AdviseMe: An Intelligent Web-Based Application for Academic Advising

Lawrence Keston Henderson
Department of Computing and Information Technology
University of the West Indies
St. Augustine, Trinidad

Wayne Goodridge
Department of Computing and Information Technology
University of the West Indies
St. Augustine, Trinidad

Abstract—**The traditional academic advising process in many tertiary-level institutions today possess significant inefficiencies, which often account for high levels of student dissatisfaction. Common issues include high student-advisor loads, long waiting periods at advisory offices and the need for advisors to handle a significant number of redundant cases, among others.**

Utilizing semantic web expert system technologies, a solution was proposed that would complement the traditional advising process, alleviating its issues and inefficiencies where possible. The solution coined 'AdviseMe', an intelligent web-based application, provides a reliable, user-friendly interface for the handling of general advisory cases in special degree programmes offered by the Faculty of Science and Technology (FST) at the University of the West Indies (UWI), St. Augustine campus. In addition to providing information on handling basic student issues, the system's core features include course advising, as well as information of graduation status and oral exam qualifications. This paper produces an overview of the solution, with special attention being paid to the its inference system exposed via its RESTful Java Web Server (JWS).

The system was able to provide sufficient accurate advice for the sample set presented and showed high levels of acceptability by both students and advisors. Furthermore, its successful implementation demonstrated its ability to enhance the advisory process of any tertiary-level institution with programmes similar to that of FST.

Keywords—*Web-Based Academic Advising; Academic Advising; Ontology; Jena; Expert Systems*

I. INTRODUCTION

A. Background - Academic Advising

In the realm of tertiary education, academic advising is a student-advisor collaborative process [1] designed to enhance a student's overall educational experience by lending academic decision support to them. This is done by analysing the student's academic records and external factors (academic capabilities, interests, daily schedules and financial constraints) in order to produce customized advice [2]. Such advice would then allow the student to make informed decisions so that they can develop an academic plan to complement their personal life goals and complete their course of study within the prescribed period, or with minimal excess from that date.

The advising process is long-term and iterative due to the continuous change of the environment it operates within [3]. Such changes include the addition and removal of courses from programmes as well as modification of prerequisite rules. It also has timely limitations, as advisors cannot lend advice for future semesters since it is difficult to predict which courses a student will pass, if any during the course of any semester [4]. As a result, student advising should be made available, at minimum, once per semester to ensure that students are guided based on the latest versions of their transcript and the rules that govern their study programmes.

Academic advising can be categorized into four major systematic models: prescriptive, developmental, integrated and engagement. In prescriptive advising, students succumb to the direct advice given by advisors, making advisors solely responsible for the decision making process. With developmental advising however, the advisor directs the student to the proper resources and the decision making process is shared between both parties with more responsibility being placed on the student, thus fostering a higher level of 'student-independence' [5]. Integrated advising is a fusion of formerly discussed methods and engagement advising is typically a type of developmental advising, with increased student-advisor meetings [6]. It was noted however that intuitive students typically endorsed a developmental advising model while others seldom valued a collaborative relationship and hence seemed more content with that of a prescriptive advising model [6].

There are two core methodologies associated with selecting courses based on interests and prerequisites completed. The 'bottom-up' approach is used in most structured programmes whereby an initial set of core courses (with no prerequisites) is selected and course selection per semester continues as these courses are successfully completed. Conversely, in the 'top-down' approach, advanced courses in which the student has interest is initially analysed. The prerequisites of these courses are then derived and the student takes these in order to reach his goal [7]. This approach is used more in unstructured programmes such as general degrees with specializations in particular fields.

It is crucial that a student receive proper advising as poor or no advising can have severe repercussions on his progress throughout his course of study, possibly resulting in delayed graduation [8]. In some cases, the advising process may demand more qualitative judgements before a reasonable decision can be attained. This can be due to personal student issues external to the university's academic context. As a result, academic advising does not only entail course advising, but is also designed to support and motivate students throughout

their academic life so that they can comfortably accomplish their educational goals [9]. In this light, advisors must have full knowledge of the student's background, academics, plans and goals in order to give effective academic advice. This may therefore require frequent 'one-on-one' student-advisor meetings so that a relationship can be forged between both parties whereby they understand the student's unique needs. Such advising however can be difficult to achieve mainly due to the availability of experienced and committed persons to undertake the task [8].

B. The Traditional Academic Advising Process

At the University of the West Indies (UWI), St. Augustine campus, integrated academic advising is handled per faculty, with sub-advisory units at each department. In the faculty of Science and Technology (FST), this manual process is spearheaded by the Deputy Dean who has assigned one advisor to each department. Each advisor is expected to handle all student matters under his purview. Also twenty-five peer advisors have been trained to handle general advisory matters inclusive of how to request overrides and determine qualification for oral assessments, among other things. Up to the first month into the semester, advisors are expected to handle student cases for extended periods of time, ranging on average between four to six hours per day. At other times, advisors would conduct advising services generally around one to three hours per day.

The advisory process begins with the student filling (or updating) a paper-based form outlining the courses already taken (if any) by semester, with the corresponding grades obtained. This represents their advising profile and therefore must be kept and maintained throughout their entire academic life. The form would also have comments made by past advisors indicating what advice was given. On meeting with a student, the advisor retrieves the student's transcript from the Student Information System (SIS) to verify that the information on the form is accurate. He then uses a 'bottom-up' methodology, mapping the student's completed courses against those required for the student's programme of study. The course listings are usually taken from references such as faculty handbooks and departmental handouts and are used to deduce what courses the student should take in the upcoming semester.

For cases where students have completed a semester of courses and simply need a new set to attempt for the upcoming semester, the advising process ends following course suggestion. For special cases influenced by external factors however, additional time is required for advisors to learn the student's situation and perform more qualitative analysis before relating sufficient advice to the student.

