the user with support regarding the developed software. Software will definitely go through change once it is delivered to the end user. Change could happen due to some unpredicted input value into the system or user have new requirement need to be added to the system. If user demands some further enhancement to be made, the process will be started again from the first phase which is requirement.

In this stage, the author will submit the project to her FYP supervisor and external examiner to be assessed. There is no support or maintenance will be provided upon the product delivered as the author is considered as already completed her FYP project. However, the project may be continued by other FYP student in the upcoming semester.

3.3 Project Activities, Key Milestone and Gantt Chart

Since the project will be carried out in 2 semesters which are 'Final Year Project 1' and 'Final Year Project 2', the project activities are divided into two major classes of tasks which are main task and sub-task. The main tasks are consisted of:

Main Task	Status
Requirement	*Accomplished*
Design	*Accomplished*
Implementation	*Accomplished*
Testing	*Accomplished*
Deliver	*Accomplished*

Table 1: Main Task

The sub-task is consisted of:

Sub-Task	Status
Progress Report	*Accomplished*
Pre-EDX	*Accomplished*
Dissertation draft	*Accomplished*
Viva	*Accomplished*
Technical paper	*Accomplished*
Final dissertation	*Accomplished*

Table 2: Sub Task

Refer appendices for project activities, key milestone and Gantt chart

3.4 Tools required

Software

- Adobe Dreamweaver CS4 (main development tool)

  Adobe Dreamweaver is a proprietary web authoring and editing software that allows users to preview websites natively in a preview pane or in locally installed web browsers. It provides transfer and synchronization features, the ability to find and replace lines of text or code by search terms and regular expressions across the entire site, and a templating feature that allows single-source update of shared code and layout across entire sites without server-side includes or scripting. Dreamweaver support many languages for web technologies such as CSS, JavaScript, and various server-side scripting languages and frameworks including ASP (ASP JavaScript, ASP VBScript, ASP.NET C#, ASP.NET VB), ColdFusion, Scriptlet, and PHP.

- MySQL (database support)

  MySQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases. The SQL phrase stands for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. Therefore the author can use the software for free to develop the system.

  The author is using phpMyAdmin, which is an open source tool written in PHP intended to handle the administration of MySQL with the use of a Web browser. It can perform various tasks such as creating, modifying or deleting databases, tables, fields or rows; executing SQL statements; or managing users and permissions. The phpMyAdmin is obtained in package with XAMPP for Windows.

CHAPTER 4

DISCUSSION

4.1 Findings

4.1.1 Current available agent meeting scheduler algorithm

Meeting scheduler using agent is not a new paradigm in this era. This concept actually had been introduced long time ago and several agent based meeting scheduler had been developed by using various algorithm. The author had conducted qualitative research by using public record to gain data, and below are the result obtained:

Current available agent meeting scheduler algorithm:

1) Fuzzy type 2 ontology algorithm (2010)
2) A*-algorithm (2006)
3) N*-algorithm (2003)
4) Intelligent Fuzzy (2003)

From the above list, the author concludes that there are no current agents meeting scheduler use genetic algorithm. Hence, the author selects genetic algorithm (GA) to be implemented in this project.

4.1.2 Genetic Algorithm

Genetic algorithm mimics the process of natural evolution. It may be used to optimize solution of given problem by applying technique inspired by natural evolution, such as mutation, inheritance, selection and crossover. There are several factors why genetic algorithm should be considered to be implemented in this system. Firstly, genetic algorithm has the ability to adapt to wide range of problem. It also has the ability to optimize problem solved by applying several techniques.

However, it does not guaranteed that result produced will be global optimum solution but the generated solution is acceptably good and quick.

According to Raymond S.T Lee (2006), basic operation of genetic algorithm:

1) Initialization of the population
2) Parent selection process
3) Reproduction process involving crossover and mutation operation
4) Fitness value evaluation
5) Iterative execution on the new population until satisfactory performance

