An Intelligent Natural Language Conversational System for Academic Advising

Edward M. Latorre-Navarro
Computer Science Department
University of Puerto Rico in Arecibo
Arecibo, Puerto Rico

John G. Harris
Electrical and Computer Engineering Department
University of Florida
Gainesville, Florida, U.S.A.

***Abstract*—Academic advisors assist students in academic, professional, social and personal matters. Successful advising increases student retention, improves graduation rates and helps students meet educational goals. This work presents an advising system that assists advisors in multiple tasks using natural language. This system features a conversational agent as the user interface, an academic advising knowledge base with a method to allow the users to contribute to it, an expert system for academic planning, and a web design structure for the implementation platform. The system is operational for several hundred students from a university department. The system performed well, obtaining close to 80%, on the traditional language processing measures of precision, recall, accuracy and F1 score. Assessment from the constituencies showed positive and assuring reviews. This work provides an assessment and technological solution to the academic advising field, i.e., the first-known advising multi-task conversational system with adaptive measures for improvement. The evaluation in a real-world scenario shows its viability, and initiated the development of a corpus for academic advising, valuable for the academic and language processing research communities.**

Keywords—Natural Language Processing; Dialog System; Conversational Agent; Academic Advising; Advising System; Engineering Education; E-learning; Human Computer Interaction

I. Introduction

Higher education institutions employ academic advisors to assist students in academic, professional, social and personal matters [1]. Today, academic advising is also a peer reviewed research area given the many important implications of a successful advising system such as student retention, graduation rates and student educational goals including academic engagement and performance, and career planning [1]-[2]. Therefore, it is essential for academic programs to offer students an effective advising experience, which requires advisors to innovate at the speed of their students, who each year tend to have higher expectations from their education institutions and a stronger synergy with digital technology. Thus we see the innovation trend in advising has been towards the use of communication technologies such as email, instant messaging, social networking, course-management systems, podcasts, mobile applications, online videos and blogs [3]–[5]. A quick survey of university websites shows that many have introduced the concept of eAdvising, that is, utilizing electronic means, usually web-based, to offer advising to students [6]–[8].

In 2008, Leonard detailed the profound effect of technology use by advisors and referenced the idea of an interactive advising expert system as a possible future trend, but few institutions had shown interest in developing such a system [3]. A fully automated system is a better solution for the innovation trend in advising and addresses the concern of advisor to student ratio, in an economically challenged environment [1]. Such a system lessens the burden of academic advisors from several mundane tasks and frees up more of their time for the deeper aspects of advising, such as career planning or managing extraordinary personal situations.

There are several research publications describing advising expert systems for helping students with straightforward repetitive tasks such as choosing majors, adhering to an appropriate curriculum sequence or accessing degree audits [9]–[13]. This work is inspired by the belief that, following current technological trends, new interactive advising systems should also include a natural language interface to allow students to communicate as freely and openly as with their actual advisors. Such an innovative system could be much more attractive to students as it would allow them to easily ask a wider range of questions than those in previous expert systems and obtain immediate responses to these, instead of waiting for peers or advisors to read and reply, as with current eAdvising methods.

The main objective of this work was to construct and deploy a real-world academic advising system that allows the students to communicate freely in natural language. The system was designed to serve students of the Department of Electrical and Computer Engineering in the University of Florida (ECE-UF). This task allows for training and testing of the algorithms developed in a well-defined, domain-specific, conversational question answering application, where there are no available corpora for machine training. This work also introduces a methodology to allow the users to manage the system scale up process.

To the authors' knowledge, two publications exist that present advising systems combined with natural language processing (NLP) techniques for communication [14]-[15]. The first system was limited to only yes/no type questions and a few phrases to manage state transitions [14]. The other system features an ontology-based information retrieval engine to guide students in searching for answers after entering keywords [15]. The system developed for this research work allows unrestricted natural language communication utilizing state of the art NLP techniques.

More information on related work and the fundamental requirements of this system are available in a previous publication [16]. That work introduces the system along with its dialog manager, justifies the design components and presents preliminary results. Section 2 briefly describes the academic advising task, the user base and the advising system developed. Section 3 describes the system dialog manager and task managing components. Section 4 explains the academic planning system, including the components for communication between the students and their advisors. Section 5 describes the field tests and the analysis of the results. Section 6 contains the conclusion and future work. Finally, the appendix includes the list of academic advising topics covered in the system, a selection of user dialog from the experiments and screenshots of the web interface.

II. THE ACADEMIC ADVISING SYSTEM

A. Academic Advising

Advising tasks are identified as prescriptive, providing expert advice, and developmental, where the advisor engages in a mutual learning process with the student, in order to help the student's problem solving, decision making and evaluations skills [17]. Other studies support the idea of educative advising, where advisors are the teachers of the philosophy of the curriculum and the principles of how students learn [18]. This work provides advisors with a tool to streamline many prescriptive tasks, allowing further developmental and educative tasks during their limited face-to-face time.

In ECE-UF, students visit their academic advisor mostly by request of the advisor. These meetings must occur at least once per academic semester to evaluate the student's current academic progress and enrollment for the following term. During these meetings, most topics involve queries with accessible answers, i.e., prescriptive advising. Many of these answers are obtained directly from facts or through logic analysis, and thus may be solved algorithmically. This system is designed for these tasks, but also includes answers to some developmental advising topics. See the appendixes for the list of topics and user dialog examples.

B. Albert, the Natural Language Academic Advising System

Albert, the natural language academic advising system, provides students with an academic advising service that reflects a human interaction experience through an online text application [16]. The system does not require student training or additional human resources from the academic departments. This system enhances the academic advising experience by offering students a service that is available at any time. Albert includes multiple advising services accessible from any device with web access. Albert respects the privacy of its users, and encourages the students to become independent and take responsibility for making decisions. As a courteous advisor employed by an academic department, the system output reflects an advisor who is polite, maintains a positive affective state with its users and simulates a personality trait intended to sympathize with the students of ECE-UF.

Albert includes knowledge about the academic programs and policies, answers to a wide range of academic frequently asked questions (FAQ), it offers recommendations for the development of a successful academic plan and referrals to other academic services. The target users are students from the ECE-UF two undergraduate degrees, Bachelor of Science in Electrical Engineering (BSEE) and Bachelor of Science in Computer Engineering – Hardware emphasis (BSCEE).

Albert contains the course scheduling information for all ECE-UF courses. A Python script reads the information from the UF Registrar's webpage and a cron job updates the knowledgebase (KB) daily. The information is stored indefinitely and is available for access when queried by date. This process assures the information is always up to date without dependency of human maintenance. All the other information in Albert is hand scripted in the KB of the system.

The main script of Albert is written in the Python programming language, version 2.7.5. This script controls all the functions of Albert and communication with each module including the web interface, the gateway interface for web communications, the user login routine, the dialog manager, the expert system for academic planning and the database. Fig. 1 shows a model of the Albert system. The dialog manager includes the natural language understanding and generation systems, and the task manager.