Advising is also a platform to handle student issues that may arise during the semester. This can include, but is not limited to, learning how to handle override and exemptions, getting information on graduation status, determining qualifications for oral assessments and learning how to handle rescindment of 'Required to Withdraw' status. In an attempt to handle these minor issues, static forms of information that lend solutions were created in an attempt to curb some of the load faced by advisors. These range from paper-based handouts to bulletin boards strategically placed in department
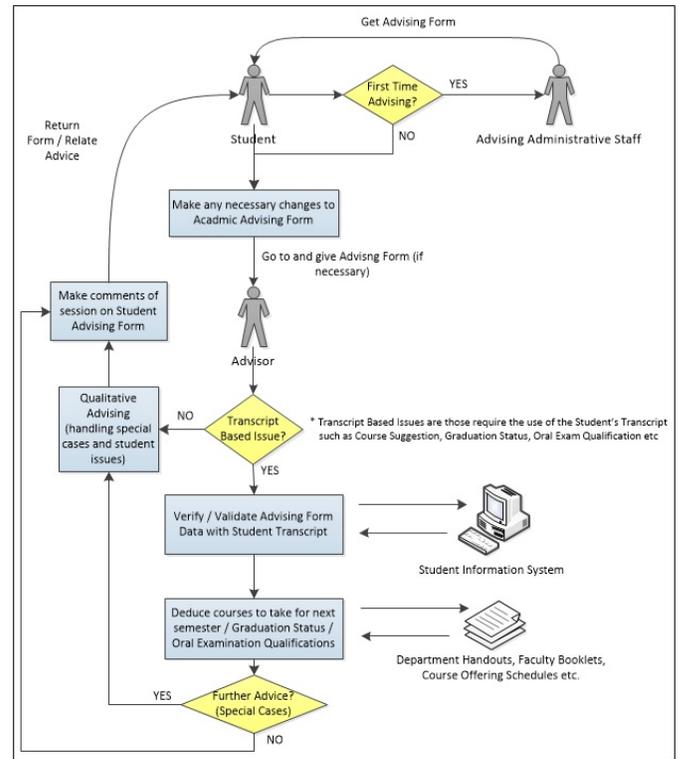

Fig. 1: The Traditional Advisory Process at FST

offices relaying information on programmes as well as steps to request overrides / exemptions and rescindment of RTW status. Attempts to broadcast such information via social media and departmental websites were also being made.

C. Issues faced in the traditional advising process

Although the process described in section I(B) may seem straightforward, further analysis show several issues that can be termed inefficient or problematic for the overall advising process.

Quality academic advising requires dedicated personnel to be available to handle the task. Financial constraints often make it unfeasible to hire staff solely for advising and hence existing staff are usually assigned this extra task, making their overall duties labour intensive. This is the case in FST as both staff and students are assigned advisory roles within their jurisdiction. While this can cut departmental costs, it raises availability issues as advising times may sometimes clash with advisors primary responsibilities [10] such as teaching and attending classes, thus forcing them to be unavailable for advising.

The quality of academic advising is also affected by the length of time that students are able to meet with advisors. Due to the high number of students per advisor, advisors tend to spend inadequate time with students or rush the advising process so that they can facilitate the load of students faced during the allotted advising sessions [1], [3]. This is often seen during the registration periods where there is often a backlog of students at advisor's offices.

On further analysis of student's issues, it was found that the majority of the advisor's time was spent answering recurrent questions [1], [10] pertaining to handling basic issues such as overrides as well as solving trivial course scheduling issues [2]. Also, when advisors try to spend sufficient time handling special cases, it often results in students having long waiting periods for advising, possibly even getting turned away and having to come again at another date. This can be quite unappealing to students, who may then resort to taking hearsay advice from peers which often leads to students making poor academic choices and then having to meet with an advisor to fix possibly more complicated student issues [10].

Even with tolerable advising loads there still exist many issues that lend to the inefficiency of the advising process at UWI. For example students are not assigned to any particular advisor and hence the possibility exists that they can interact with different advisors at different instances of their academic life. Such advisors would not have full knowledge of the student's background and hence would only be able to give advice based on the information provided to them at the point of advising. While the student should possess their paper-based profile which should have the student's academic history, the reality is that these papers are often lost or forgotten at the time of advising, forcing advisors to work with whatever student information is readily available. This can often result in students being led down 'blurred' academic paths as their past is not fully known. While this can possibly be resolved by the advisor simply asking the student to expand on his advising history, the process can be quite time consuming as well as inaccurate as the student might forget to relate critical information that can determine the advice given at the present meeting.

Improper representation of information can also cause unnecessary hiccups in the advising process. Advisors can be forced to work with multiple documents at a time, making the process more tedious than necessary when having to switch between them [2]. Also hand filled forms such as the one used in UWI's advisory process suffer a high possibility of having incorrect information due to human error by student or even past advisors. As a result, the advisor would have to validate form information against the student data within the SIS, which can usurp useful minutes from the advising session. In light of such redundancy, some institutions have chosen to eliminate the use of paper-based forms and simply work off the online student transcript generated from the SIS. The problem then is that such systems lack the ability to provide decision making capabilities based on student data and hence the advisor would still need to thoroughly analyse the student information before making a reasonable conclusion.

Administrative issues are also a factor in determining the quality of advising received by students. Advisors must be well equipped with knowledge on degree requirements, study plans and other rules pertaining to their advising scope to ensure that valid advice be given. In many cases, programmes are often under review and as such advisors are not always up-to-date with the changes made due to lack of dissemination of information from higher administration. Such shortcomings can also result in inaccurate advice being dealt to students causing them dissatisfaction and frustration, since it can possibly lead to delayed graduation [1].

These aforementioned issues, along with the issue of students not being able to attend advising sessions due to geographical constraints have all forced FST administration to explore ways to enhance the process altogether.

D. Academic Advising and Computer Science: Integration and Possible benefits

With the perpetual evolution of computer technology today, it is clear that at some point, institutions would seek to somehow computerize their advising process in an attempt to solve its underlying issues. Academic advising programmes should make full use of all existing modern technology, if necessary, to deliver the advising process.However, technology as a means of offering advisory services can be viewed by some to be cold and impersonal. It is therefore encouraged that technology be harnessed not to fully replace, but to improve the efficiency of the overall advising process, still allowing students to physically meet a human advisor if necessary.

