Implementation of an Intelligent Course Advisory Expert System

Cased-Based Course Advisory Expert System

*Olawande Daramola, Onyeka Emebo, Ibukun Afolabi, Charles Ayo
Department of computer and information sciences, Covenant University, Nigeria
olawande.daramola, emebo.onyeka, ibukun.afolabi, Charles.ayo{@covenantuniversity.edu.ng}

Abstract*— Academic advising of students is an expert task that requires a lot of time, and intellectual investments from the human agent saddled with such a responsibility. In addition, good quality academic advising is subject to availability of experienced and committed personnel to undertake the task. However, there are instances when there is paucity of capable human adviser, or where qualified persons are not readily available because of other pressing commitments, which will make system-based decision support desirable, and useful. In this work, we present the design, implementation, of an intelligent Course Advisory Expert System (CAES) that uses a combination of rule based reasoning (RBR), and case based reasoning (CBR) to recommend courses that a student should register in a specific semester by making recommendation based on the student's academic history. The evaluation CAES yielded satisfactory performance in terms of credibility of its recommendations, and usability.

***Keywords*—Academic advising; expert system; case-based reasoning; JESS; rule-based reasoning; evaluation**

I. INTRODUCTION

The quality of academic advising received by a student is crucial to the overall performance of the student. Good advising yields a good outcome while bad advising will be frustrating, and have a damaging in effect on students' progress. However, a staff advisor needs to keep up with the academic history of advisees in order to be an accurate and effective guide. This requires a lot of patience, commitment and ingenuity, which does not always exist, because humans have their limitations. In many scenarios, the rules for guiding students may change from time to time due to curriculum reviews, changes in course structure, or the circumstances of specific students. This makes it necessary for the human advisor to be adept in all the nuances of academic advising at all times. In many academic departments, the roles assigned to staff may change periodically, making it compulsory for the staff concerned to learn new rules that pertain to advising a new set of students. In addition, academic advising is time-consuming and mentally exacting involving many psychological and people management skills. All of these present a complex scenario that require good decision making, which at the same time places huge responsibility on the human advisor. Therefore, there is a need to alleviate the drudgery associated with academic advising through use of expert system to aid decision making. The use of an expert system will ensure the automation a significant part of the advisory process in a way that allows humans to do what it can do best, while the system complements human expertise by doing what it can do best, thereby creating a synergy that augur well for both staff and student. Hence, the essence of a course advisory expert system is not to replace the human advisor, but to minimize the cognitive load, and the time expended by the human advisor on academic advising, and to improve the quality of academic advising.

Course advising involves an academic staff giving counsel to a student on the courses to be register in a semester in order to satisfy established academic requirements that pertain to the student's academic programme. Students in a University are generally expected to satisfy some performance criteria in order to progress from one level to another, with a specified number of credit units to be passed among a set of compulsory (core), electives, and optional course. The role of the human course adviser is to ensure that a student makes good decisions on courses that should be registered relative to the student's current level, and academic history, in a way that satisfies graduation requirements. The course advisory task is a domain for application of expert system because – it is based on the use of domain specific knowledge, uses voluminous data, is difficult to characterize accurately, curriculum changes constantly and decisions have to be made based on the specific rules of a university. A lot of the decisions made by a human advisor during the process of advising a student are based on reasoning drawn from previous episodes and experiences that the advisor had gained over time, known rules of the University that relates to course registration. This suggests that a model of expert system that uses Case Based Reasoning (CBR), and rule-based reasoning for decision making would be viable for academic advising.

CBR is pattern-based problem solving paradigm that relies on knowledge gained from previous episodes to resolve new problems once sufficient similarity can be established between the current case (problem) and past cases that are stored in case base (repository) [1]. The attraction for using CBR as the mode of reasoning for academic advising is based on fact a lot of similarities exist in the nature of academic problems and concerns that students have in the process of course registration. Hence, the combination of CBR and rule-based reasoning – which enables the consideration of specific university rules for decision making – in order to develop an expert system for student advising is to emulate human expertise to reasonable extent, by drawing on the similarity that exist in experiences gained in previous cases of course advising, and an awareness of relevant university rules.

This paper describes the implementation of an intelligent course advisory expert system (CAES). The expert system uses the combination of rule-based reasoning and case-based reasoning to generate credible recommendation to guide students on courses to register. The objective of the system is to reduce the effort, and time used in the process of student advising, and to improve quality.