This website is hosted in a desktop computer at ECE-UF facilities and accessible via the Internet address http://advising.ece.ufl.edu. The computer has an Intel Pentium 4 processor, 1.8 GB of RAM and is running the operating system (OS) Red Hat Enterprise Linux 6.4.

The website contains scripts written in Python, PHP, JavaScript, HTML and CSS programming languages, in addition to Unix scripts to manage the daily tasks. Communication is through socket technology, which is widely used in web-based software. The website was designed for simplicity and speed, with a load time of approximately one second on contemporary versions of the popular web browsers. Users who access Albert using the Google Chrome web browser can send messages using speech recognition software. The website includes an open-source speech recognition API that with the required hardware can interpret speech data [19].

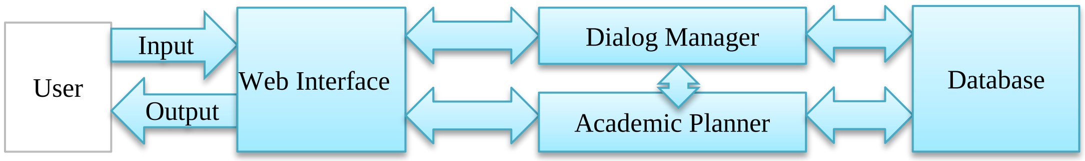
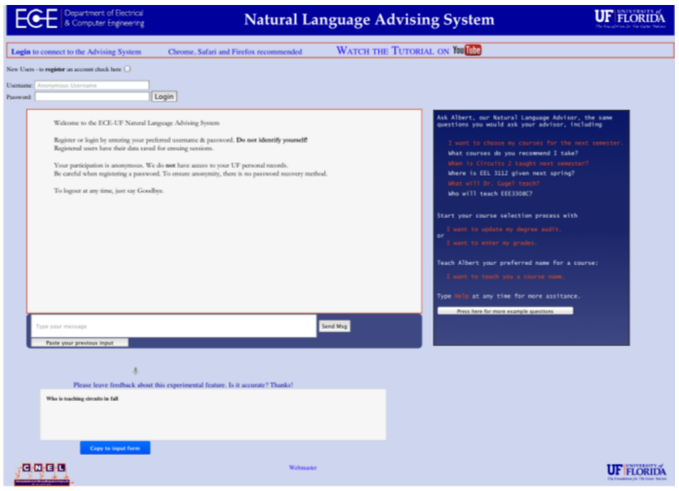


Fig. 1. A model of the main components of Albert


ECE Department of Electrical & Computer Engineering
Natural Language Advising System
UF UNIVERSITY of FLORIDA
Username:
Password:
Login
Send Msg
Paste your previous input
Press here for more example questions
Copy to input form
Webmaster
CNEL

Login to connect to the Advising System Chrome, Safari and Firefox recommended WATCH THE TUTORIAL ON YouTube

New Users - to register an account check here

Welcome to the ECE-UF Natural Language Advising System

Register or login by entering your preferred username & password. **Do not identify yourself!**
Registered users have their data saved for ensuing sessions.

Your participation is anonymous. We do **not** have access to your UF personal records.
Be careful when registering a password. To ensure anonymity, there is no password recovery method.

To logout at any time, just say Goodbye.

Type your message

Ask Albert, our Natural Language Advisor, the same questions you would ask your advisor, including

I want to choose my courses for the next semester.
What courses do you recommend I take?
When is Circuits 2 taught next semester?
Where is EEL 3112 given next spring?
What will Dr. Gugel teach?
Who will teach EEE3308C?

Start your course selection process with
I want to update my degree audit.
or
I want to enter my grades.

Teach Albert your preferred name for a course:
I want to teach you a course name.

Type Help at any time for more assitance.

Please leave feedback about this experimental feature. Is it accurate? Thanks!

Who is teaching circuits in fall

Fig. 2. A screenshot of the website for Albert in its initial state

Fig. 2 shows a screenshot of Albert in its initial state. The mid-upper left area shows the forms for logging into the system. Below these forms, the main window for communicating with Albert shows the instructions for connecting with the system. On the right of the main window, an independent web frame shows examples of FAQs the system contains. A screenshot showing this right-side frame is available in Appendix C. The website meets of the accessibility compliances of the Americans with Disabilities Act and the Workforce Rehabilitation Act of 1973.

To protect the information of the students, the system requires users to register an account using an anonymous username and password combination. This data is stored encrypted in the Albert web server using the standard secured hash algorithm SHA-512 – 64 bits. The data are case sensitive; furthermore, the system does not allow repeated usernames entered in different letter case. These security features allow students to share their usernames with their advisors, yet keep their data secured through the password. Anonymous accounts also encourage students to communicate freely without repercussions, which is the best way to obtain sincere feedback about the system and the academic services. The account also stores the user information for recurrent advising sessions.

III. THE ACADEMIC ADVISING DIALOG MANAGER

The heart of Albert is the NLP system that drives all the input, output and states of the system, i.e., the advising Dialog Manager (DM). Given the advising task requirements, Albert's DM is built around the ChatScript (CS) scripting language [16]. This work uses CS version 2.0. The design of the DM allows a straightforward method for redrafting and distributing the system for other academic programs, by using variables in the input patterns for the proper nouns and the contents of the well-defined knowledge base. Correspondingly, the advising FAQ corpus input patterns allow for deploying to other academic programs by adding the data to the KB, without reworking the existing input templates.

The DM controls communication between CS and Albert's other functions written in Python and PHP. For example, to manage the unique technical terms and proper nouns, a supplementary spell-checker was constructed in Python using methods surveyed from the literature [20]-[22]. When CS finds a candidate for the spell-checker, it sends the term to the Python spell checker, which returns the possible corrections to CS to determine the system response.

Regarding computational performance, a query response through the website takes approximately one second when tested from computers connected through a local area network.

A. *Natural Language Generation*

Language generation in the DM consists of templates containing text, pointers, variables and other control functions. To create the KB, as no data corpus is available, the output information was obtained through university documents and interviews with the academic advisors. The answers are constructed using logic functions, random generators, control structures and queries. Queries include course-scheduling information from the Registrar's Office webpage and student academic information saved with the academic planning process (APP). The KB has the information organized as a list of triples containing a question, its answer and the topic to which the answer was classified.

The system also contains a trivial amount of grammar rules, mostly for handling unrecognized input and extending off-topic dialog. For example, when responding to unrecognized questions, the system may either change the verb tense or rearrange the parts-of-speech (POS) to let the user know the answer to that question is not available.