As a result, technology can enhance the academic advising experience by assisting in the making of better informed decisions as well as providing improved services by migrating repetitive tasks on software. This would allow any student-advisor time to be dedicated to helping a student select the most appropriate path or handling any non-academic issues that may have an impact on the student's performance [5]. Such semi-automated advising would also significantly reduce the time for student-advisor interaction since students would only meet human advisors if their needs were not satisfied by the automated advisor. Furthermore, such systems would reduce the workload of staff that had to previously take on the extra job of advising, allowing them to focus on their primary areas of work, and by extension alleviate the issue of having too few advisors within the institution.

Such systems would typically eradicate the need for multiple hard-copy documents as all information could be available via a single interface, making the analysis of student data, if required, possibly easier than switching between physical documents. In addition, automation would help to remove inconsistencies in student information, especially if some sort of student profile is maintained, so that a history of all academic records and advising comments are available to the current advisor. The information would also be 'perpetually' available for the student, who can easily view his information when needed, as opposed to taking notes at advisory meetings or forgetting suggestions that was orally offered to him.

Finally, although providing a remote alternative to students who are unable to physically meet with advisors, an automated advising solution would seek, not to replace the human advisor altogether, but to alleviate his workload and cognitive stress [8]. This would be achieved by handling all student data and making optimized deductions so that the human advisor can focus on what he can do best, which is taking care of qualitative issues that the student may possess; thus improving the quality of academic advising. With such possible improvements it is clear that institutions would opt to transition to some form of an automated system, in an attempt to reap some of the many benefits that technology can provide.

E. Solution: AdviseMe : Student Advising Services

On scrutiny of the issues outlined in FST's academic advising programme, a computerized solution coined 'AdviseMe', an intelligent, web-based application for academic advising was proposed. This came as a result of the desire to raise the quality of the faculty's advising, eliminating inefficiencies where possible as well as migrating from a paper-based system. The solution is designed to complement the current process and hence is expected to work alongside the traditional human advisory system. It serves as an effort to enhance the efficiency, integrity and transparency of any tertiary-based advising system, similar to that of FST.

The system's core features, with the help of its rule-based inference engine, utilizes a student's transcript information and maps it against a set of configurable rules pertaining to programme information and university regulations. Results are then generated , rendering course suggestions for the upcoming semester, as well as information about the student's graduation status and eligibility for qualifications for oral assessments. It also provides students with reference material that provide answers to common issues and questions such as how to request overrides and apply for rescindment of RTW status, among others.

In the event that the advice generated for the student is insufficient, of that he has a special case that requires human attention, the system allows the student to remotely interact with an advisor associated with the student's course of study via email. At this point, it is left to the discretion of the student and human advisor, as to whether the issue can be handled remotely, or if a face to face appointment needs to be made.

AdviseMe also offers human advisor support, allowing them to see via a single interface, all previously mentioned student information as well the ability to place comments on student profiles which can then be used by the student as well as other advisors for future reference. This removes the need for a paper-based form to be maintained by the student and ensures that future advisors have a clear history of all student information before lending necessary advice. Advisors also have access to all reference manuals, eliminating the need for managing multiple documents and thus relieving some of the drudgery associated with the advising process.

An easy to use administration interface is included whereby from a web browser, administrators can manage student and course information, set customized prerequisite and exemption rules for courses, manage university regulations and maintain the overall health of the system, among other features. It also promotes the easy dissemination of rule changes to advisors by sending broadcasts when changes are made to rules within the system. This ensures that advisors are always up-to-date with the latest versions of rules and system settings as they are made in real time. All of these features, backed by an appropriately secure, efficient, scalable system with a simple, user-friendly interface create a compact but effective suite of services designed to enhance the overall advising process.

II. LITERATURE REVIEW

A. Description of Past Solutions

Surveying existing literature, we see that many institutions have implemented computerized solutions in order to enhance their overall advising experience. We also note that most institutions tend to write their own system, not only so that the solution is tailored to suit their direct needs, but also to eliminate the cost of licensing multiple copies of commercial software such as expensive expert system shells [11]. From our research, we see that solutions can be classified based on the level of automation they apply to the overall advising process. We define these systems as Basic Computerized Systems (BCS), Intelligent Interactive Automated Systems (IIAS), Advanced Automated Systems (AAS) and Intelligent Advanced Automated Systems (IAAS).

1) Basic Computerized Systems (BCS): Systems which either facilitate simple remote communication for the advisory process or those which migrate from the paper-based approach, but simply represent data in a computerized form or perform simple calculations are termed Basic Computerized Systems. In such systems, human advisors are still required to analyse information before any advice can be generated.

Reference [12] gives an example of a BCS as their institution uses an online 'Virtual Classroom' where web technologies are used to foster student-advisor communication. Such a system simply facilitates conventional advising, without the need for a fixed geographical location. A more technologically inclined BCS is presented by reference [2]. Created using VBA scripts and Microsoft Excel, the system automates some repetitive tasks in the advising process by performing functions such as GPA calculation. It should be noted however, that system operation requires two excel documents to be provided by the department; the first being a four-year schedule of the study programme and the other being a translation of the student transcript, since the system is not integrated in any way with the SIS. A web-based tool coined "The Online Advisor" however, utilizes existing data within their SIS (inclusive of prerequisite rules and graduation requirements) in order to produce an organized, colour coded representation of all advising-relevant information, centralized in a single display [5]. This is then used to complement the advising process, making it easy for advisors to create academic schedules by semester or year. It was designed to eliminate the use of multiple documents in the advising process by consolidating all information at a single interface.