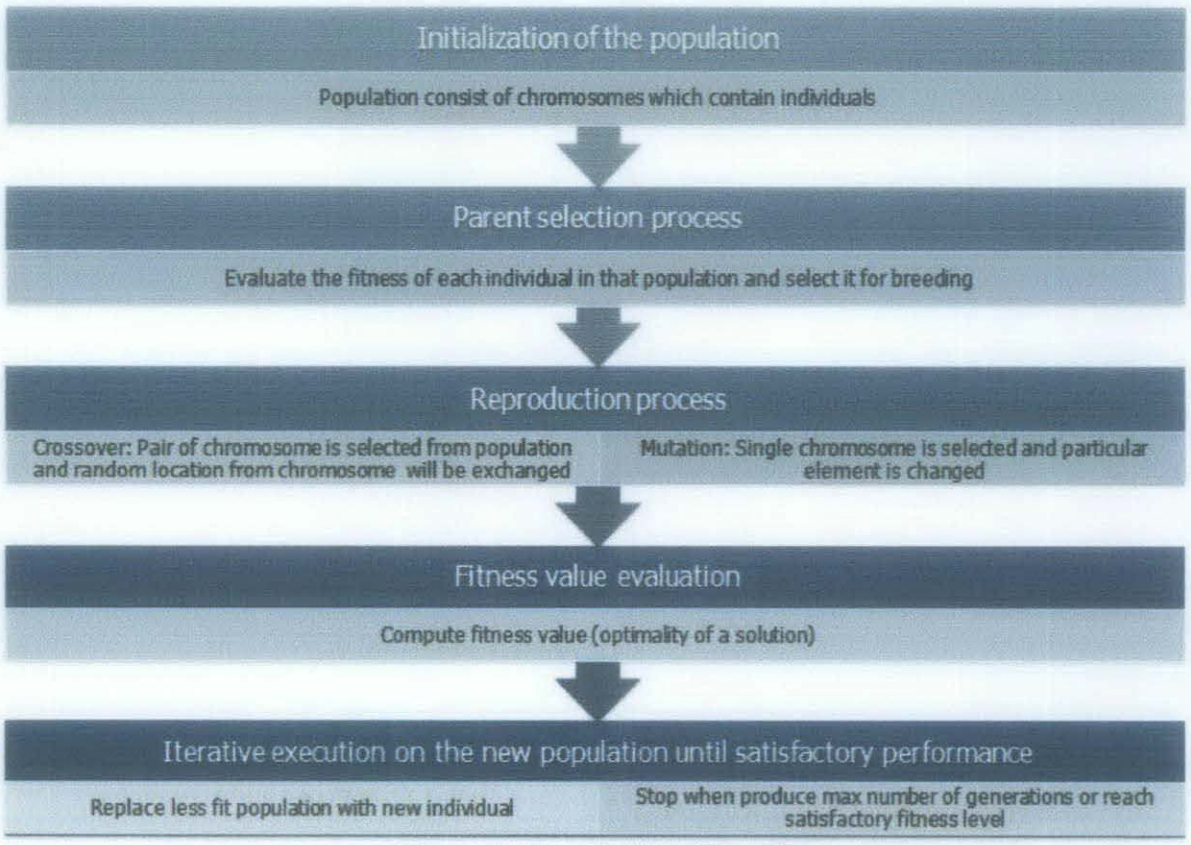

Initialization of the population
Population consist of chromosomes which contain individuals
Parent selection process
Evaluate the fitness of each individual in that population and select it for breeding
Reproduction process
Crossover: Pair of chromosome is selected from population and random location from chromosome will be exchanged
Mutation: Single chromosome is selected and particular element is changed
Fitness value evaluation
Compute fitness value (optimality of a solution)
Iterative execution on the new population until satisfactory performance
Replace less fit population with new individual
Stop when produce max number of generations or reach satisfactory fitness level

Figure 3: Genetic Algorithm Flow

Generally, this is flow of genetic algorithm. First, population consist of chromosomes which contain individual is selected. Population represents solution of problem available while individual represents possible solution that can be used to solve the problem. In this case, the problem is deciding meeting time. Then the phase of parent selection begins. In this phase, fitness level of every individual is tested. Fitness level is optimality of the solution. Higher fitness level indicates the individual is better than the rest. From the fitness test conducted, pair of individual is selected as parent to be used for breeding process in next phase.

The process then continues to third stage, which is reproduction process. There are 2 method of reproduction in genetic algorithm. First reproduction method is crossover. In this method, random location of selected parent will be exchanged. This ensures that there is no update lost between all individual and they can gather every possible change in the population. This method occurs occasionally in genetic algorithm. Meanwhile, second method of reproduction is mutation. One chromosome is selected randomly from the population and particular element of it is changed. This happen randomly and the possibility of occurrence is far lower than the first method. However, second method is important to create diversity of the population.

After reproduction process, fitness level of the individual in population will be tested again. Then the process will continue with reproduction and so on. This process of replacing less fit population with new individual will continue until performance satisfactory is achieved. This performance satisfactory can be achieved by reaching specified maximum number of generation or reaching satisfactory fitness level. However, it is advisable to stop upon reach of satisfactory fitness level, not based on maximum number of generation because the fitness level might not be satisfied and reach its optimum value yet.

Below is flowchart of the genetic algorithm:

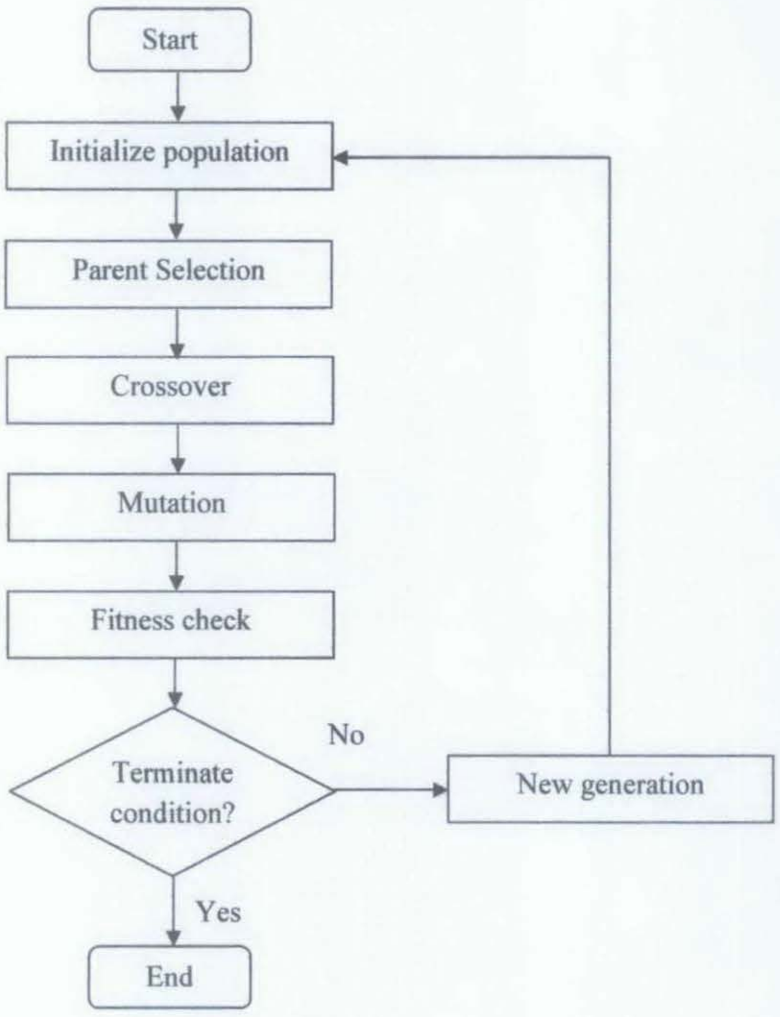


Figure 4: Flowchart of genetic algorithm

4.1.3 Integration of Genetic Algorithm into the system

In order to implement genetic algorithm to the system, the author need to identify problem that need to be solved by using genetic algorithm. Genetic algorithm is way to solve problem where the generated answer is the best possible solution that can solve particular problem. For this agent meeting scheduler, the problem is best sequence of time slot to where all meeting participants are available. The solution will be in time slot form, from 8 am to 5 pm where meeting participant have most high availability during that time.

In normal meeting scheduler, the process of checking meeting availability is done iteratively. As example, meeting scheduler first check all participant's availability on given date at 8 am. If not all participant is available then the system will check availability at 9 am, then 10 am until 4pm. However by using the genetic algorithm, instead of checking the meeting time in sequence from 8 am to 4 pm it will check by using the proposed time slot with high user availability. For instance, the system may check time at 4pm, then 10am according to the solution generated by genetic algorithm.

The component will be explained in details below according to the genetic algorithm sequence used in this agent meeting scheduler. First step is initialization of the population. Population consist of chromosomes which represents solution which is proposed time slot. In this system, chromosome was initialized with random value from 8 to 16 which represents meeting time slot (8 am to 4pm). Since genetic algorithm mimics natural process therefore quantity of generated chromosomes also will be random.