The remaining part of the paper is described as follows. In section 2, we present related work. Section 3 discusses the course registration process and the requirements for a Course Advisory Expert System (CAES). Section 4 gives a description of the architecture of the CAES and the process of applying the CAES. Section 5 gives an outline of algorithms that enable some of the core functionalities of the CAES. Section 6 reports a case study of the application of the CAES in a tertiary institution. Section 7 a preliminary evaluation of the CAES and the result. The paper is concluded in section 8 with a brief comments and outlook of work for the future.

II. Related Work

The desire for technology-supported academic advising has been around for a while, and a number of efforts have been reported in the literature. In [2], the evaluation of a Web-based decision support tool that aids student advising was reported. The evaluation of the tool showed that large percentage of respondents regard it as effective and efficient for academic advising, however, the details of its implementation was not provided in the paper. In [3], the design of a i-Counselling system that combines ontology-based information retrieval and optimization-based search technology to provide relevant answers to queries posed by new and current students was reported. The academic advising module of the system is able to answer questions from current students on specific programmes, study plans and graduation requirements. The system was adjudged effective after an evaluation was conducted. HE-Advisor [4], is a multidisciplinary Web-based higher education advisory system that offers academic advisory services in order to students make the best decision in selecting a degree to study. It also incorporates guidance on course registration to assist students to stay on the right path towards concluding their degree, information on graduation requirements and statistics for timetable planners were also provided by the system. The ViCurriAs [5] is a visual tool that facilitates the registering of new curriculum plans and track the progress of students enrolled for a degree programme.

Other types of expert systems or hybrid intelligent systems that have been used for academic advising include [6, 7, 8]. In addition, in [9], the concepts of intelligent agents and semantic Web were used to develop an academic advisory system. The domain knowledge was modeled with the OWL ontology language, while the agents reason on stored domain knowledge by using an inference engine. The work in [10] presents the architectural framework of an intelligent advisory system that uses the concepts of object-orientation and knowledgebase rules for academic advising. The objective of the system is to help students to know what to do and how to do it.

In [11] the Interactive Virtual Expert System for Advising (InVEStA) was reported. InVEStA was designed to assist undergraduate students and their advisors in providing timely, accurate and conflict-free schedules. The system was implemented using Java and an object-relational database. It comprises a Database Layer, Transaction Layer, Scheduler and the web-based Front-End.

The Graduate Course Advisor (GCA) is a rule-based expert system that advises graduate students of computer science [12]. The GCA is a Prolog-based system that was modelled after MYCIN. GCA divides advising into four phases such that each phase may apply the inference engine to its own rule base and invoke other procedures. The CBR Recommender for Academic Advising (AACORN) was presented in [13]. AACORN uses course histories to generate recommendations for course advising. By reusing the knowledge embedded in a student's academic history as captured in student's transcripts, AACORN is able to make reasonable suggestions with a limited amount of domain knowledge. The edit distance was used to determine the similarity between the course history of a new student and other course histories in the case base.

The intelligent Course Advisory Expert System (CAES) presented in this work differs from other previous approaches because it integrates the use of CBR and rule-based reasoning to generate intelligent recommendations for students on courses to register. In sequel section, we shall discuss the architecture of system in more detail.

III. An Overview of the Course Registration Process

The procedure for course registration by a student entails a series of activities. The procedure includes

1) authenticate the status of student to determine if student qualify to be registered into a particular level based on previous academic performance;
2) select a course to be added to the list of registered courses by student;
3) add or drop a course after initial registration; validate course prerequisites; and
4) check the rules that guides total numbers of course to register and combination of courses to register.

It is expected that for a healthy process both the student and the advisor much have adequate understanding of the procedure in order to avoid violations. Fig. 1 shows the key uses cases that pertain to a course registration scenario.