B. *Natural Language Understanding*

Input template design involves identifying the keywords, POS tags, noun-phrase chunks and lexical relations of each input statement, then selecting the features that define the keywords of the template and respective topic. Fig. 3 shows an algorithm for the input *who will teach a specified course*. In this example, the topic *course schedule* includes as keywords, the list of all course names, all professor names and the words *professor* and *teach*. Alternatively, the user could also ask, *who is the professor of C++*, or if the request is within the context of the previous input, *who is teaching it*.

```
Algorithm: Respond to Who teaches X course?
    Input string S: Who teaches C++ next semester?
    Desired output: C++ is taught next semester by Name
or C++ is not offered next semester
    If S contains a keyword of the topic course schedule
    If S matches with a Who-Teaches pattern
      If S contains a Term-Phrase keyword
        Calculate the Term value
      Else
        Use currently stored Term value
      If the course C++ exists in the schedule of the Term
        Find the corresponding data element Instructor
          Return C++ is taught by Instructor in Term
      Else
          Return C++ is not offered in Term
```

Fig. 3. Example of a procedure to match an input requesting who teaches a specified course during the next semester

These examples require additional templates either mapped to the algorithm of the template defined above or with a method to determine the context of the previous input, followed by the same mapping. The system identifies the context using features such as the current and previous input keywords, the keywords for the topics matched, the tense of any verb and the state of the variables representing potentially missing keywords. Additional details about language processing in Albert are available in the previous work [16].

For reference resolution and elliptical questions, in addition to the method described above for recognizing context, the system uses the CS feature of rejoinders. Rejoinders are input templates, which follow parent templates that elicit some expected user response. Rejoinders also allowed some input templates at the end of topics to assume certain keywords were implied. The entity recognition problem is managed by defining in CS case-insensitive concepts with pre-classified POS tags. In addition, the system has concepts defined for all the technical terms, neologisms, slang and significant LUs not available in the CS or Word Net dictionaries.

The system also has measures to deal with nonlinguistic ambiguity that the users inadvertently convey. In some cases, the best solution was to return the most likely answers, in other cases, the best solution was to request more information from the user. For instance, when a user asks, *who teaches Circuits in the next term*, the meaning of next term depends on the current date and during the spring semester, it could refer to either the summer or the fall term. For this case, the system will decide on a value for the term variable, determine the response and return the response including a method to quickly obtain the response for another term value. As evident in Fig. 3, once the user states an academic term, the system stores this value to use as current default.

When writing the input templates, the key tradeoff is between over-fitting and not generalizing well, thus increasing missed inputs, or under-fitting and causing false positives. In this work, the precision of the template is inversely proportional to the rate of occurrence of the template. That is, the responses that users most seek have a lower accuracy and higher coverage. In contrast, the precision of the template is proportional to the intricacy of the response, i.e., very specific answers have templates with higher accuracy.

When an input statement does not match with any template, the system will respond with an estimated match or request a new entry. For evaluation purposes, the system classifies these responses generated as not answered correctly. For the tests described in this work, Albert had approximately 415 input templates, for over 200 unique responses.

C. Task Manager

The task manager (TM) represents all the functions Albert executes to complement the dialog task. These functions include user account management, database management, input validation, spell checking, automatic updates, a scaling-up routine, statistical data collection and the APP. The APP is discussed in the next section. The TM is built with Python.

When the DM makes a spelling correction, the system alerts the user that a correction was made and encourages them to verify the new input. When the DM cannot make a definitive correction, the system will give the closest answer and a short list of alternative responses with shortcuts for answers.

Albert has two main procedures for automatic updates; one procedure updates the schedule of courses, the other updates the CS scripts daily. As all template-based systems are limited by the amount of patterns for input matching, ideally the system will include methods to allow scaling-up with minimal involvement from the developers. For this reason, the design of Albert has measures in place to allow such improvements to the system. This work includes the design of one scaling-up routine, specifically, to allow users to suggest unofficial names for courses, such as the nicknames, abbreviations and acronyms that the community commonly uses.

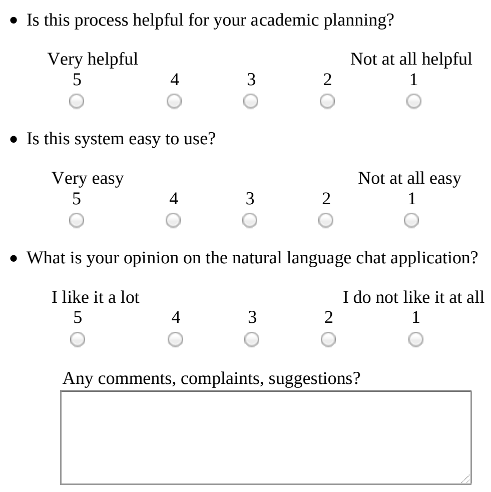
Users have two methods to submit course names to Albert. The first method is through a direct request, i.e., they implicitly state that they want to submit a course name. The second method occurs when the system recognizes a request for course information, but the name of the course is not recognized and no spelling correction was obtained. The system obtains from the user the official name of the implied course and the recommended course moniker. This 2-tuple is automatically sent to the ECE-UF advisor database, which the advisors can access online through an independent webpage developed in this work for the task.

The second phase of the scaling-up procedure involves the ECE-UF advisors using the online system to accept or reject the user proposed course name. If the name is rejected, the process ends. If the name is accepted, the 2-tuple is added to a table in CS, which was designed in this work for this purpose. With the automatic daily updates, this data pair is available for users by the next calendar day.

To evaluate Albert, following the academic advising guidelines of the National Academic Advising Association [23] and the Council for the Advancement of Standards in Higher Education [24], assessment includes direct and indirect evaluation, qualitative and quantitative methodologies, and data collected from students and other constituencies. For qualitative analysis, the measures include collecting feedback from the students and assessment reports from the ECE-UF academic advisors.

For quantitative analysis, data from the user log files are used to measure the information of the input-output messages, login events, message timestamps and suggestions each user made through the scaling-up process. In addition, the system classifies its responses in three categories, a positive response, a negative response and off-topic responses. The system does not provide an automatic estimate of false positive (FP) outcomes, that is, input statements incorrectly matched to a determinate response, or false negative (FN) outcomes, i.e., input that should have matched. Therefore, the analysis of the FP and FN outcomes is done through an estimation of the data. The results from the estimation allow computing the standard statistical evaluation metrics for similar NLP systems, namely, precision, recall, accuracy and the F1 measure [20], [25].

Albert also includes a questionnaire for students who complete the APP.



- Is this process helpful for your academic planning?

Very helpful				Not at all helpful
5	4	3	2	1
○	○	○	○	○

- Is this system easy to use?

Very easy				Not at all easy
5	4	3	2	1
○	○	○	○	○

- What is your opinion on the natural language chat application?

I like it a lot				I do not like it at all
5	4	3	2	1
○	○	○	○	○

Any comments, complaints, suggestions?