2) Intelligent Interactive Automated Systems (IIAS): While BCS introduces technology into the advising process and also alleviates some of the drudgery associated with handling paper-based documents and making manual calculations, we see that introduction of higher processing capabilities and expert system technology can significantly reduce the human advisor's responsibility and student load in the overall process by directly handling advisory issues such as course suggestions. Intelligent Interactive Automated Systems seek to use such technology to emulate a real life student-advisor conversation, in order to gather sufficient data to generate substantial advice for students.

"A WWW Delivered Advising System" using the Exsys CORVID Professional Expert System Shell seeks to deliver a

'perpetually available' academic advisor specifically designed to handle cases of students who are unable to physically meet with a human advisor [11]. A similar system was put forward by reference [10] using Java Expert System Shell (JESS) and XML.

The drawback of both systems however is that students are required to supply solutions with a significant amount of information for results to be generated. This can sometimes result in the possibility of students entering inaccurate information and hence getting inaccurate advice, or students becoming disinclined to use such a system as they would prefer a system less demanding of them.

3) Advanced Automated Systems (AAS): In an attempt to make solutions more 'student friendly', we sought to turn attention to systems that produced similar results to IIAS with less student interaction. A subset of these was termed Advanced Automated Systems, which utilized prescribed algorithms and computational power to generate advice based on existing data.

Reference [4] discuss a system which uses database queries on the information stored within each student's transcript in the SIS in order to give students advice as to what courses they should take in the next semester as well as give their graduation status. Another system surrounding PHP, MySQL and Email technology is presented by reference [7] which again uses database queries to group all related student and course information for the purposes of generating a list of suggested courses to be taken in the next semester. A similar system is also shown in reference [13] whereby developers used Wxpython alongside an access database to deploy a desktop solution to facilitate postgraduate students.

4) Intelligent Advanced Automated Systems (IAAS): AAS can be quite effective and produce satisfactory results. However, the database queries used to generate such results can be quite complex and resource intensive. Furthermore, the rules within advising solutions that govern programme structure and university regulations are frequently reviewed, which can then require the need to modify the SQL queries which act as rules governing the system's functionality. A more practical approach, promoting change and easy maintenance would be using expert system technology to manage the rules that govern the advising system. Such systems that provide this functionality, similar to that of AAS are coined as Intelligent Advanced Automated Systems and research showed that this was the most favoured approach when creating advising solutions for institutions.

Such systems were observed to use various reasoning strategies to achieve their appropriate results. One for example showed how Case-Based Reasoning (CBR) was used to develop a system that recommended a suitable major to students based on comparing their student information against similar historical cases [9]. The system was proven to be quite effective when advising students who were reading for general degrees, provided there were sufficient historical cases within its knowledge base (KB). Rule-Based Reasoning (RBR) systems were also designed whereby developers used Forward and Backward Chaining procedures in order to generate appropriate advice. Such capabilities as well as a 'cognitive and emotional filter' were used in a solution proposed by

reference [1] in order to provide course advising to students. "IS-Advisor" also followed a rule-based approach alongside its Object Oriented Database [6]. Other systems such as the "Course Advisory Expert System" went a step further by providing both reasoning capabilities in order to facilitate an even higher quality of advising when prescribing course recommendations [8].

With the emergence of ontologies and semantic web technology, we saw that developers harnessed ontology driven methodologies to tackle the dynamic and complex nature of student academic planning and scheduling by creating "E-Advisor", a multi-agent intelligent advising system [3]. Designed for the Master of Science in Information Systems in Athabasca University in Canada, it allowed students the ability to add preferences of specialization to their profile and then recommend courses based on these preferences. Its multi-agent nature also made system maintenance easy, as it allowed the use of other agents while updating others making it a highly available system, and possibly one of the better intelligent web-based advising systems in the world today.

B. AdviseMe as opposed to previously implemented solutions

After much scrutiny of the problem domain as well as the solutions discussed in section II(A), it was clear that a system with an IAAS architecture be the optimal solution for enhancing the level of advising currently experienced in FST, as it would seek not only to minimize the need for student input and generate satisfactory results, but also significantly reduce human advisor workload, among other benefits. This places "E-Advisor" as the top candidate for consideration. However, several issues arose which lent to the ultimate dismissal of such a solution.

While E-Advisor has proved its successful application at Athabasca University, it is noteworthy that the system was tailored to advise students in a single MSc Programme [3] and not a range of programmes that would be required of a system serving FST. Also, the system focuses on course scheduling, but no mention of the system handling other important features required by FST was made in the literature. Determining oral assessment qualifications and graduation status are but only some of the important student issues that human advisors often have to repeatedly solve.

As a result, AdviseMe's design was proposed since not only does it facilitate aforementioned services as well as tracking of student history and other useful features, but also facilitates both undergraduate and postgraduate programmes that can be managed on a department, faculty or even campus based level. Furthermore, AdviseMe's solution requires less staff involvement as it only needs a minimum of 1 administrator to configure the environment while E-Advisor's functionality is dependent on every instructor in the department for the proper working of the system. While this extended staff involvement poses benefits in the context of Athabasca University, it fails to alleviate the staffing issues faced in FST. For these reasons, as well as the fact that no single previously mentioned solution efficiently solves the issues outlined in section I(C), the design and implementation of AdviseMe was born.
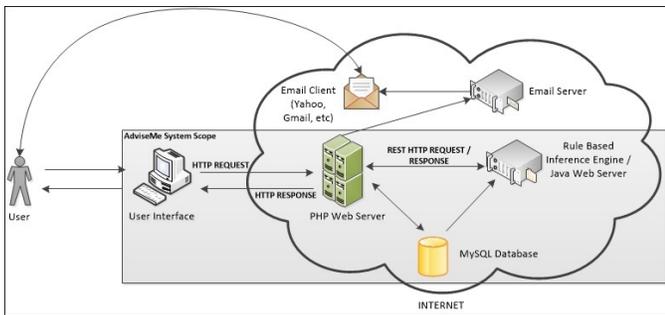
Fig. 2: Flow of data between modules within AdviseMe

Fig. 3: Overview of the PHP Web Server (PWS) Architectural Design

III. System Design

A. Overview

AdviseMe typically has 3 main user groups (students, advisors and administrators) all of which interact with the system via a single browser rendered user interface.