A more detailed analysis of the use cases captured in Fig. 1, revealed a number of specific requirements that a course advisory system must meet. These include:

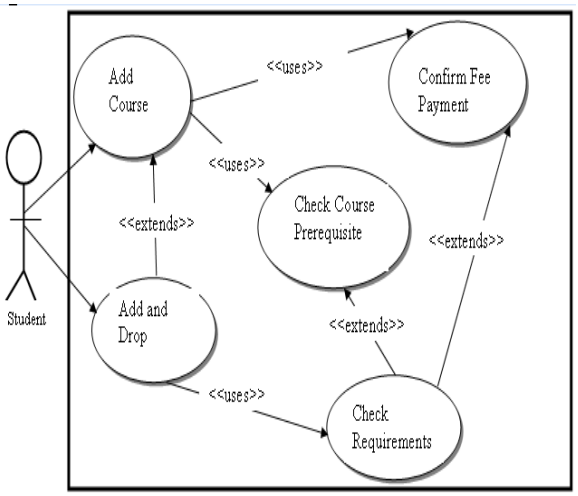

Fig. 1 A Use Case Diagram of the Course Registration Process

1. *The System shall be able to authenticate the status of every user as either a valid student user or staff user.*
2. *The System shall only allow courses to be added or dropped during the date period allocated for course registration.*
3. *The System shall capture detailed general student information including as department, level, and college.*
4. *The System shall capture detailed information on students examination results including failed, passed and dropped courses.*
5. *The System shall capture all relevant university rules that pertain to registration.*
6. *The System shall be able to give recommendation to a user once the valid his valid status is determined.*
7. *The System shall provide explanation for all recommendations suggested to the user.*
8. *The System shall provide real-time feedback when the user requests a recommendation.*

These set of requirements provided the basis for the design and implementation of the CAES.

IV. Architecture of the Course Advisory Expert System (CAES)

The CAES is based on a three-tier architecture that consists of a presentation layer, a middle layer and a data layer (see Figure 2). The presentation layer enables the user to access the application via a browser by using client devices such as desktop, laptop, or mobile phones. The various graphic user interfaces (GUIs) through which the user interacts with the system are contained in this layer.

The middle layer consists of the Web application server, which facilitates communication in form of requests and notifications between the clients and the CAES application using the HTTP protocol. Apache Tomcat was used as the web application server for the CAES. The middle layer also contains the rule-based engine (RBR), which was

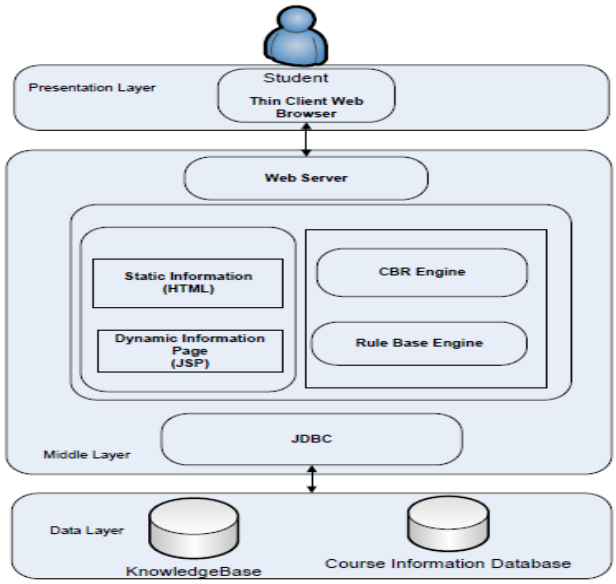

Fig. 2 The 3-tier architecture of the system

implemented using Java Expert System Shell (JESS)[1] in order to enable reasoning on the rules that pertain to student registration; the case-based reasoning (CBR) engine enables case based reasoning. The RBR and CBR engine are deployed on the web application server. The middle layer also contains the Java servlets and JSP components that provided basis to weave java codes round the RBR and CBR engines of the CAES. The Java Data Base Connectivity (JDBC) protocol that enables interaction with the data layer of the architecture is also a contained in the middle layer.

The Data Layer contains the data and knowledge artifacts that the system relies on to deliver its functionality. This layer consist of a knowledge base that contains the facts and rules (Jess fact files and rules) that is used by the RBR engine, and the relational database that contains information on all courses that are available in the University.

A. Using the CAES for Advising

In order to use the CAES for academic advising, the user will need to do the following:

1) Input a valid identification number at the CAES GUI
2) If successful, the CAES interface will display student details from the course information database. Displayed information will include current cumulative grade point average (CGPA), passed courses with grades obtained, failed courses, dropped courses, and the set of courses to register for the current semester.
3) Click recommend to generate a list of suggested courses to register for the new semester
4) Click on view explanation to see rationale for recommended courses.