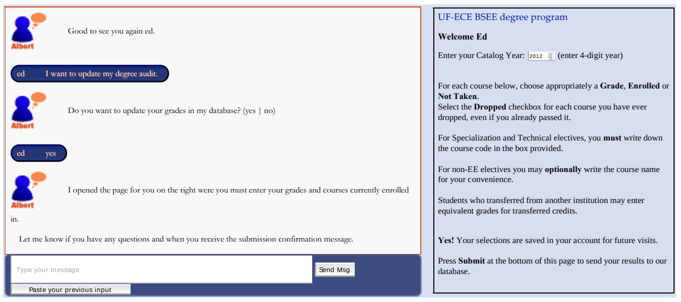
Fig. 4. Questionnaire required to submit the APP information

To submit the APP information, users must answer a questionnaire of three Likert items, as shown in Fig. 4. The figure also shows an optional write-in text area where students can leave feedback. As the system is anonymous, these results and comments are not sent directly to the ECE-UF advisors.

IV. THE ACADEMIC PLANNING PROCESS

The APP is an expert system designed to offer students guidance and recommendations when preparing their course plans for each term. Students can enter their academic record in the APP and receive recommendations on how to develop their academic plan up to graduation, based on courses completed, course prerequisites and all academic rules. Since Albert knows the schedule of ECE courses for each semester, the APP helps students create their course plan for the next semester and send it electronically to their advisor for review.

The algorithms of the system were developed in Python, while the user interface was built with PHP. The user interface of the APP runs inside the Albert webpage as an independent frame, allowing users to work on both systems simultaneously, analogous to a student filling up a form in the advisor's office. Students will complete a course plan and submit it to a database, which advisors can access via an independent website. This process is part of the objective of allowing advisors to integrate Albert into their daily advising practice. Fig. 5 shows a screenshot of Albert, where the web frame on the right shows the initial webpage for APP. The course information is accessible by scrolling down on the frame.



Good to see you again ed.

ed I want to update my degree audit.

Do you want to update your grades in my database? (yes | no)

ed yes

I opened the page for you on the right were you must enter your grades and courses currently enrolled in.

Let me know if you have any questions and when you receive the submission confirmation message.

UF-ECE BSEE degree program

Welcome Ed

Enter your Catalog Year: 2012 (enter 4-digit year)

For each course below, choose appropriately a **Grade**, **Enrolled** or **Not Taken**.
Select the **Dropped** checkbox for each course you have ever dropped, even if you already passed it.

For Specialization and Technical electives, you **must** write down the course code in the box provided.

For non-EE electives you may **optionally** write the course name for your convenience.

Students who transferred from another institution may enter equivalent grades for transferred credits.

Yes! Your selections are saved in your account for future visits.

Press **Submit** at the bottom of this page to send your results to our database.

Fig. 5. A screenshot of Albert with the right side showing the first page of the APP opened inside the frame. The left side, the main dialog window, shows the conversational exchange that took place to initiate the APP

Students can initialize the APP by explicitly requesting it, with expressions similar to *I want to update my degree audit* and *I want to enter my grades*, as evident in Fig. 5. Upon initialization, the APP presents a template with the student's academic curriculum, similar to how it appears in the UF catalog. Using drop-down menus and text-boxes, students can enter the following academic information in this template; all fields in the template are validated upon submission.

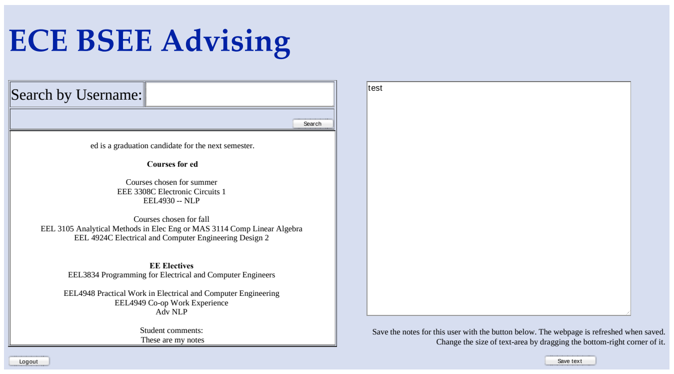
- Catalog year (admission into the academic program)
- Grades in completed courses
- Courses currently enrolled in
- Courses dropped

Ideally, students would access their academic information from the university's digital records and upload the data to Albert. However, at the time, it was not possible to access the university's data and not practical to develop an interpreter for print scans of the data. In any case, freshmen had no records while others only have to enter the data once for their account, and having students examine their grades is a favorable self-assessment exercise for writing the course plan, albeit an exercise most students do not dedicate the effort to complete.

After submitting their records, Albert will invite students to prepare a course plan for the next term. For this process, Albert opens the second webpage of the APP, which contains the courses remaining to complete the degree requirements, the courses the student can take, any course the student can repeat, a checkmark for graduation candidates and the questionnaire in Fig. 4. Appendix C contains a screenshot with this APP frame. Students can repeat at any time, any step of this process, as advisors are not automatically notified of student submissions.

When a student is prepared to discuss the academic plan with the advisor, he will provide the username with Albert, to allow the advisor to obtain the plan from the online database. Students do not need to share their passwords. Fig. 6 shows this webpage, after a successful search of the user *ed*. The webpage provides advisors with a text pad area to save notes about this user.

Regarding computational performance, the response time of the APP is also within the approximately one second delay between events the rest of Albert offers.



ECE BSEE Advising

Search by Username:

ed is a graduation candidate for the next semester.

Courses for ed

Courses chosen for summer
EEE 3308C Electronic Circuits 1
EEL4930 -- NLP

Courses chosen for fall
EEL 3105 Analytical Methods in Elec Eng or MAS 3114 Comp Linear Algebra
EEL 4924C Electrical and Computer Engineering Design 2

EE Electives
EEL3834 Programming for Electrical and Computer Engineers

EEL4948 Practical Work in Electrical and Computer Engineering
EEL4949 Co-op Work Experience
Adv NLP

Student comments:
These are my notes

test

Save the notes for this user with the button below. The webpage is refreshed when saved.
Change the size of text-area by dragging the bottom-right corner of it.

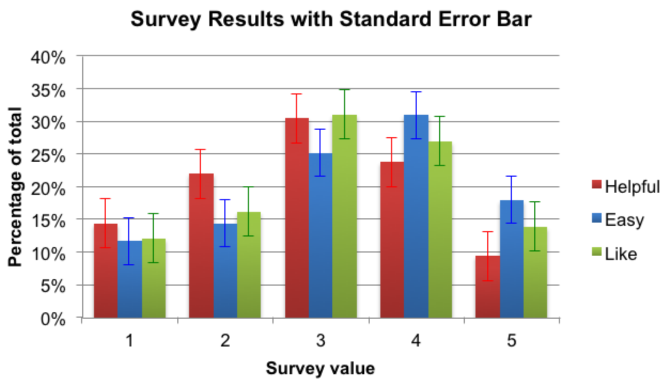
Fig. 6. The advisor database website, with an example student report

V. EXPERIMENTS AND ASSESSMENT

The experiments were designed for students of the BSEE and BSCEE programs, though the APP was only available to BSEE students. There were approximately 950 students in these programs, with approximately 500 in the BSEE program. The tests took place between October 4 and November 27, 2013. The tests began when the ECE-UF advisors informed the students of the availability of Albert and provided the web address for the system. The address was not available in any other medium. During this period, the advising staff was unexpectedly short-handed, so students were encouraged to utilize Albert to get their academic plan approved in time for registration period, except first-year students, who were encouraged to personally visit an advisor. The students did not have workshops on how to use the system. A video tutorial was available via a link in the website; see Fig. 1. The tutorial's website registered about 90 unique cookies during the period.