After population initialization, fitness test of each chromosome will be conducted. Fitness value determines optimality of the solution. This can be done by checking 2 important value of the time slot which is availability for every meeting participants and participant's preferred time slot. Each time slot can have 0 to 2 points for every meeting participants involved. If the meeting participant is available during that time, the time slot score will increase by 1. If the time slot match with participant's preferred meeting slot, its score will increment by 1. This fitness test will be done to all time and include involved meeting participants only.

Next step is parent selection process. According to fitness value, 2 best chromosomes will be selected as parent and it will perform reproduction process. This is done by applying roulette wheel selection. In roulette wheel selection process, each individual probability of being selected is directly proportional to the individual's fitness, or score value. This means individual with higher score has higher possibility to be selected as parent. 2 individual will be chosen randomly and assigned as parent to proceed with next step which is reproduction.

Reproduction is done by crossover and mutation. This process is important to vary chromosome from one generation to another. In crossover process, genetic algorithm combine 2 chromosome to produce a new chromosome which may be better than both parent if it takes best characteristics from each of the parent. Crossover point is randomly selected and selected parent's chromosome will exchange to produce new chromosome. There is low percentage of mutation happen where specific time slot will be assigned with random value to add variety to the chromosome.

This process will be done iteratively until it meets its satisfactory performance. In this case, the algorithm will stop if the proposed time slot is same with previous time slot or if the algorithm reaches its max generation. Number of generation for the algorithm is assigned randomly to make it more natural process.

4.2 Data analysis

4.2.1 System Flow

Below is the flow of the system. This will define roughly how the system will work and function provided by the system.

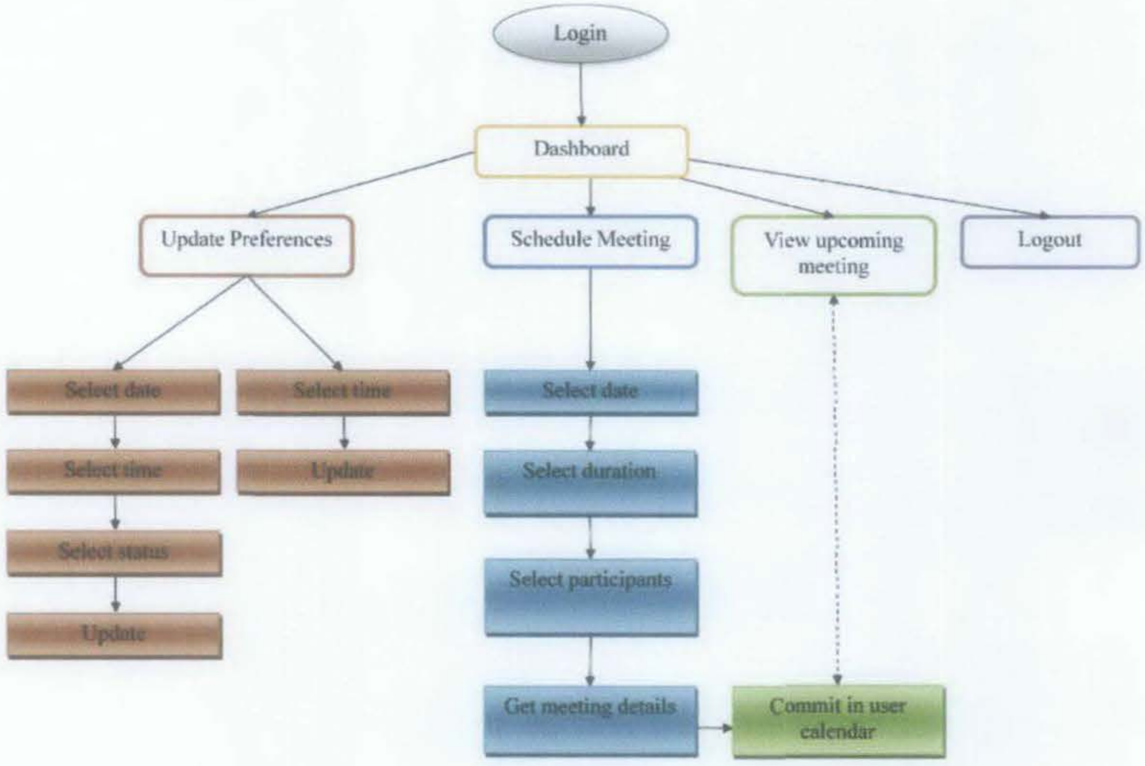


Figure 5: Flow of agent meeting scheduler

Basically, this is how the system prototype works: This system require user to log in. User can do so by entering their username and password to get access to the main page of the agent meeting scheduler. Upon logged in, user will be brought to dashboard page, where user may see their upcoming meeting details and options available offered by the system.

User has 3 options, whether to logout, update meeting preferences or to schedule a new meeting. Upon log out of the system, user will be redirected to login page. If user wants to update meeting preferences, user may do so in by clicking 'preferences' link. In 'preferences' page, user have option whether to update his or her availability for meeting, change preferred meeting time slot or both.

If user wants to schedule a new meeting, the user may do so by selecting "Schedule Meeting" link. User then will be redirected to another page where user needs to enter preferred meeting date, duration and participant involved for the meeting. Host agent then will perform its operation in getting the suitable meeting date and time here. Once the meeting date and time is decided, the meeting details will be displayed and committed to all participants' calendar. The details then can be viewed by participant in their dashboard.