When requests are made, they are passed to the PHP Web Server (PWS) which acts as the mediator of all data flow within the system. The system provides both 'intelligent' services (eg. Course Advising) as well as 'non-intelligent' services such as system management and email communication. For all intelligent services, the PWS communicates with the RESTful Java Web Server (JWS), which handles all intelligent processing via its use of ontologies and its rule based inference engine.

The PWS also sends requests to the Email Server for disseminating messages to users based on the services accessed. For example, when an administrator updates a rule in the system, the PWS would send a request to the Email Server in order to broadcast a message to advisors, notifying them of changes made.

All other facilities provided by the PWS are fuelled by information retrieved from or sent to the MySQL database via the UI. The JWS also retrieves information from the MySQL database before processing data and sending results to the PWS. Ultimately, the PWS would manage all processing within the system and render the output in a sleek, intuitive form via the UI.

B. PHP Web Server (PWS)

The PWS was designed and implemented using the CodeIgniter Framework and facilitates the server side implementation of the web application, as well as the interface for communication with the Email Server (Google's Gmail SMTP Server), MySQL database and RESTful JWS (by means of the Curl URL Library). CodeIgniter's extensive documentation and 'Model-View-Controller' architectural style promoted quick and easy implementation of the PHP based application server. Controllers accept data from the models and pass them to views which render the information in an intuitive format for end users. In AdviseMe, the models handle two types of data requests, the first being calls to the MySQL database using CodeIgniter's Database Library and the other being requests being sent to the RESTful JWS. Information from the JWS is retrieved via Uniform Resource Locators (URLs) and the use of the Curl URL Library, for the consumption of REST
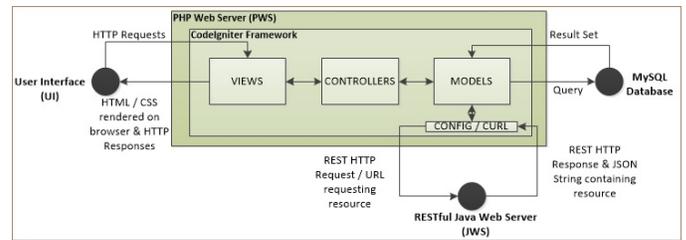
services. An overview of the PWS architecture is shown in Figure 3.

C. RESTful Java Web Server (JWS) Design

Although not directly accessed by users, the JWS is essentially the driving force behind AdviseMe's core functionality. Using Apache Jena alongside ConnectorJ to create an ontology-based reasoning environment, it proves to be an ideal reasoning engine when seeking to determine non-trivial inferences on a student's academic record. These inferences are then exposed as JSON objects via RESTful web services, made possible by using tools such as Jersey and JSON-Simple.

Jena's API provides a wide collection of classes and interfaces for the management of 'OWL-Based' technologies. OWL, an acronym for 'Web Ontology Language', is typically an extension of the Resource Definition Framework Schema (RDFS) which in turn is an extension of the Resource Definition Framework (RDF). RDF forms the foundation of how resource information should be structured, with RDFS and OWL enhancing the ways in which resources are described.

With RDF being a suitable standard for structuring data and OWL having an extensive vocabulary that can be easily interpreted by machines, Jena's Ontology API was used to create a simple ontology termed 'AdviseMeOnt' to be used within our system context. Comprising of a set of resources / classes immediately surrounding the academic advising environment and their appropriate properties, it was used to model all information within the student's transcript in order to produce meaningful inferences to support the advising process. A listing of some of the concepts within 'AdviseMeOnt' is given in Table 1.

Student Profiles were then created by extracting information from the MySQL database to create and populate an ontology model using the Jena Ontology API, and then exporting the information to a RDF file.

In addition to creating RDF-Based student information, the JWS was also designed to extract information from the database and create a set of user defined rules to be applied to the RDF files. These custom rules included prerequisite and exemption rules, as well as rules containing other user defined variables. The JWS also can generate a set of fixed, system-defined rules by calling the appropriate functions within the server.

Once RDF-Based student profiles and the set of executable rules are existent, the JWS can then generate non-trivial

TABLE I: Listing of some concepts and properties within 'AdviseMeOnt'

advMe:COURSE
advMe:COURSECODE
advMe:COURSENAME
advMe:SEMESTEROFFERED
advMe:MARKOBTAINED
advMe:NUMBERCREDITS
advMe:PROGRAMME
advMe:PROGRAMMENAME
advMe:LEVELONECREDITS
advMe:CORECREDITS
advMe:ELECTIVECREDITS
advMe:FOUNDATIONCREDITS
advMe:HASCORECOURSE
advMe:HASELECTIVECOURSE
advMe:HASFOUNDATIONCOURSE etc...
advMe:STUDENT
advMe:STUDENTID
advMe:STUDENTNAME
advMe:STUDENTTYPE
advMe:ACADEMICSTANDING
advMe:FUNDINGSTATUS
advMe:CANGRADUATE
advMe:COMPLETEDCOURSE
advMe:ALLOWEDCOURSE
advMe:LOCKEDCOURSE
advMe:FAILEDCOURSE
advMe:POSSIBLEORALCOURSE etc...

Fig. 4: Overall Structure of Inference Machinery used within Jena

Fig. 5: Overview of the Java Web Server (JWS)Architectural Design

backward chaining capabilities, ideal for handling the ruleset within the JWS. The student profile and ruleset are bound to this generic reasoner and after processing, the reasoner outputs an InfGraph object. This is typically a new model of the student profile containing previously asserted as well as newly inferred information. The information is then saved to storage as an RDF file which is later parsed using the Jena Ontology API as well as ARQ (a query engine supported by Jena providing RDF querying capabilities) for extracting results.