Results refer to users as unique accounts created. Identifiable log files were removed, including those from faculty and advisors. Users who made less than three statements were also removed. The remaining user files were included in the results, even when the user never meant to converse about academic advising. The system compiled data from 387 users. The data showed that 53% of these users made less than ten entries. Many of these used the system explicitly for the APP process. Registering and completing the APP process required a minimum of four statements. The average login per user was two, as 78% of users had at most two.

From the user total, 292 completed the first part of the APP. From the catalog year data, as expected, most students were in the mid years of academic progress for the four-year program. For the second part of the APP, Albert collected survey results from 224 students. Fig. 7 shows the results from the survey. The results reflect mostly positive reviews, as a majority of students gave values of three and four for how much they liked it and how easy it was. The rating of helpfulness is smeared over two, three and four. Conversational systems with similar surveys obtained averages within the low threes, to four and a half [26]-[27]. However, these systems have a reduced input domain, as the system is who drives the dialog, and users know beforehand what to expect from the system.



Question	Helpful	Easy	Like
Average	2.92	3.29	3.14

Fig. 7. Survey results with a standard error bar for each question

To support the survey results, the comment section was added to the survey on October 12, 2013. Of the 191 users who completed the APP after this date, 61 wrote comments, of which eleven were positive feedback, 45 were negatives remarks and five were technical suggestions with neutral sentiment. The positive remarks were general compliments and appreciating the course plan system. Of the negative comments, 47% criticized the system as not appropriate for substituting a human advisor, 42% criticized having to manually enter their academic record, while the remaining 11% reported technical difficulties and unique situations.

The disapproval of the method for entering the grades was understandable and expected. As described previously, given the privacy concerns, the method was a quick resolution. Once academic departments manage the system, they will have the resources to access the official data upon student request.

To address the comments about substituting a human advisor, the main solution is to inform the students about the objectives for Albert, as the advisor integrates the system into the regular tasks. A basis throughout the design of Albert is that to appreciate the value of this service, students must understand that the objective is to assist and not replace. The fact that almost half of the negative comments concern this statement validates its significance. Unfortunately, the disturbance in the advising services hampered this guideline. Altogether, 89% of the negative remarks comprise two realistic provisions that are straightforward to implement. Appendix B shows dialog extracted from the user log files.

The ECE advisor, who processed the student submissions from Albert, completed a review of the system and the course submission process. In general, the advisor was very encouraging of the system and service provided, with recommendations in line with student reviews. For the scale up process, six users completed submissions for course names, three of which made meaningful contributions.

A measure to evaluate the NLP of the system was developed, where user input is classified as the following.

- Literal match (LM) are input statements that exactly match a FAQ listed on the webpage FAQ examples.
- Partial match (PM) are statements that have partially matching templates.
- Outcome negative (ON) includes false negatives (FN), i.e., statements not recognized and true negatives (TN), i.e., statements outside the design scope.
- Outcome positive (OP) includes correct responses or true positives (TP), and false positives (FP).

The LM statements are the 71 FAQs listed on the webpage. These include the examples for initiating the APP, the help command and an example from each topic in Albert. The LM statements do not have any uncertainty for recognition; therefore, these are subtracted from the total input to evaluate the system error. As the APP was a main feature of Albert, eliminating these commands increases the estimated error. As a subsequent improvement, the webpage includes fewer examples, to encourage users to utilize original expressions.

The PM statements are to help the user obtain the information of interest, return incomplete answers and for statements that are outside the scope of the system, to which the response is a statement related to that topic and lets the user know that more information on that topic is not available. Although these templates were successful in their NLP design, given the user did not directly receive the desired response, these are not classified as outcome positive.

Results are available from 366 users between October 7 and November 27, 2013. Table I shows the results from these users and the amount of input statements under each classification. All non-LM statements are, accordingly, Original statements.

The results in Table I show that about 60% of the students copied an instruction exactly as written, which for all purposes is akin to making a selection from a menu. While this result suggests that the interface could benefit from menu selections, the objective of this experiment is to encourage unrestricted expressions. Having approximately 80% of the students initiate the APP process and 60% using example input shows the student preference of using the quickest possible method to achieve a goal.

Nevertheless, to serve as an educational tool, the previous data showed students benefit from a display of the FAQ. A solution is to include a display of topics with less example statements, while extending the NLP routines that allow users to access data by navigating through topics. While this approach is not within the scope of the vision for Albert, the data shows that including both approaches would increase the user-base by allowing users to explore the capabilities of the system through a more familiar experience.

Table I also shows that less than 15% of all input was not recognized, however, these statements came from 55% of the users. This result is expected from a system with a restricted domain that accepts all type of input. Any user who decides to test the boundaries of the system will contribute to this result. Fig. 8 shows the distribution of the ON classified statements.

To reduce the ON responses, false negatives were continuously identified and updated in the system. As the data collection increased, the number of false negatives decreased. By the fifth week of the almost eight-week period, the amount of false negatives per user had dropped to almost zero. Approximately 25% of the users accessed the system during this culminating period. During these last three weeks, no measures were taken to update the system.

To determine the number of FP and FN statements, it is necessary to manually evaluate the OP and ON statements. For this evaluation, a FP outcome is a response that is not relevant to the input statement. This definition of FP error conditions the system's capabilities of responding to the statement with respect to the available information.

TABLE I. RESULTS OF THE INPUT STATEMENT CLASSIFICATION

	Total	LM	Original	PM	ON	OP
Users	366	60.5%	-	11.9%	55.0%	99.7%
Input	4952	12.5%	4332	1.7%	14.7%	83.5%

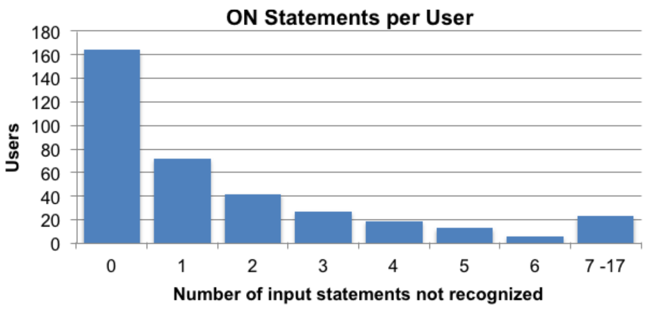


Fig. 8. Distribution of ON input statements

For example, if a user asks, *Who is my financial advisor*, the system response, *For financial information please see the following website*, is not classified FP as the answer is relevant and it is the best response available in the system. If for the same input the system replied *Your BSEE advisor is Mrs. Chillingworth*, the response is classified as FP, given the BSEE advisor is not relevant to the user question.