4.2.2 System Architecture

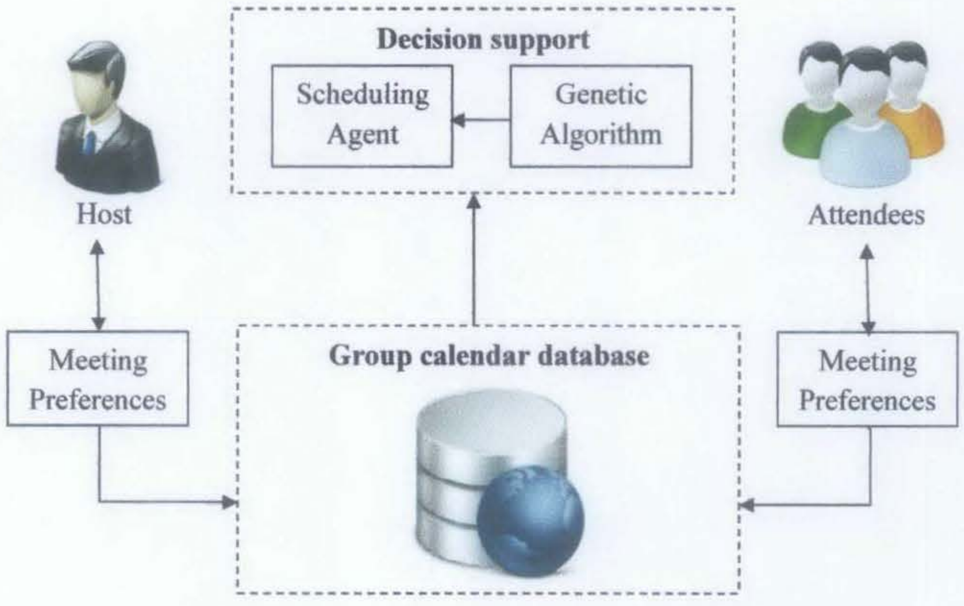


Figure 6: System Architecture

This is proposed agent meeting scheduler system architecture. It is supported by its main component which is decision support component. This component purpose is to schedule suitable meeting time given date by host agent. Every user of agent meeting scheduler has their personal agent. Each agent represents their owner. The project should is developed as a web based service in order to realize the agent usage in the system. The architecture is based on client-server where the meeting information is stored in web server database not in local desktop of the user. If a user wants to set up a meeting, the user is known as host and its personal agent is known as host agent. Host agent will pass meeting details such as date, duration and participants to scheduling agent. All the participant details of free time are gathered by scheduling agent by accessing the server database.

Scheduling agent will apply genetic algorithm to decide sequence of suitable meeting time for every participant. Upon receiving result of using genetic algorithm, scheduling agent then will check the participant's availability by using the time slot proposed from the result before. If there is available free time for all participants for the meeting on the selected date, scheduling agent will inform the host agent details of the meeting and the agent will pass it to the meeting host. If there is no mutual available free time of participant on the selected date, the scheduling agent will apply the result to the next date until it found time slot where all meeting participants are available. The result then will be sent to host agent and it will be passed to the meeting host.

4.2.3 Database Design

The author is using MySQL to develop database of the system and it will be maintained by using phpMyAdmin tool offered by XAMPP for Windows. There are 3 tables being used by the system which are meetingInfo, userPreferences and meetingUser.

The author first draft the necessary information that need to be stored by using quick notepad document. Later on, the author creates the database by using phpMyAdmin tool. This allowed the author to get rough overview of the database and test it directly by using local server.

The initial draft resulted in the following:

- meetingInfo – store meeting information of user (date, time and duration). This will be useful in order to track user free time. Each user will have its individual meeting information table. The holiday column is to indicate user unavailability to attend meeting. This table can be used to display user's personal schedule.
- userPreferences – this table store user's meeting preferences. These preferences will be used to determine suitable meeting time for that user.
- meetingUser – store user login information which is user name and password.

Given that user will input date, duration and meeting participants, the author try to track user free time by comparing the input value with content of MeetingInformation table. Before creating tables, the author examines the initial draft. Firstly the author start designing MeetingInformation table because it is required to be compared with user input data. Primary key of this table is MeetingId, where user can access the content from other table.

From this table, the author creates userPreferences tables which are relevant to be used in the system and lastly, the author create meetingUser table which is needed to implement login page. Below is entity relationship diagram (ERD) which presents relationship between the entities with the attributes or characteristics of entities in Agent Meeting Scheduler.

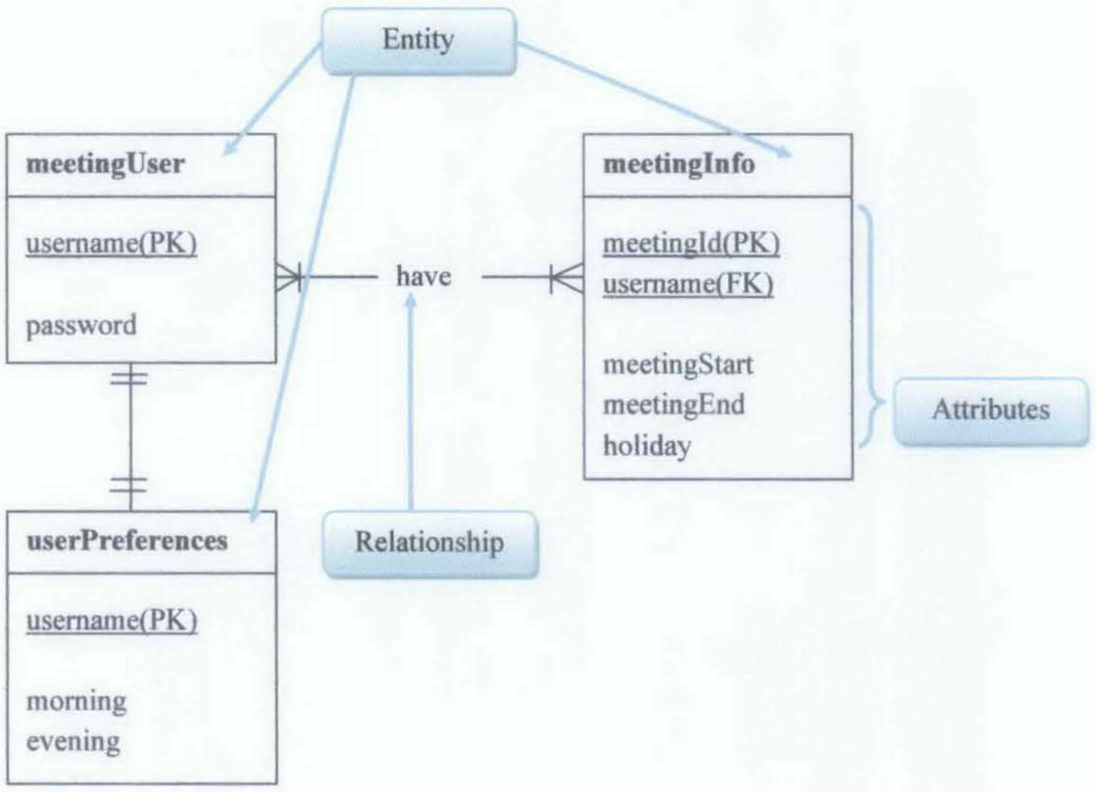


Figure 7: Entity Relationship Diagram

4.3 Prototype

Below is proposed agent meeting scheduler prototype GUI designs. The prototype consists of 4 main pages which are Login Page, Dashboard page, Schedule meeting page and Meeting preferences page. User is required to login before they can see their personal schedule in their 'dashboard'. The 'dashboard' will display user upcoming meeting and recently scheduled meeting. Dashboard also consist links to several features, which enable user to set up a meeting, update their meeting preferences and also logout.

User can change his meeting availability and time preferences in 'preferences' page. If user wishes to set up a meeting, user needs to input date of meeting, duration and meeting participants. Upon successful meeting coordinating, the user will be informed about meeting details and the meeting will be committed in all participants' schedule.