In order to expose inference results to users in a presentable format such as via a structured website, information extracted from the InfGraph model was transformed into JSON format before being made available to external users via RESTful web services. This was done using JSON-Simple and Jersey Java packages respectively which allowed information to be passed to the PWS via an HTTP response.

As a result, the JWS communicates seamlessly with the PWS in order to make useful advising information for rendering via the UI. Administrators are also allowed to access the PWS via the UI in order to perform 'remote' tasks on the JWS such as re-creating a student's RDF-Based Profile and refreshing the user-defined rule base. Figure 5 shows an overview of architecture of the JWS.

IV. METHODOLOGY

A. Pre-Implementation Phase

Preceding system implementation, research was conducted by acquiring information from both students and advisors currently at UWI. Student's thoughts on a computerized advising system were collected via an anonymous questionnaire. Participants were selected from first year undergraduates straight up to postgraduate students in order to get a wide range of responses from those who would have been new, as well as quite accustomed to the traditional advisory process. Results from the questionnaire showed that the majority of students were dissatisfied with the traditional advising process, with 90% of them visiting advisors for the handling of general matters such as course scheduling, determination of graduation status and the handling of trivial student issues. Furthermore, students stated that they would expect such a system to offer 24/7 accessibility, course advising capabilities and support for handling other student issues such as holds and academic standing issues. They also suggested that the system possess

inferences within the student profile using the Jena Inference Subsystem (JIS). This module, accessed via the Jena API, is designed to allow a wide range of inference engines to be integrated with Jena projects for the derivation of additional RDF data, by making inferences on existing RDF assertions. Figure 4 illustrates the overall structure of the inference machinery used within Jena and by extension the JWS.

The JWS accesses the JIS using the ModelFactory class to associate the RDF-Based student profiles with the appropriate reasoner. Jena provides a Generic Rule Reasoner which supports the user of user-defined rules as well as forward and
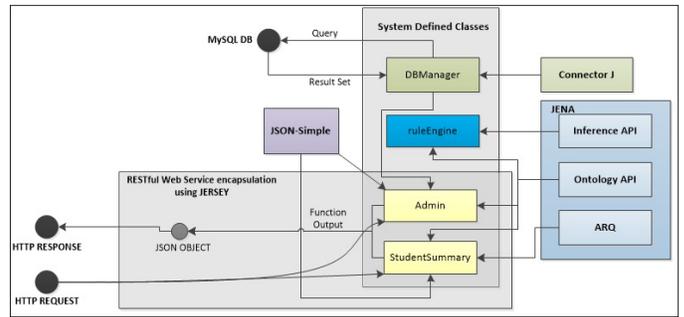
some level of intelligence so to give possibly better advice than existing advisors who are sometimes unaware of critical changes in the process.

Representing the advisory body was the Deputy Dean and Head of the Advisory unit at FST. In conclusion of her interview she acknowledged that an intelligent computerized system would have immensely benefited advisors by reducing their advisory load, promoting more dedication to special cases and enhancing the overall workflow of the advisory process. She verified the hypothesis that such a system would be able to handle a significant amount of student cases without the need for a human advisor and also highlighted that the ability of students to remotely contact human advisors if necessary could also help to streamline the other aspects of the advisory process. As a result of her confidence in the system, the design and implementation quickly came under way as it posed some value to UWI and more specifically FST.

B. Implementation Phase

As stated in section III, AdviseMe comprises of two distinct modules, the RESTful JWS which provides all intelligent processing with respect to generating academic advice and the PWS which converts this advice into a presentable format to be rendered via a web browser, among other functions. Section IV(B1) gives a summary of some of the services offered by the JWS while Section IV(B2) summarizes the functionality of the overall system as controlled by the PWS.

1) RESTful JWS Implementation: As stated before, the JWS provides RESTful services accessible via HTTP requests. These services comprise of the following:

- Creating / Refreshing the 'AdviseMeOnt' Ontology

- Creating / Refreshing the rule base (comprising of user and system defined rules)

- Creating / Refreshing all student profiles (based on transcript information in SQL database)

- Creating / Refreshing a single student profile(based on transcript information in SQL database)

- Generating advisory information for a given student (based on his student profile)

All of these services return a JSON string to the requesting agent after processing. With exception to the last service, all requests return a success value of "TRUE" if the request was successfully processed or "FALSE" otherwise (see Figure 6). In the case of generating the advisory information however, the request,once successful, returns a JSON object containing the following information:

- Student Name, ID and Student Type (Full Time / Part Time)

- Academic Standing and Funding Status (Whether or not the student can be covered for sponsorship payments)

- Number of years in current programme of study

- Number of credits required for each course segment of programme (Eg. Core, Elective, etc.)

Fig. 6: Sample of service output when genaral request to JWS is made

Fig. 7: Sample of service output when request to retrieve student's advising information is made (Part A)

- Number of credits completed for each course segment of programme (Eg. Core, Elective, etc.)

- List of courses that can be taken by student (based on priority in programme of study)

- Whether or not a student can graduate from current programme of study

- Number of Oral Assessments that the student currently has remaining

- List of all courses which the student can apply for an oral assessment

- Progress in all programmes similar to student's current programme of study

An illustration of this is shown in figures 7 and 8. This information is then consumed by the PWS which then renders it to the user as required.

2) PWS Implementation : The PWS, responsible for controlling all functionality within AdviseMe, provides a wide range of functions and serves three types of users: students, advisors and administrators. A list of the core of AdviseMe is given in the table 2.