A FN outcome occurs when the input statement has an available answer, yet the system did not respond correctly. Following the same example as above, if a user asks, *Who is the person in charge of financial advising*, the system response, *I do not understand that question*, is classified FN, as this question should have been responded as stated previously. The FN outcomes include cases caused by technical difficulties in any area of the system; however, with respect to overall system performance, they are incorrect responses. TN outcomes occur when the input statement is outside the design scope.

We estimated FP and FN by selecting random samples from the OP and ON statements. To select the samples, a uniform random number is assigned to each of the 3618 OP statements and to each of the 641 ON statements. Each statement is tested for the binary outcome FP or not FP, and FN or not FN respectively. By treating each statement as an independent random variable, the outcomes of the tests follow a Bernoulli distribution. In this distribution, the maximum variance, $p * (1 – p)$, occurs when the mean p is equal to one half, thus the variance is equal to one fourth. For a given sample size n, by the central limit theorem, the distribution is closest to the standard normal distribution when p is near one half. Under this scenario, a conservative estimate of the sample size n needed to estimate p with a confidence level $1 – \alpha$ and margin of error e is given in (1), where the [ ] operator represents rounding to the next integer and Z_α is the estimated standard score for a given two-sided confidence level [28].

$$n = [\ Z_\alpha / (\ 4\ \square^{\square}\ )\ ] \qquad (1)$$

For a maximum margin of error equal to 10% and a confidence interval of 95%, (1) returns a minimum size of 97 samples. Table II summarizes the result for each test.

TABLE II. RESULTS FROM THE OUTCOME ERROR ESTIMATION

Outcome	Total Statements	Confidence Interval	Error Margin	Sample Size	Error
Positive	3618	95%	10%	97	15.46%
Negative	641	95%	10%	97	24.74%

The tests returned 15 false positive statements, for 15.46% of the OP samples and 24 false negatives for 24.74% of the ON samples. Using these results, it is possible to determine the statistical measures precision, recall, accuracy and the F1 measure [20]. Table III shows the result of each measure.

The results show the system performance metrics are all close to 80%. These results are estimated minimums; given the error is an estimated maximum. Recent studies for comparable e-learning systems show Albert offers a high-level performance for the complexity of the task [29]–[30]. Regarding the main NLP tasks in Albert, specifically keyword extraction, question answering and natural language interfaces, recently published systems that require data corpora for training, obtain results that show the performance of Albert is competitive [31]-[35]. Another performance measure is a study of human accuracy and response time, though such a study is not available. While Albert cannot yet compete in accuracy with a human, its instant response time is hard to beat. In any case, the results show Albert offers a competitive performance with much promise for advancement as data collection continues.

The results for precision and recall are in line with the design goal of providing students with high precision answers and minimizing false positives. At the same time, the results show that over 75% of the queries to the system obtained a reply that is effective versus an input-not-recognized type answer. This result is valuable considering users did not have any training on how to use the system.

Overall, ECE-UF students and personnel were appreciative of the service provided and supportive for future developments. The results offer advisors valuable assessment for the areas that most concern students.

The data shows that, as is customary in electronic text communications and online search engines, students prefer to ask questions using the minimum amount of words possible, i.e., entering isolated keywords, instead of through a Standard English sentence. Therefore, Albert should include methods to allow this user preference, while concurrently inspiring students to use complete expressions. Facilitating speech recognition will also expedite this design.

Initially, students were reluctant to follow a new process for the advising chores. Indeed, more effort is required in marketing the service as a practical option versus visiting a human advisor, who could be a short walk away. However, during periods when the waiting time is long or when the advisor is not available, the predisposition to experiment with an option is very high.

VI. Conclusion and Future Work

A. Conclusion

The fundamental objective of this work is to provide students with an automated online advising experience that is as close as possible to traditional human interaction.

TABLE III. SYSTEM NLP PERFORMANCE ESTIMATION RESULTS

Precision	Recall	Accuracy	F_1 measure
$\geq$ 84.54%	$\geq$ 77.36 %	$\geq$ 79.90%	$\geq$ 0.81

This work presents the design, development, deployment and evaluation of an intelligent natural language conversational system for academic advising. The online system includes an advising dialog engine, an academic planning system that integrates the students with the advisors, and a method to allow the users to contribute to the system's KB.

This work contributes real-world solutions for the academic community, through a unique combination of software applications and advanced NLP techniques. With the experimental data collected, this work has originated a KB of academic advising FAQs that will serve to build a corpus on the topic, to allow the development of systems for other academic programs by mostly replacing answers.

This system is currently in operation for students of the BSEE and BSCEE degree programs of ECE-UF. Experiments with the students showed Albert is a practical and valuable application for academic advising, based on positive reviews from all constituents and system performance analysis. Within a confidence interval of 95% and a margin of error of 10%, Albert showed an estimated precision of 85%, a recall of 77%, an accuracy of 80% and a F1 measure of 0.81. Overall, users were supportive and excited about future developments.

B. Future Work

Albert continued to serve the ECE-UF department during the year 2014. During Spring 2014, the system featured a twofold process for the APP, in which students could simultaneously prepare a course plan for summer and fall.

The immediate development is adding a computerized method for submitting the academic record, preferably through the university's online system. Obtaining the full academic record from the students will increase the services the system can implement for students and advisors.

Additional future work includes an escalation mechanism to forward selected student conversations to the advisors, adding academic information from other academic departments and increasing the KB. With an extensive KB, the system can include statistical learning techniques to the NLP system.

VII. Appendixes

A. Topics included in Albert

- Academic curriculum and rules
- Course schedules
- Internships
- Graduate programs and minors
- Financial assistance
- Faculty and staff
- The academic calendar
- Student organizations
- University campus
- Information about Albert

B. User Dialog from the Experiments

The following are excerpts from the log files of three users. User statements are in bold font weight and their syntactic errors are preserved for authenticity. Italic type represents actions. Albert's answers are curtailed with an ellipsis.