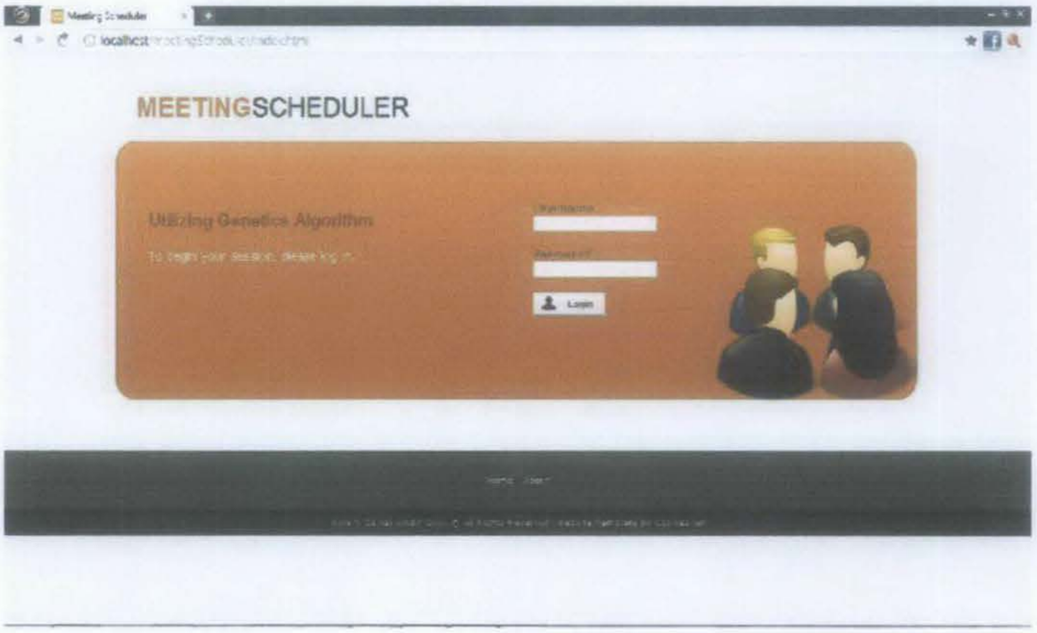
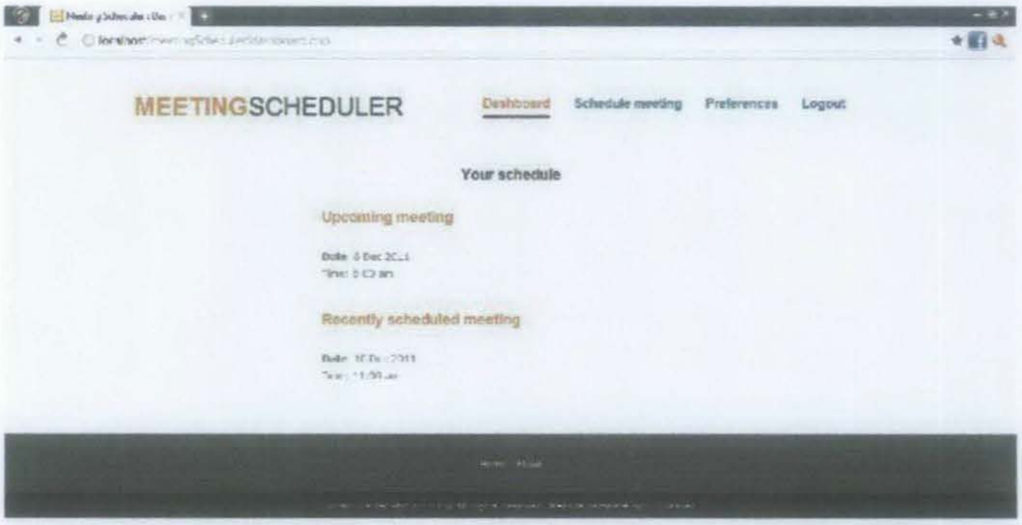


Figure 8: Login page



Your schedule

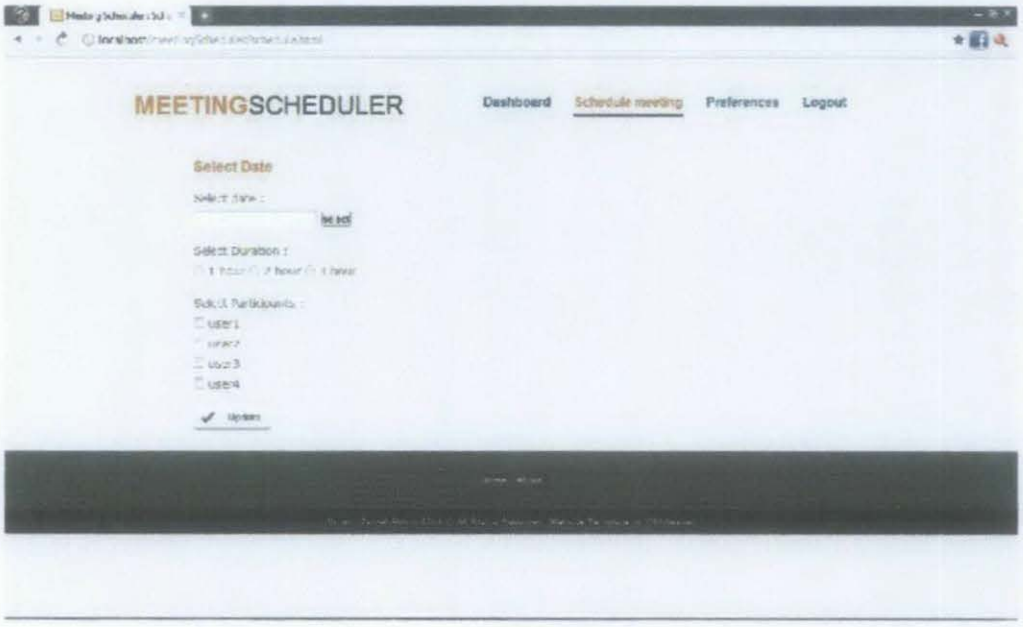
Upcoming meeting

Date: 8 Dec 2011
Time: 8:00 am

Recently scheduled meeting

Date: 10 Dec 2011
Time: 11:00 am

Figure 9: Dashboard page



Select Date

Select date :