The Inference engine comprising of the rule engine and the CBR engine are used to generate recommendation of courses to be registered in a current semester.

[1] http://herzberg.ca.sandia.gov/

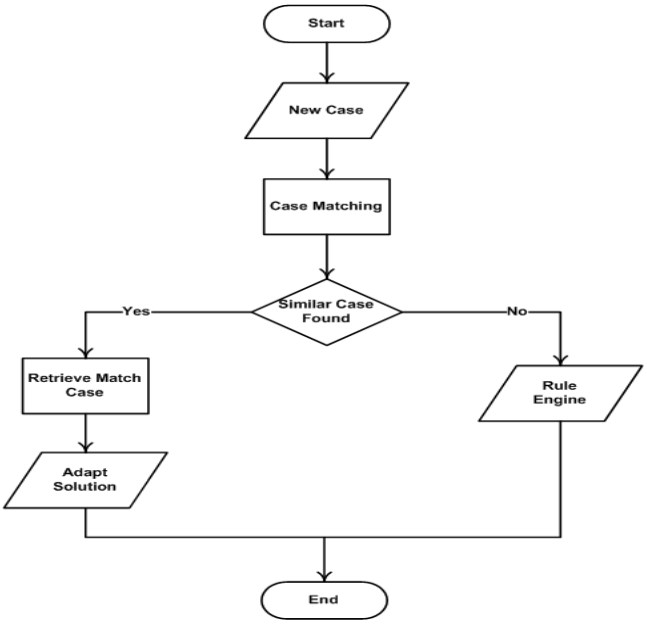

Fig 3. Schematic representation of CAES recommendation

The Inference mechanism checks to see if there are previous cases that are similar to the current case by taking note of the courses failed and dropped by a student and his current level. All of these factors are considered in generating an advice for the student. Case based reasoning is used because the system computes a recommendation by scanning the case base for instances that are similar to the one at hand and adapts the most similar old solution in a new scenario. The report is sent back to the student via the CAES GUI. If no similar case exists then rules contained in the rule base are used to construct a recommendation based on deductions that can be made using information available on student's level, failed courses, failed prerequisites, and maximum total of credits that can be registered. CAES retains in the case base, all cases that have been handled successfully.

An investment to be made in order to productively engage the CAES is that an administrator must continually maintain the case base to ensure that course information, the rules in the knowledge base are regularly updated. This is to ensure that the CAES system have the correct basis to make its recommendation during academic advising.

The Figure 3 is a schematic representation of the recommendation process of the CAES using a program flowchart.

V. THE REASONING MECHANISM OF THE CAES

In this section, we give some insight into the reasoning behind some of the recommendations of the CAES.

When CAES starts, the student course information is considered as a new case. CAES then computes a similarity score for the new case using the algorithm.

$$Similarity\ (NC, OC) = \frac{common}{common + different}$$

Where *NC* is the new case, *OC* is the old case present in the case base.

Common refers the matching pair between the new case and an old case.

Different refers the mismatch pair between the new case and an old case.

The case with the highest similarity score is picked as the candidate for adaptation in order to recommend courses to register to a user. If a similar case does not exist, then a decision algorithm based on the rule engine is used to generate recommendation. The case adaptation procedure is rule-based, whereby university rules are used to guide selections. Two rules were used 1) a course with a higher credit unit should be selected over a course with a lower credit unit; 2) compulsory courses take precedence over electives and optional courses; 3) a course that is pre-requisite for another that is failed, should be considered over courses that are not prerequisite for any other course.

VI. CASE STUDY AND DISCUSSION

A case study of Covenant University a tertiary institution based in Ota, Nigeria was undertaken using students of the Computer science study program of the University as subjects. For a student intending to register a course at the beginning of a new semester these scenarios exist.

i) The student could have just the current semester course to register.

ii) The student could have failed course(s) alongside the current semester courses.

iii) The student could have dropped course(s) alongside the current semester courses.

iv) The student could have failed and dropped course(s) alongside the current semester courses.

In recommending the set of courses to register for the current semester, CAES uses the scenario above that is applicable to that particular student together with the set of rules outlined by the University policies for course registration, putting into consideration the different course status (course perquisites, compulsory or elective courses).

The different conditions were modelled as rules and stored in the rule base of CAES. The set of algorithms showing the rationale for specific in the rule base of CAES.