User 1 – F9788

What courses do you recommend I take? ==> I will tell you which courses you could take next semester … Do you want to enter your grades into the database? (yes | no)

y ==> *The APP is launched at this time.*

confirm ==> *The first part of the APP is completed.* Do you want to choose your courses for the next semester and submit the list to your advisor?

y ==> *The second part of the APP is launched at this time.*

what courses do you recommend I take? ==> Assuming your info is up to date, these are the courses you could take next semester. <br> <br> EEL3472 Electromagnetic Fields …

User 2 – Willwrdn

Welcome back willwrdn

I want to drop a course ==> So you want to drop a course? Are you in upper division or lower division? …

upper ==> Students are given TWO drops after completing 60 credit hours. Any unused lower division drop…

what is the deadline for dropping a course ==> I'll tell you this, in rare cases it is possible to drop a course after the …

am I able to drop a course via ISIS? ==> So you want to drop a course? Are you in upper division or lower division? …

upper ==> Students are given TWO drops after …

so how do I drop a course? ==> You drop classes via ISIS. If the add/drop period has ended you can still do it, but …

alright! thank you! That's all the information I need today ==> Well, I am learning every day. What can I do for you?

logout ==> Before you go, please share any feedback …

Perhaps it was just that my queries were better and fewer this time, but the system seems to have improved. Good job! I hope it gets to access to more information soon so that it can answer more questions instead of some variation of I don't know ==> All right, thanks! <br> Later, Gator!

User 3 – Novae

show my schedule ==> You want me to show you the courses you chose for next semester? OK. These are your …

programming classes ==> Electrical Engineering students must take one programming course in Java, C or C++ …

prerequisite for eel 3701c ==> The prerequisite for EEL 3701C is a Programming Course.

is eel3834 being taught next semester? ==> The course EEL 3834 is not offered in Spring 2014.

what classes can i take ==> Assuming your info is up to date, these are the courses you could take next semester …

circuits 2 ==> I recognize CIRCUITS 2 is a course in our department, but could you be more specific about …

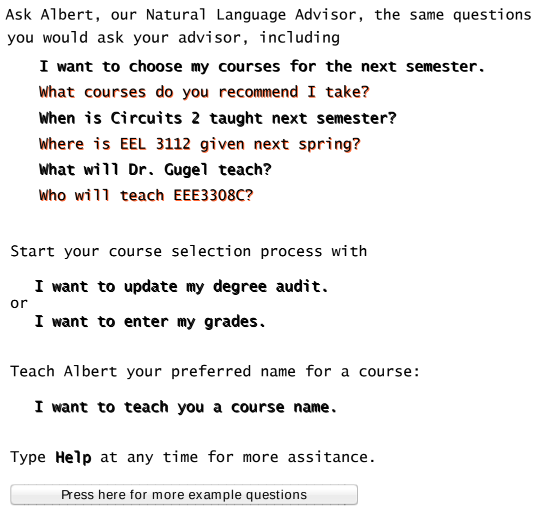
Who will teach EEE3308C? ==> For Spring 2014 we have <br> EEE 3308C by Yoon,Yong Kyu

eel3701c any sections available ==> For Spring 2014 we have <br> EEL 3701C by Gugel,Karl S

Is it ok if I take less than 12 credit hours? ==> UF policy states that the minimum full-time load for undergraduate…

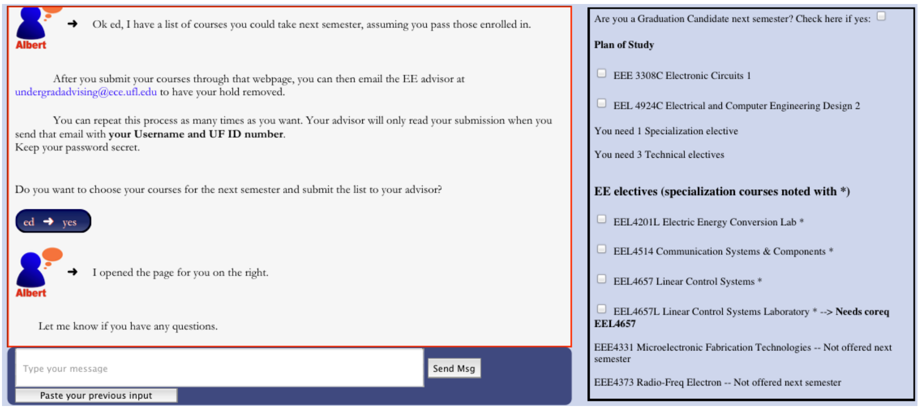
How many critical tracking courses are required for my major? ==> You need to pass 8 critical tracking courses...

C. Screenshots of the Website



```
Ask Albert, our Natural Language Advisor, the same questions
you would ask your advisor, including

   I want to choose my courses for the next semester.
   What courses do you recommend I take?
   When is Circuits 2 taught next semester?
   Where is EEL 3112 given next spring?
   What will Dr. Gugel teach?
   Who will teach EEE3308C?


Start your course selection process with

   I want to update my degree audit.
or
   I want to enter my grades.


Teach Albert your preferred name for a course:

   I want to teach you a course name.


Type Help at any time for more assitance.
```

Fig. 9. Main FAQ webpage



→ Ok ed, I have a list of courses you could take next semester, assuming you pass those enrolled in.

After you submit your courses through that webpage, you can then email the EE advisor at undergradadvising@ece.ufl.edu to have your hold removed.

You can repeat this process as many times as you want. Your advisor will only read your submission when you send that email with **your Username and UF ID number**.
Keep your password secret.

Do you want to choose your courses for the next semester and submit the list to your advisor?

ed → yes

→ I opened the page for you on the right.

Let me know if you have any questions.

Are you a Graduation Candidate next semester? Check here if yes: ☐

Plan of Study

☐ EEE 3308C Electronic Circuits 1

☐ EEL 4924C Electrical and Computer Engineering Design 2

You need 1 Specialization elective

You need 3 Technical electives

EE electives (specialization courses noted with *)

☐ EEL4201L Electric Energy Conversion Lab *

☐ EEL4514 Communication Systems & Components *

☐ EEL4657 Linear Control Systems *

☐ EEL4657L Linear Control Systems Laboratory * --> **Needs coreq EEL4657**

EEE4331 Microelectronic Fabrication Technologies -- Not offered next semester

EEE4373 Radio-Freq Electron -- Not offered next semester

Fig. 10. Main webpage for Albert showing an example of the second part of the APP

ACKNOWLEDGMENT

We appreciate the support from participating ECE-UF students, advisors and personnel of the student services office.

REFERENCES

[1] V. N. Gordon, W. R. Habley, T. J. Grites and National Academic Advising Association. Academic Advising: A Comprehensive Handbook, 2nd ed. San Francisco: Jossey-Bass, 2008.

[2] K. Soria, "Advising satisfaction: implications for first-year students' sense of belonging and student retention", in The Mentor: An Academic Advising J., Oct 2012. [Online]. Available: http://dus.psu.edu/mentor/2012/10/advising-satisfaction/ Retrieved Feb 2013.

[3] M. J. Leonard, "Advising delivery: using technology", in Academic Advising: A Comprehensive Handbook, V. Gordon, W. R. Habley, T. J. Grites and National Academic Advising Association, Eds. San Francisco: Jossey-Bass, pp. 292-306, 2008.

[4] L. Waldner, D. McDaniel, T. Esteves and T Anderson, "The eQuad: a next-generation eAdvising tool to build community and retain students", in The Mentor: An Academic Advising J., Oct 2012. [Online]. Available: http://dus.psu.edu/mentor/2012/10/equad-eadvising-tool-build-community-retain-students/ Retrieved Feb 2013.