Select Duration :
1 hour 2 hour 3 hour

Select Participants :
user1
user2
user3
user4

Figure 10: Schedule Meeting page

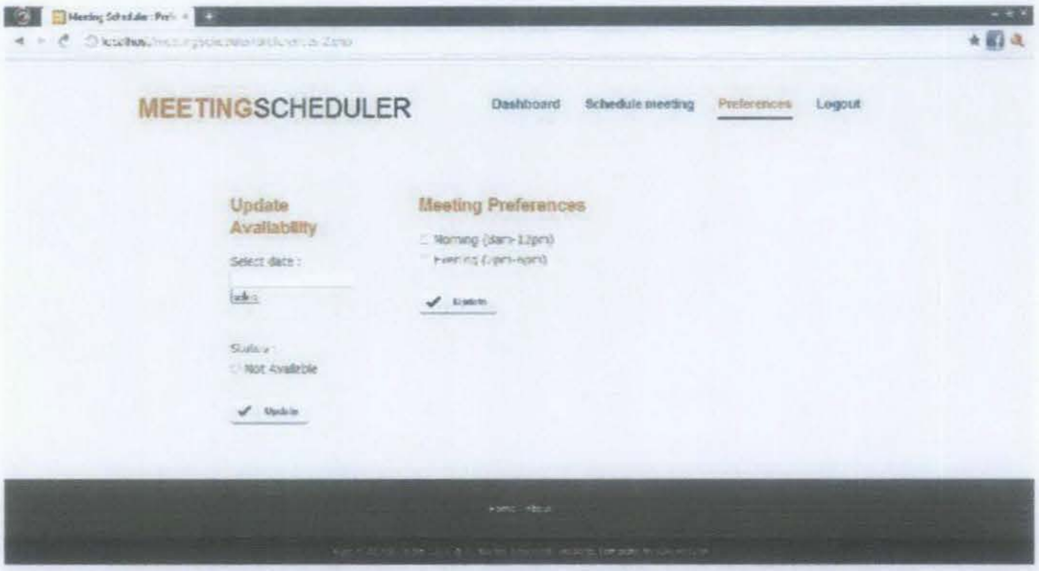


Figure 11: Meeting Preferences page

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

In the nutshell, the system development and progress are relevant to the project objectives which are to select suitable algorithm and prove that the system can be implemented by using the selected algorithm. In this project, genetic algorithm was selected based on their feature which is currently no agent meeting scheduler developer using sole genetic algorithm in designing the system. The author had designed the agent meeting scheduler to prove that it can also works by using genetic algorithm. This is a significant contribution to software agent field. Moreover, by using this agent meeting scheduler, meeting time coordination task can be carried out more efficiently. This product will save user time and make the company more organized. With the support of agent development tool, this project can be transformed into reality which will benefit all user especially corporation.

5.2 Recommendation

5.2.1 Suggested Future Work for Expansion and Continuation

- Use the most advance agent development tool for developing agent meeting scheduler
- Support more user in coordinating meeting time
- Combine genetic algorithm with other suitable algorithm to improve meeting scheduler efficiency
- Display the upcoming meeting in tabular calendar format for user convenience
- Integrate the system with other meeting scheduler

REFERENCES

1. Zunino, A., & Campo, M. (2009). Chronos: A multi-agent system for distributed automatic meeting scheduling. *Expert Systems with Applications , 2009* (36), 7011–7018.
2. Wooldridge, M. (2002). *An Introduction to MultiAgent Systems.* UK: John Wiley & Sons Ltd.
3. Shakshuki, E., & Koo, H.-H. (2006). Software Agents for Meeting Scheduler. *20th International Conference on Advanced Information Networking and Applications (AINA'06). 1550-445X.* IEEE Computer Society.
4. Jackson, A. J., & Jennings, N. (1995). Agent-based Meeting Scheduling: A Design and Implementation. *IEE Electronics Letters , 31* (5), pp. 350-352.
5. Lamsweerde, A. v., Darimont, R., & Massonet, P. (1995). Goal-directed elaboration of requirements for a meeting scheduler: problems and lessons learnt. *Second IEEE International Symposium,* (pp. 194-203). IEEE.
6. Garrido, L., & Sycara, K. (1996). Multi-Agent Meeting Scheduling: Preliminary Experimental Results. *Second International Conference on Multiagent Systems* (pp. 95-102). AAAI.
7. Lee, C. S., & Pan, C. Y. (2004). An intelligent fuzzy agent for meeting scheduling decision. *Fuzzy Sets and Systems* , 467–488.
8. Sugumaran, M., Narayanasamy, P., & Easwarakumar, K. S. (2006). An Effective Approach for Distributed Meeting Scheduler. *International Journal of Information Technology , 12* (8), 73-92.
9. Chun, H. W., & Wong, R. Y. (2003). N*—an agent-based negotiation algorithm for dynamic. *Advanced Engineering Informatics* , 1-22.
10. K Dahal, A Hossain, B Varghese, A Abraham, F Xhafa, A Daradoumis (2008). Scheduling in Multiprocessor System Using Genetic Algorithms. *2008 7th Computer Information Systems and Industrial Management Applications* (pp.281-286)
11. Knapik, M., & Johnson, J. (1998). *Developing Intelligent Agents for Distributed Systems.* McGraw-Hill.

12. Lee, R. S. (2006). *Fuzzy-Neuro Approach to Agent Application.* (T. Ishida, N. Jennings, & K. Sycara, Eds.) Hong Kong: Springer Berlin Heidelberg New York.

13. Lee, C.-S., Wang, M.-H., Wu, M.-H., Hsu, C.-Y., Lin, Y.-C., & Yen, S.-J. (2010). A type-2 fuzzy personal ontology for meeting scheduling system. *Fuzzy Systems (FUZZ), 2010 IEEE International Conference* (pp. 1-8). Barcelona: IEEE.

14. Bin Cai, Shilong Wang, Haibo Hu (,2011). Genetic Algorithm with Local Search for Job Shop Scheduling Problem. *AISS: Advances in Information Sciences and Service Sciences, Vol. 3, No. 9*, pp. 42 – 49

15. Asadzadeh, L., & Zamanifar, K. (2011). Design and implementation of a multi-agent system for the job shop scheduling problem. *International Journal of Computer Science and Security (IJCSS), 5*(2), 287.

16. Dahal, K., Hossain, A., Varghese, B., Abraham, A., Xhafa, F., & Daradoumis, A. (2008). Scheduling in multiprocessor system using genetic algorithms. *Computer Information Systems and Industrial Management Applications, 2008. CISIM'08. 7th*, pp. 281-286.

17. Lieberman, H. (1997). Autonomous interface agents. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 67-74.

18. Maulik, U., & Bandyopadhyay, S. (2000). Genetic algorithm-based clustering technique. *Pattern Recognition, 33*(9), 1455-1465.