The *REGISTERDROPPEDFAILEDCOURSE* algorithm in Table 1 caters for the scenarios i) –iii), while the *REGISTERCOURSE* algorithm in Table 2 caters for scenario iv). Table 3 shows sample JESS rule that states that a compulsory course have precedence over other types of courses. Also, Figure 4.1 and Figure 4.2 show snapshot of the CAES application.

Table 1. Register Dropped and Failed Courses Algorithm

Algorithm *REGISTERDROPPEDFAILEDCOURSE (**V, E, S**)*
Input: *A vector **V** of courses failed and/or dropped in the previous session of the same semester, E a vector of elective courses and **S** a vector of courses to register in the current session of the same semester.*
Output: *A vector **R** containing the list of courses recommended for registration by the student in that semester.*

Initialize R.
[Considering Failed and Dropped courses]
for *all courses $v_i \in V$ ordered by coursecode in ascending order*
while *registeredCredit < maxRegistrable AND i < count(V)*
Add v_i to R.
registeredCredit ← registeredCredit + courseCredit(v_i)
increment i.
[Considering failed prequisite course]
If *registeredCredit < maxRegistrable*
for *each course $C_j \in S$*
while *registeredCredit < maxRegistrable **AND** j < count(S)*
if *prequisite(C_j) is failed **OR** dropped*
then *Add C_j to D*
else
Add C_j to R
S ← S- C_j
registeredCredit ← registeredCredit + courseCredit(C_j)
increment j.
[For the remaining courses]
If *registeredCredit < maxRegistrable*
for *each course $K_p \in S$ that is compulsory ordered by course credit in descending order*
while *registeredCredit < maxRegistrable **AND** p < count(S)*
Add K_p to R
registeredCredit ← registeredCredit + courseCredit(K_p)
increment p.
If *registeredCredit < maxRegistrable*
for *each course $M_e \in E$ that is elective*
while *registeredCredit < maxRegistrable **AND** e < count(E)*
Add M_e to R
registeredCredit ← registeredCredit + courseCredit(M_e)
increment e.
return *the vector **R** containing the list of recommended course for the semester.*

Table 2. Register Dropped and Failed Courses Algorithm

Algorithm REGISTERCOURSE (**E, S**)
Input: A vector **E** of Elective courses and **S** a vector of courses to register in the current session of the same semester.
Output: A vector **R** containing the list of courses recommended for registration by the student in that semester.

R ← NULL [initialize R]
for each course $C_i \in S$
while registeredCredit < maxRegistrable AND i < count(S)
if prequisite(C_i) is passed
Add C_i to R
registeredCredit ← registeredCredit + courseCredit(C_i)
increment i
If registeredCredit < maxRegistrable
for each course $K_j \in S$ that is compulsory ordered by course credit in descending order
while registeredCredit < maxRegistrable AND i < count(S)
Add K_j to R
registeredCredit ← registeredCredit + courseCredit(K_j)
increment j
If registeredCredit < maxRegistrable
for each course $M_e \in E$ that is elective
while registeredCredit < maxRegistrable AND e < count(E)

Table 3. Sample JESS rule to select a Compulsory Courses

```
(defrule recommend-compulsory-course
 "If there is a compulsory course, recommend for registration"
 ;; The course belongs to the type department and is compulsory
 (course (belongsTo department) (ccode ?code)(ctitle ?title) (cunit ?unit) (cstatus compulsory))
    ;; and we haven't recommended this type yet
 (not (recommendcourse (ccode ?code) (ctitle ?title)))
 =>
 ;; Recommend the course.
 (assert (recommendcourse (ccode ?code) (ctitle ?title) (cunit ?unit) (because "compulsory departmental course"))))
```