[5] National Academic Advising Association, Advising Technology Innovation Awards. [Online]. Available: http://www.nacada.ksu.edu/Events-Programs/Awards/Association-Awards/Technology/Technology-Winners.aspx Retrieved Feb 2013.

[6] H. S. Hart, R. B. Hussey, M. J. Leonard, J. Levin and S. M. Winck, "eLion: Penn State's comprehensive web-based academic advising system", in The Mentor: An Academic Advising J. [Online]. Available: http://dus.psu.edu/mentor/bookstore/elion-advising-system/ Retrieved Feb 2013.

[7] University Advising Center, The University of Texas at Arlington, Arlington, TX. [Online]. Available: http://www.uta.edu/universitycollege/current/academic-planning/uac/index.php Retrieved Feb 2013.

[8] Academic Advising and Career Center, eAdvising, University of Michigan-Flint, Flint, Michigan. [Online]. Available: http://www.umflint.edu/advising/eadvising.htm Retrieved Feb 2013.

[9] R. M. Siegfried, A. M. Wittenstein and T. Sharma, "An automated advising system for course selection and scheduling", J. Comput. Sci. Coll., vol. 18, no. 3, pp.17-25, Feb 2003.

[10] F. Albalooshi and S. Shatnawi, "HE-Advisor: A multidisciplinary web-based higher education advisory system", Global J. Computer Sci. and Tech., vol. 10, no. 7, pp. 37-49, Sept. 2010.

[11] A. N. Nambiar and A. K. Dutta, "Expert system for student advising using JESS", Int. Conf. Educational and Inform. Tech., vol. 1, pp. V1-31 – V1-315, Sept. 2010.

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

[13] E. Onyeka, D. Olawande and A. Charles, "CAES: A model of an RBR-CBR course advisory expert system," Int. Conf. Inform. Soc., pp.37-42, June 2010.

[14] B. C. McMahan and R. A. Bates, "An automatic dialog system for student advising", in J. Undergraduate Research, Minnesota State University, Mankato, 2010.

[15] C. M. Leung, E. Y. M. Tsang, F. S. S. Lam and D. P. C. Wai, "Intelligent counseling system: a 24 x 7 academic advisor," EDUCAUSE Quart. 33, no. 4, 2010. [Online]. Available: http://www.educause.edu/edUCaUSe+Quarterly/edUCaUSeQuarterlymagazineVolum/intelligentCounselingSystema24/219101 Retrieved Feb 2013.

[16] E. Latorre and J. Harris, "A natural language conversational system for online academic advising", Florida Artificial Intell. Research Soc. Conf., North America, May. 2014. Available: http://www.aaai.org/ocs/index.php/FLAIRS/FLAIRS14/paper/view/7842 Retrieved Dec 2014.

[17] D. C. Appleby, "Advising as Teaching and Learning", in *Academic Advising: A Comprehensive Handbook*, V. Gordon, W. R. Habley, T. J. Grites and National Academic Advising Association, Eds. San Francisco: Jossey-Bass, pp. 85-102, 2008.

[18] P. L. Hagen and P. Jordan, "Theoretical foundations of academic advising", in *Academic Advising: A Comprehensive Handbook*, V. Gordon, W. R. Habley, T. J. Grites and National Academic Advising Association, Eds. San Francisco: Jossey-Bass, pp. 17-35, 2008.

[19] Contributors to the Web Speech API Specification, Web Speech API, Speech API Community Group, 2012. [Online]. Available: https://dvcs.w3.org/hg/speech-api/raw-file/tip/speechapi.html Retrieved Feb 2013.

[20] D. Jurasky and J. H. Martin, Speech and Language Processing An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, 2nd Ed, New Jersey: Pearson Education, 2008.

[21] S. Bird, E. Klein and E. Loper, Natural Language Processing with Python, O'Reilly Media, 2009. [Online]. Available: http://nltk.org/book/ Retrieved Feb 2013.

[22] P. Norvig. *How to Write a Spelling Corrector*. [Online]. Available: http://norvig.com/spell-correct.html Retrieved Feb 2013.

[23] National Academic Advising Association, Kansas State University, Manhattan, KS. https://www.nacada.ksu.edu

[24] Council for the Advancement of Standards in Higher Education, "The Role of Academic Advising Programs", Washington DC, 2011.

[25] R. A. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*, Addison-Wesley Longman Publishing, Boston, MA, 1999.

[26] S. Gandhe, N. Whitman, D. Traum and R. Artstein, "An integrated authoring tool for tactical questioning dialog systems", *Proc. 6th IJCAI Workshop on Knowledge and Reasoning in Practical Dialog Systems*, Assoc. for the Advancement of Artificial Intell., 2009.

[27] D. Mori, R. Berta, A. De Gloria, V. Fiore and L. Magnani, "An easy to author dialog management system for serious games", *J. Comput. Cult. Herit.*, vol. 6, no. 2, pp. 10:1-10:15, May 2013.

[28] J. A. Gubner. Probability and Random Processes for Electrical and Computer Engineers, Cambridge University Press, Cambridge, United Kingdom, 2006.

[29] A. Olney, M. Louwerse, E. Matthews, J. Marineau, H. Hite-Mitchell, and A. Graesser, "Utterance classification in AutoTutor", HLT-NAACL Building Educational Applicat. using NLP, *Assoc for Computational Linguistics*, vol. 2, 1-8, USA, 2003.

[30] W. Ward, R. Cole, D. Bolaños, C. Buchenroth-Martin, E. Svirsky, S. Van Vuuren, T. Weston, J. Zheng and L. Becker, "My science tutor: a conversational multimedia virtual tutor for elementary school science", *ACM Trans Speech and Language Process.*, TSLP, vol. 7, no. 4, 2011.

[31] D. de Castro Reis, F. Goldstein and F. Quintao, "Extracting unambiguous keywords from microposts using web and query logs data", *2nd Workshop Making Sense of Microposts*, 21st Int. Conf. World Wide Web, 2012.

[32] E. Sneiders, "Automated FAQ answering with question-specific knowledge representation for web self-service", *Proc 2nd Int Conf Human Syst Interaction*, IEEE, May 21-23, Italy, pp. 298-305, 2009

[33] M. Minock, "C-Phrase: a system for building robust natural language interfaces to databases", *Data & Knowledge Eng.*, vol. 69 no. 3, pp. 290-302, 2010.

[34] M. Olvera-Lobo and J. Gutiérrez-Artacho, "Open-vs. restricted-domain QA systems in the biomedical field," *J. Inf. Sci.*, vol. 37, no. 2, pp. 152-162, 2011.

[35] V. Lopez, C. Unger, P. Cimiano and E. Motta, "Evaluating question answering over linked data", *Web Semantics: Sci., Services and Agents on the World Wide Web*, vol. 21, pp. 3-13, 2013.