19. Nakano, R., & Yamada, T. (1991). Conventional genetic algorithm for job shop problems. *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 474-479.

20. Shakshuki, E., & Hoo, H. H. (2006). Software agents for meeting scheduler. *Advanced Information Networking and Applications, 2006. AINA 2006. 20th International Conference on, , 2.* pp. 252-256.

21. Hunter, T. (2009, August 18). *Genetic Algorithm Traveling Salesperson PHP*. Retrieved October 18, 2011, from Renowned Media: http://www.renownedmedia.com/blog/genetic-algorithm-traveling-salesperson-php/

22. Jakobsen, T. (2006, May 30). *Classifier System Abstracts*. Retrieved October 20, 2011, from Troel's Home Page: http://subsimple.com/classifier.asp

23. Jankovic, M. (2008, January 22). *Making a Class Schedule Using a Genetic Algorithm*. Retrieved October 10, 2011, from The Code Project: http://www.codeproject.com/KB/recipes/GaClassSchedule.aspx

24. Obitko, M. (2008). *Introduction to Genetic Algorithms - Tutorial with Interactive Java Applets*. Retrieved October 13, 2011, from Obitko: http://obitko.com/tutorials/genetic-algorithms/

25. Skinner, M. (n.d.). *Genetic Algorithms Overview*. Retrieved October 3, 2011, from Genetic Algorithms Warehouse: http://geneticalgorithms.ai-depot.com/Tutorial/Overview.html

APPENDICES

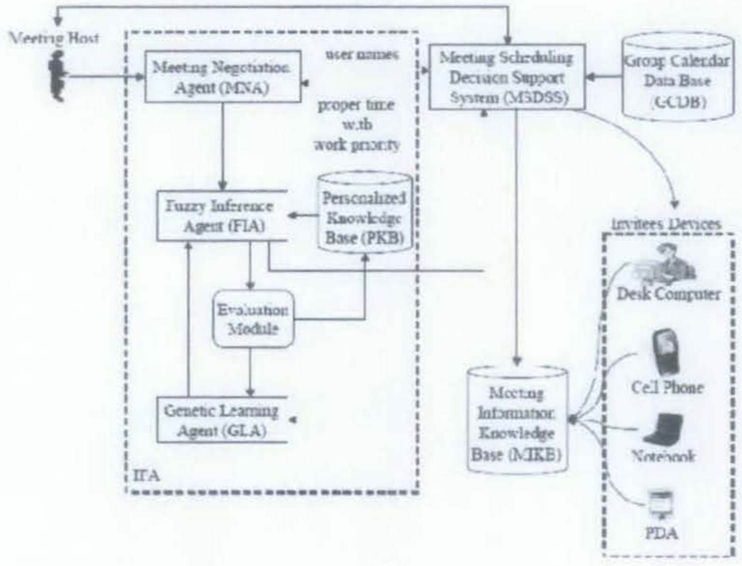

Meeting Host
Meeting Negotiation Agent (MNA)
user names
proper time with work priority
Meeting Scheduling Decision Support System (MSDSS)
Group Calendar Data Base (GCDB)
Fuzzy Inference Agent (FIA)
Personalized Knowledge Base (PKB)
Evaluation Module
Genetic Learning Agent (GLA)
IFA
Invitees Devices
Desk Computer
Cell Phone
Notebook
PDA
Meeting Information Knowledge Base (MIKB)

Figure 1: The architecture of the meeting scheduling decision support system using Fuzzy algorithm