Fig. 8: Sample of service output when request to retrieve student's advising information is made (Part B)

Fig. 9: Sample of Academic Advising Page (Part 1)

Fig. 10: Sample of Academic Advising Page (Part 2)

TABLE II: Core Features of AdviseMe

Description	Stu	Adv	Adm
View student advising history giving all comments previously placed on student profile by advisors.	YES	-	-
View course advising, by generating a list of courses to be taken per semester as well as show progress in similar programmes in the event of a transfer being considered.	YES	-	-
Request to contact a human advisor in the event that advice given is not sufficient.	YES	-	-
View graduation status of student, showing progress in each of the programme sections.	YES	-	-
View oral examination information, giving the number of orals that a student can still pursue, the number of outstanding credits for completion of programme requirements and the list of courses for which a student can request an oral examination.	YES	-	-
Download a complete summary of all aforementioned information for a student in PDF format.	YES	YES	-
View Information on programme structure as well as common student issues such as requesting overrides and rescindment of RTW status.	YES	YES	-
Place a comment on a students record to be used for future reference by student and advisors.	-	YES	-
Management of all rules within the system inclusive of creation and modification of user defined rules such as prerequisite and exemption rules.	-	-	YES
Management of all courses, programmes, departments, faculties, students' transcripts and users within the context of the university and system.	-	-	YES
Management of the entities associated with the JWS (as mentioned in the previous section).	-	-	YES

As a student, possibly the most useful feature is the "Academic Advising" option, which allows him / her the ability to retrieve information generally sought in academic advising sessions. This initially includes basic student information, his / her academic standing and the class of degree he / she is currently in line for (Figure 9). The proceeding section then gives the maximum and minimum number of credits the student can take per semester as well as the number of credits remaining for each section of the programme (Figure 10). This is then followed by a list of courses (Figure 11) within the current programme of study that can be taken for programme advancement based on semester and also ordered by two levels of priority; the first being how much courses they are prerequisites for and the second being classification by course type (eg. core courses are given higher priority than elective courses). The student is also notified in what semester's possible future courses will be offered so to promote future course planning. Finally, in the last section of the page, the student is able to see his / her progress in similar programmes offered (Figure 12), allowing them to know their stance in other programmes in the event they decide to switch to another programme.

All of these aforementioned sections, in addition to information of graduation status and oral exam assessments give satisfactory advice that answer most common questions faced in academic advising. If however, the student believes that the advice received is insufficient, he / she can then opt to contact a human advisor via email by clicking the "contact advisor" button shown in Figure 9. This then redirects to a new page which allows the student to send an email to the advisor via adviseMe's interface.

Reccomended Courses for Semester One			
Course	Number of Credits	Course Type	Prerequisite For
COMP2000 - Data Structures	4	CORE	COMP3000 - Design and Analysis of Algorithms (offered in semesters ONE and TWO)
			COMP3300 - Programming Languages I (offered in semester TWO)
			COMP3850 - Intelligent Systems (offered in semester ONE)
			COMP3250 - Software Engineering (offered in semester TWO)
COMP2200 - Computer Architecture	4	CORE	COMP3100 - Operating Systems (offered in semester ONE)
COMP2700 - Database Management Systems I	4	CORE	COMP3700 - Database Management Systems II (offered in semester TWO)
COMP2600 - Theory of Computing I	4	ELECTIVE	
FOUN1301 - Law, Governance, Economy and Society	3	FOUNDATION	

Fig. 11: Sample of Academic Advising Page (Part 3)

Thinking of switching to a different programme?

Below shows your current progress in similar programmes*. If you think you might desire to change programmes please contact an advisor before taking any further step.

B.Sc. Computer Science
91%

B.Sc. General (Major Computer Science)
100%

B.Sc. General (Double Major Computer Science)
85%

B.Sc. General (Minor in Computer Science)
100%

* Percentages shown refelect a figure based on the requirements to attain each qualification but does not necessarily reflect graduation status. For example having a percentage of 100% for a Minor only suggests that you have completed the requirement of the minor and not a BSC. General Degree.
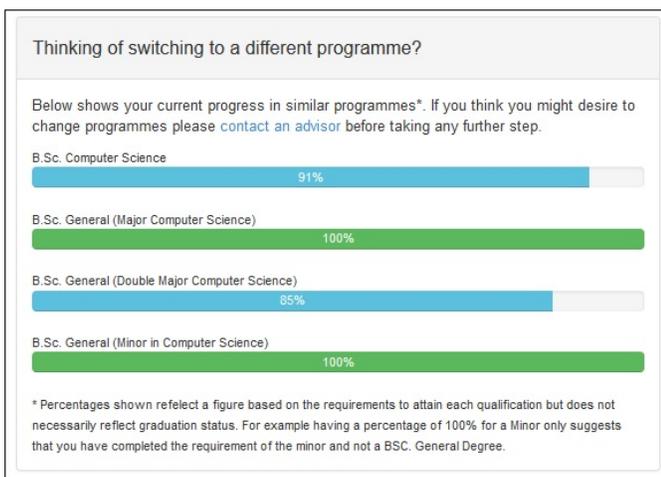
Fig. 12: Sample of Academic Advising Page (Part 4)

C. Post Implementation Phase

To ensure overall acceptance of the system, students of varying aptitudes within the university were allowed to assess the system by either volunteering their own transcript or using a model of a sample transcript. From the 50 percent of students who produced their transcripts for testing, all received accurate advice in all aspects of the system with regard to student advising. In addition to its functional success, all interviewed students found the system to be exceptionally usable and intuitive, commenting that its simplicity and use of visual charts to illustrate student progress made the system very appealing. They also in particular, appreciated the record of

advisory history appended to student profiles and the ability to contact a human advisor if necessary. 20 percent of the sample set however, still held strong to the fact that while they would receive sufficient information from the system, they would still opt to speak to an advisor if required. They did appreciate however that the process of meeting an advisor could now be less time consuming as they can now schedule and appointment beforehand, eliminating the need to wait long periods in the advisory offices.

V. LIMITATIONS OF THE SYSTEM

The system is designed to give accurate advice only to those pursuing "special degrees" or programmes which follow a clear cut path of courses. While this limits the system to serve only a portion of students within FST and other similar institutions, it would still result in the overall reduction of advising load for human advisors, making their duties less demanding.

Another limitation is that while the student can contact a human advisor via email and successfully have their issues solved, the advisor's response time is subjective to when he / she chooses to respond to the student as opposed to face to face conversations where solutions are immediately discussed. This can be unsatisfactory to students, especially if advisors take too long to respond to student concerns.