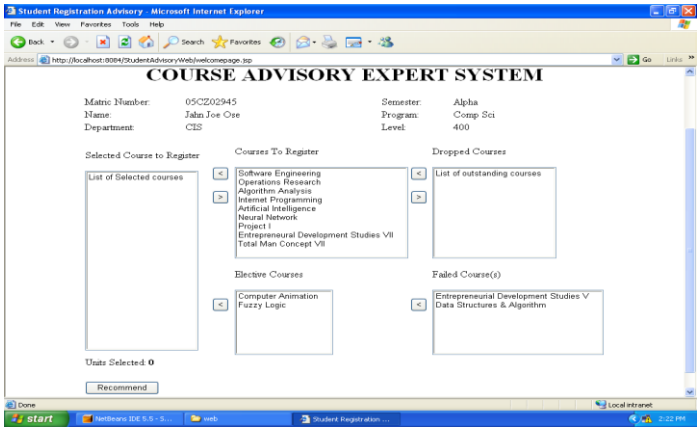

Fig. 4. The CAES Interface

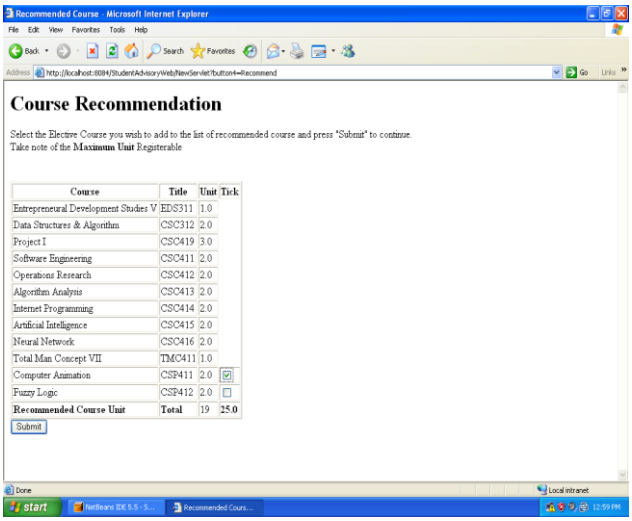

Fig. 5. CAES Recommendation Page

VII. EVALUATION

Human experts conducted a usability evaluation of the prototype in order to assess the level of user satisfaction with the system. This was then validated through the direct method of evaluating expert systems as used by Salim et al. [14].

A small experiment to test the system's recommendations against those of human advisors was conducted using the direct method. Course Advisers across each level from the Department of Computer and Information Sciences of Covenant University were asked to participate in the survey. Each received an identical set of questionnaire, and had a running version of CAES installed for them. The course advisers were asked to rank the recommendation of CAES on a likert scale of 0-5 to assess the degree of how true or false are the recommendations of CAES.

A brief overview of the direct method of expert system evaluation used by each evaluator is as follows:

1. The evaluator obtains demonstration or sample copies of the software packages to be evaluated.
2. The evaluator selects a benchmark problem, based on his experience, and runs this problem on CAES.
3. After running the bench-mark problem, the evaluator responds to the 14 questions in the questionnaire instrument and estimates a quantitative answer to each question on a 0 to 5 scale with 5 being very true and 0 being very false.
4. Each numerical result is multiplied by a weighting factor as given in the weight column.
5. The weighted values are summed and then divided by the sum of the weights (19) to give a result in the numerical range of 0 to 5.

The Figure 6 gives a computation of the evaluation experiment conducted by one of the evaluator.

	Question	Assessment Value	Weight	Value x Weight
	Correctness of Answer			
1	Is there enough information to evaluate the software?	4	(2)	8
2	Does the software give the same answer that a human advisor would give?	5	(2)	10
3	Does the software provide the right answer for the right reasons?	5	(2)	10
	Accuracy of Answer			
4	Is the software accurate in its answer(s)?	5	(2)	10
5	Is the answer complete? Does the user need to do additional work to get a usable result?	4	(2)	8
	Correctness of reasoning technique			
6	Does the answer change if new but irrelevant data is entered into the software?	5	(1)	5
7	Does the system require a lot of irrelevant question to reach its answer?	0	(1)	0
	Sensitivity			
8	Does the answer change if irrelevant changes are made to the system rules?	5	(1)	5
	Reliability			
9	Does the software crashes or hang ups in its host computer?	2	(1)	2
10	Does the system give warnings for cases involving incomplete data or rules?	5	(1)	5
	Cost Effectiveness			
11	Does the software still provide answers with incomplete knowledge	2	(1)	2
12	Is the cost of the system justified by its performance?	5	(1)	5
	Limitations			
13	Can limitations of the system be detected at this point in time?	4	(1)	4
14	Can the system learn from increased data or experience?	2	(1)	2
	Result = Σ (weight x value) / Σ (weight)		19	76
			4.00	

Fig. 6 Evaluator's questionnaire

A subset of the summary result in calculating the experimental evaluation of the evaluators is given in the Table 4.