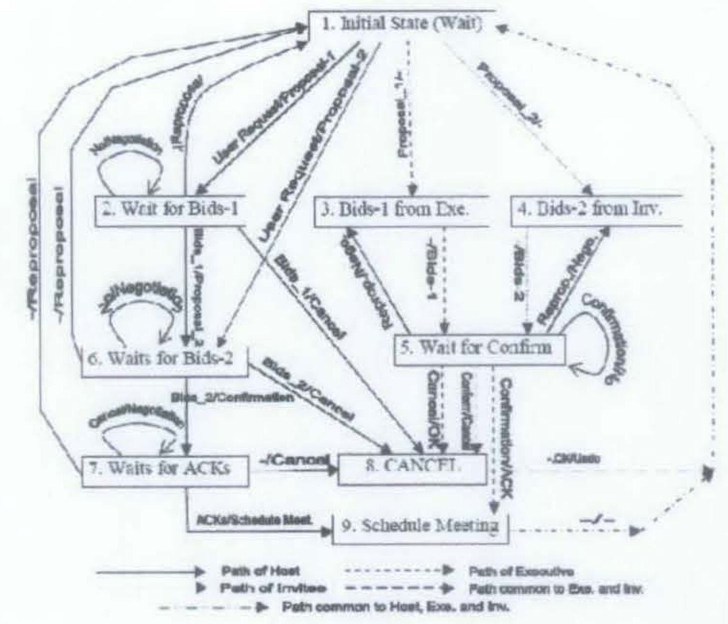


Figure 2: State transition diagram of agent using A*-algorithm

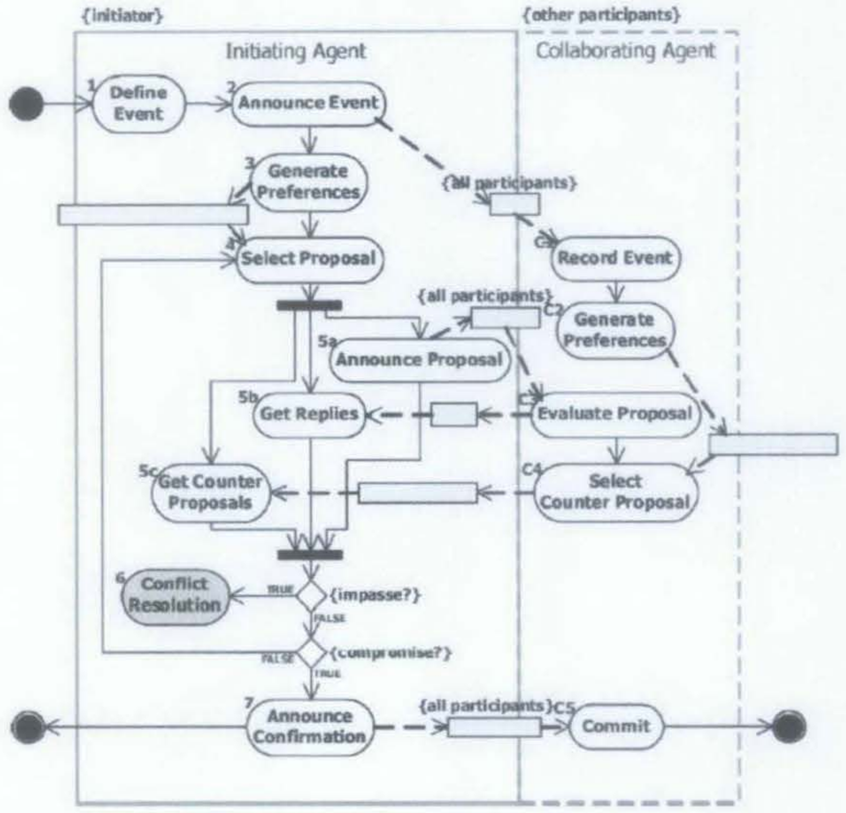


Figure 3: UML Activity diagram of negotiation N* algorithm

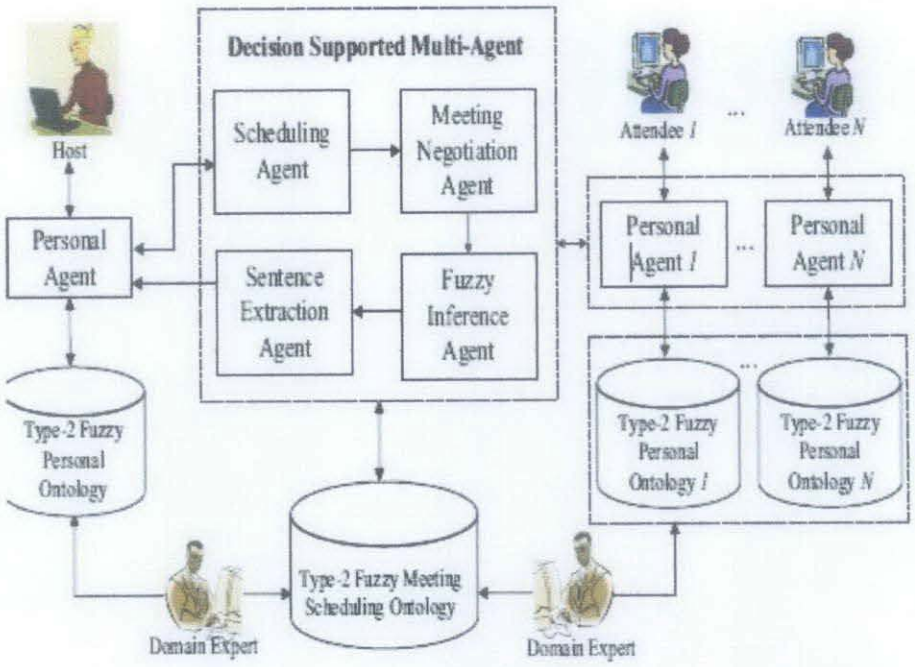
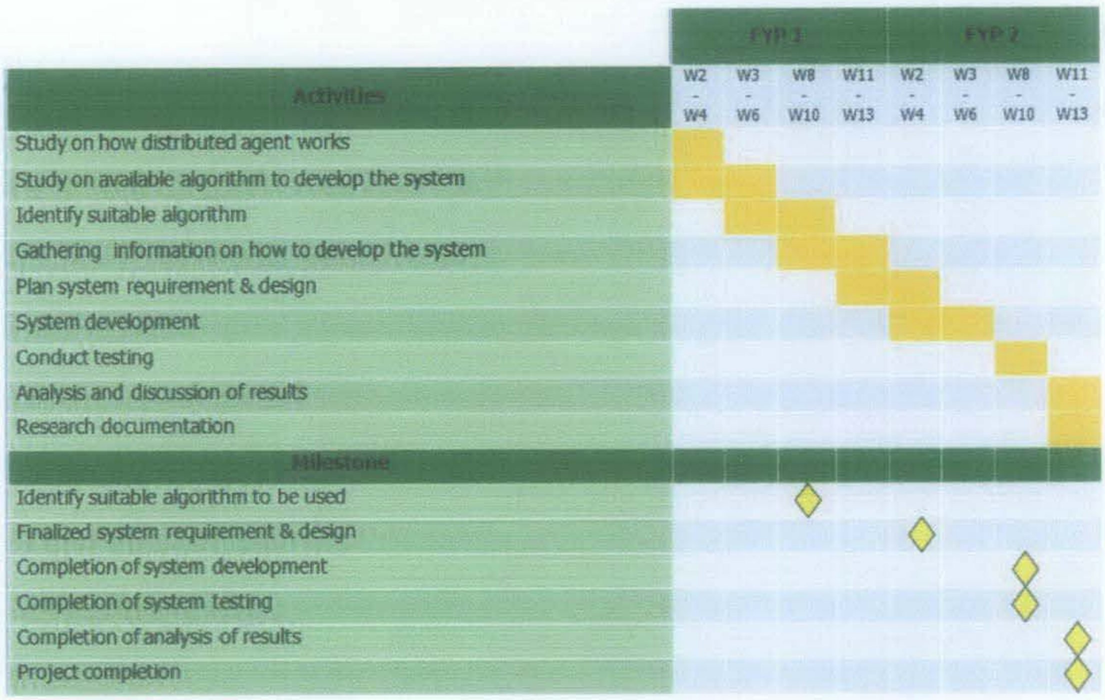

Decision Supported Multi-Agent
Host
Scheduling Agent
Meeting Negotiation Agent
Personal Agent
Sentence Extraction Agent
Fuzzy Inference Agent
Attendee *1*
...
Attendee *N*
Personal Agent *1*
Personal Agent *N*
Type-2 Fuzzy Personal Ontology
Type-2 Fuzzy Personal Ontology *1*
Type-2 Fuzzy Personal Ontology *N*
Type-2 Fuzzy Meeting Scheduling Ontology
Domain Expert
Domain Expert

Figure 4: System architecture of Fuzzy Type-2 Ontology Agent



Activities	FYP 1				FYP 2			
	W2 - W4	W3 - W6	W8 - W10	W11 - W13	W2 - W4	W3 - W6	W8 - W10	W11 - W13
Study on how distributed agent works	🟨							
Study on available algorithm to develop the system	🟨	🟨						
Identify suitable algorithm		🟨	🟨					
Gathering information on how to develop the system			🟨	🟨				
Plan system requirement & design				🟨	🟨			
System development					🟨	🟨		
Conduct testing							🟨	
Analysis and discussion of results								🟨
Research documentation								🟨
Milestone								
Identify suitable algorithm to be used			◆					
Finalized system requirement & design					◆			
Completion of system development							◆	
Completion of system testing							◆	
Completion of analysis of results								◆
Project completion								◆

Figure 5: Project Activities, Key Milestone & Gantt Chart