The system is also in the prototype phase and hence requires administrators to input the student transcripts for system processing via the user interface. While UI design makes the uploading process possible in minimal time, it still results in an unnecessary action by a human entity. However, this timeframe is relatively short, which can be deemed quite acceptable as the number of hours saved by both students and advisors using the system would clearly offset the setup period.

VI. FUTURE WORK

With respect to the limitations discussed above, further enhancements are proposed to investigate ways in which the system would be able to sufficiently accommodate students pursuing general degrees. Also in terms of data integration, measures to have the system automatically process information from the Banner Student Information System will be explored, in an attempt to reduce administration involvement further.

With respect to added system functionality, measures would be put in place to offer a higher level of student advisor communication, possibly including live chats with advisors via Instant Messaging or Video Conferencing. Also the idea of a blog whereby a community handling frequently asked questions or common issues would be explored. This would possibly increase the level of acceptance of the system by students and solve the issue of delayed responses that can be incurred while waiting for advisors to attend to emails.

Finally, with respect to future long term enhancements of AdviseMe, a programme planning module can be implemented, using collected data from student advising to generate statistics for use by departmental administration to assist in the allocation of teaching resources for upcoming semesters. This would not only add to the value of the system, but increase its scope of alleviating manual processing issues within tertiary education institutions.

VII. CONCLUSION

Utilizing AdviseMe in order to facilitate academic advising without the possible involvement of human advisors will definitely enhance the efficiency, integrity and transparency of any tertiary based advising process similar to that of FST. The successful prototype discussed in this paper highlights its feasibility and practicality in the context of UWI and their degree programmes. Serving a significant portion of the student body, it would provide sufficient advisory services to the majority of its users thus reducing the student load faced by human advisors at advisory offices. For those students whose issues go beyond the scope of its assistance, it allows an avenue of communication to human advisors that was previously non-existent, by means of email technology. Not only does this create a flexible way of seeking advice, but it can also improve the quality of the qualitative advice received from advisors, since more time can now be dedicated to handling these special cases.

Its current architecture supports the use of a PHP based web application interacting with an intelligent, RESTful Java Web Server, in order to provide expert advice on course scheduling issues via any device that supports internet browsing. However, the fact that the intelligence processing is exposed via web services indicates that the system can be adapted to suit any future front end deployment, provided that access to the JWS API is given. This promotes acceptance of AdviseMe, outside of UWI, as solutions can be tailored to any university context, once their programme structure is the similar to that of UWI. Returning to the context of FST, we see that a significant number of students showed possible acceptance of the system in the infant stages of conceptualization. This figure sought to increase nearing the end of implementation however as all students interviewed for testing made positive remarks about the system's functionality, usability and applicability to the advising context of FST; with the majority of them stating that they would use such a system. Furthermore, when demonstrated to the Head of the Advisory Unit of FST, she was highly pleased with the outcome, to the point of suggesting possible practical implementation within the faculty in the near future.

With such positive feedback, and the fact that the resulting product captured all the requirements that was proposed of such a system, it is clear that the implementation of AdviseMe was indeed a success, lending its services to support new and existing advising processes in order to enhance the overall quality of academic advising received by students today.

REFERENCES

[1] E. Nwelih and S. Chiemeke, "Framework for a web-based spatial decision supportsystem for academic advising," *African Journal of Computing and ICT*, vol. 5, no. 4, pp. 121–126, 2012.

[2] M. T. Al-Nory, "Simple decision support tool for university academic advising," in *Information Technology in Medicine and Education (ITME), 2012 International Symposium on*, vol. 1. IEEE, 2012, pp. 53–57.

[3] F. Lin, S. Leung, D. Wen, F. Zhang, and M. Kinshuk, "e-advisor: A multi-agent system for academic advising," *International Transactions on Systems Science and Applications*, vol. 4, no. 2, pp. 89–98, 2008.

[4] F. Albalooshi and S. Shatnawi, "Online academic advising support," in *Technological Developments in Networking, Education and Automation*. Springer, 2010, pp. 25–29.

[5] T. Feghali, I. Zbib, and S. Hallal, "A web-based decision support tool for academic advising," *Journal of Educational Technology & Society*, vol. 14, no. 1, pp. 82–94, 2011.

[6] M. A. Al Ahmar, "A prototype student advising expert system supported with an object-oriented database," *International Journal of Advanced Computer Science and Application*, vol. 1, no. 3, pp. 100–105, 2011.

[7] M. Beheshti, T. Trang, K. Kowalski, and J. Han, "Student advising system," in *World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education*, vol. 2006, no. 1, 2006, pp. 2727–2732.

[8] O. Daramola, O. Emebo, I. Afolabi, and C. Ayo, "Implementation of an intelligent course advisory expert system," *International Journal of Advanced Research in Artificial Intelligence*, vol. 3, no. 5, pp. 6–12, 2014.

[9] L. Mostafa, G. Oately, N. Khalifa, and W. Rabie, "A case based reasoning system for academic advising in egyptian educational institutions."

[10] A. N. Nambiar and A. K. Dutta, "Expert system for student advising using jess," in *Educational and Information Technology (ICEIT), 2010 International Conference on*, vol. 1. IEEE, 2010, pp. V1–312–V1–315.

[11] K. Kowalski, J. Goetz, and M. Alam, "Intelligent on-line advising with expert system shell," in *World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education*, vol. 2006, no. 1, 2006, pp. 687–694.

[12] M. J. Martínez-Argüelles, E. Ruiz-Dotras, and E. Rimbau-Gilabert, "The academic advising system in a virtual university," in *Technology Enhanced Learning. Quality of Teaching and Educational Reform*. Springer, 2010, pp. 345–350.

[13] A. Al-Ghamdi, S. Al-Ghuribi, A. Fadel, and F. AL-Ruhaili, "An expert system for advising postgraduate students," *International Journal of Computer Science and Information Technologies*, vol. 3, no. 3, 2012.