Table 4. Result of Evaluation Experiment

Evaluator	Computed Satisfaction Level
1	4.00
2	4.16
3	4.21
4	3.52
5	3.57
Mean Satisfaction Level	3.89

From the statistical analysis of the results obtained from the evaluation of the human experts that participated in the experiment, CAES had a mean satisfaction level score of 3.89 out a maximum of 5.0, which is indicative of a 77.8% level of user satisfaction.

The result revealed that the system had a performance rating of 77.8% by the human expert that evaluated the system.

VIII. EVALUATION

The CAES system that was developed is intended for use in a mid-range universities. Currently, its experimental version being utilized by the Department of Computer and Information Sciences of Covenant University, its modular structure and web-based design makes it possible to be launched and used in other departments of the University.

In our future work, we shall improve on the case revision and case adaptation capability of the CAES, because we observed some complex cases, which the system did not handle adequately. This had to do with students that have changed from one programme to another - many of them more than once - , and have failed and dropped courses that are spread among different departments. We observed that in such scenarios, it was difficult for the current implementation of CAES find good cases to use as basis for adaptation to construct a recommendation. We do not consider this to be major drawback of CAES, because even for the human course adviser, cases where a student has failed multiply in different departments are more intricate to handle, yet we seek to improve CAES in this areas.

As its contribution, this work offers a demonstration of application of artificial intelligence technology (AI) to support academic advising, which is very crucial to the academic well-being of students. The CAES was not intended to eliminate the role of human (staff) advisors, rather it enables students to concentrate on real issues that pertain to course registration, and affords unrestricted access to expert advice thereby reducing the burden placed on the human advisor.

REFERENCES

[1] Aamodt, A., E. Plaza, 1994. Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI communications, 7 (1): 39-59.

[2] Feghali, T., I. Zbib and S. Hallal, 2011. A Web-based Decision Support Tool for Academic Advising. *Educational Technology & Society*, *14* (1): 82–94.

[3] Chun M., Y. Eva, S. Tsang, S. Lam, C. Dominic, and C. Pang, 2010, "Intelligent Counseling System: A 24 x 7 Academic Advisor", *EDUCAUSE Quarterly* 33(4).

[4] Albalooshi, F., and S. Shatnawi, 2010. HE-Advisor: A Multidisciplinary Web-Based Higher Education Advisory System, Global Journal of Computer Science and Technology, 10(7):37-49.

[5] Zucker, R., 2009. ViCurriAs: a curriculum visualization tool for faculty, advisors, and students. Journal of Computing Sciences in Colleges, 25(2):138-145.

[6] Harlan R. M, 1994. The Automated Student Advisor: A Large Project for Expert Systems Courses. *ACM SIGCSE Bulletin* 26(1):31- 35.

[7] Rao T., S. Coleman, and C. Hollenbeck, 1987. ADVISOR – An Expert System for Student Advisement. *Proc. 15th Annual Conference on Computer Science, St Louis*, pp. 32-35.

[8] Murray W., and L. LeBlanc, 1995. A Decision Support System for Academic Advising. *Proc. 1995 ACM Symposium on Applied Computing*, pp. 22-26.

[9] Dunkel J., and R. Bruns, 2005. Software architecture of advisory systems using agent and semantic Web technologies. Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, 19:418 – 421.

[10] Gunadhi H., L. Kwang-Hui, Y. Wee-Yong, 1995. PACE: a planning advisor on curriculum and enrolment. Proceedings of the Twenty-Eighth Hawaii International Conference on System Sciences, 3:23 – 31.

[11] Pokrajac D., M. Rasamny, 2006. Interactive Virtual Expert System for Advising (InVEStA). 36th Annual Frontiers in Education Conference. pp. 18-23.

[12] Valtorta M., B. Smith, and D. Loveland, 1984. The graduate course advisor: a multi-phase rule-based expert system. Proceedings of the IEEE Workshop on Principles of Knowledge-Based Systems.

[13] Sandvig, J. and R. Burke, 2005. AACORN: a CBR recommender for academic advising. Technical Report, TR05-015, DePaul University.

[14] Salim M., A. Villavicencio, and M. Timmerman, 2002. A Method for Evaluating Expert System Shells for Classroom Instruction. *Journal of Industrial Technology,* Vol. 19(1).