A DYNAMIC DIALOG SYSTEM USING SEMANTIC WEB TECHNOLOGIES

by

Mohammad Ababneh
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Information Technology

Committee:

______________________________	Dr. Duminda Wijesekera, Dissertation Director
______________________________	Dr. Arun Sood, Committee Member
______________________________	Dr. Rao Mulpuri, Committee Member
______________________________	Dr. Paulo Costa, Committee Member
______________________________	Dr. Stephen Nash, Senior Associate Dean
______________________________	Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering
Date: _________________________	Spring Semester 2014 George Mason University Fairfax, VA

A Dynamic Dialog System Using Semantic Web Technologies

A Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

by

Mohammad Ababneh
Master of Science
Naval Postgraduate School, 2000

Director: Duminda Wijesekera, Professor
Department of Information Technology

Spring Semester 2014
George Mason University
Fairfax, VA

DEDICATION

This is dedicated to my wife Huda, my three wonderful children Leen, Zaid and Ahmad, and my parents Fatimah and Ahmad Ababneh for their never ending encouragement and support.

ACKNOWLEDGEMENTS

I would like to thank the many friends, relatives, and supporters who have made this happen. Dr. Duminda Wijesekera provided great guidance and mentorship. Drs. Paulo Costa Arun Sood and Rao Mulpori of my committee were of invaluable help. Finally, special thanks to Dr. Mark Pullen who had kept me interested in getting my degree from George Mason University during the years that I couldn’t start my studies after my initial offer of admission because of my service commitments.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF ABBREVIATIONS AND SYMBOLS

Item Response Theory	IRT
Computerized Adaptive Testing	CAT
IRT Ability Estimate	θ
IRT difficulty parameter	b
IRT discrimination parameter	a
IRT guessing parameter	c

ABSTRACT

A DYNAMIC DIALOG SYSTEM USING SEMANTIC WEB TECHNOLOGIES

Mohammad Ababneh, Ph.D.

George Mason University, 2014

Dissertation Director: Dr. Duminda Wijesekera

A dialog system or a conversational agent provides a means for a human to interact with a computer system. Dialog systems use text, voice and other means to carry out conversations with humans in order to achieve some objective. Most dialog systems are created with specific objectives in mind and consist of preprogrammed conversations. The primary objective of this dissertation is to show that dialogs can be dynamically generated using semantic technologies. I show the feasibility by constructing a dialog system that can be specific to control physical entry points, and that can withstand an attempt to misuse the system by playing back previously recorded conversations. As a solution my system randomly generates questions with a pre-specified difficulty level and relevance, thereby not repeating conversations. In order to do so, my system uses policies to govern the entrance rules, and Item Response Theory to select questions derived from ontologies relevant to those policies. Furthermore, by using contextual reasoning to derive facts from a chosen context, my dialog system can be directed to generate

questions that are randomized within a topic, yet relevant to the task at hand. My system design has been prototyped using a voice interface. The prototype demonstrates acceptable performance on benchmark data.

CHAPTER ONE: INTRODUCTION

A dialog system or a conversational agent provides a means for a human to interact with a computer system. Dialog systems use text, voice and other means to carry out conversations with humans in order to achieve some objective. Most dialog systems are created with specific objectives in mind and consist of preprogrammed conversations.

Access control points such as computing resource access mechanism, human guarded gates, border control points and embassy visa issuance counters aim to provide rights for access or entry into facilities or geographical regions to those that can be identified correctly and authorized legitimately. Legitimacy is usually determined by rules, regulations or policies known to entry control personnel or enforcement points whose duty is to ensure that these policies are enforced while admitting requesters. Correctly identifying a person may require an examination of presented electronic passports, identity cards and paper documents in addition to questioning about the information directly contained in them or related to some attributes present in them.

In order to determine these two aspects, controlling authorities hold an interview, in which an aspiring entrant is asked a series of questions, and possibly show some documents and demonstrate some knowledge about the contents of the documents or attributes contained in them. Successful interviews should have questions that are relevant, of a reasonable level of difficulty (i.e. not too difficult nor common knowledge) and not to have been asked in prior interviews for the same purpose without drawing accusations of bias from rejected entrants. Ideally, a successful interview should accommodate differences in accents and provide assurance that it is unbiased against similar attributes. A similar situation occurs when one attempts to reset a forgotten password where the password resetting system asks a set of questions related to information stored when the original identity was established within the system.

1.1 Interactive Dialogs

Given the recent success of Interactive Voice Response (IVR) systems such as auto attendants, satellite navigation, personal assistants and mobile applications supported by Apple's Siri [41], Google's Voice [42], Microsoft's Speech [43], I investigated the possibility of specializing IVR systems for access control such as: Visa interviews, entry point interviews, biometric enrollment interviews, password reset, granting access to resources, etc. This kind of dialog supported by voice technology can help illustrate the objective of automating dialogs of critical purposes. Although IVR systems have come a long way in recognizing human voice, and responding to human requests as if responses come from another human, most of the existing IVR systems known to us are pre-programmed with questions and their acceptable answers, and consequently have limited capability in satisfying computerized access control.

There are limitations to the use of IVR's in granting or denying rights to requesters. The first minor limitation of current IVR systems comes from the fact that the human starts and drives the conversation. The second but major limitation is that most IVR systems have a finite number of pre-programmed conversations. Therefore the set of questions generated by such a system are the same for every conversation. This limitation may expose the set of questions so that aspiring entrants may come with prepared question-answer pairs, even if the subject matter of the questions may be unfamiliar to them. Consequently, having the ability to select questions from a large pool may resolve this limitation. The third limitation is that when selecting a random set of questions from large pool, the set of questions asked may not have the desired overall level of difficulty to challenge the user. Solving this issue is relevant because all aspiring entrants expect to have a fair interview. The forth limitation is that questions must be able to discriminate between someone that knows the subject matter from someone who guesses an answer.

As a solution, I have designed and implemented a prototype of an ontological inference based IVR system that uses Item Response Theory (IRT) to select the questions and estimate trustworthiness [5]. The system uses the access control markup language (XACML) as a base to specify attributes of eligible subjects and enforces access policy. Unfortunately, these policies are very static and consist of rules and consequently cannot accommodate implicit knowledge or semantic alternatives to attribute values. If one adds that in the case of an IVR, the possibility of wrong answers caused by the quality of recognition software, network delays or accent variations then system may provide an incorrect admission decision that may render the system unusable. It is here, where I use ontology and reasoning to avoid some of the aforementioned issues and design a better system for access control using voice technologies. As a solution, my system uses an ontology that is used to generate the terms in the access control policy and the application context, and generate questions derived from those ontologies. Because my system does not store any information or save previous question-answer pairs, there is no historical information that can be used to defeat my system.

In this dissertation, I initially introduce a policy-based interactive voice response system to be used to control access to resources [1]. At the second stage, I enhance the system with IRT to overcome the previously mentioned limitation of selecting a relevant question with an acceptable level of difficulty by selecting questions from large amount of attributes present in a policy [2]. Subsequently, I present an enhancement that integrates ontologies and reasoning, where IRT is used to select questions from large amounts of asserted and inferred facts present in ontology [3]. The distinction between the two is that questions asked from XACML polices are about attributes contained in the policy while the questions generated from ontology after an inference provide a wider range of related attributes that can provide a better discernment between genuine requester and an imposture with some stolen identity attributes.

Thirdly, I extend the Ontology-aided identity and access control system by including questions related to the base attribute’s current context and the historical context of the same requester with the system in order to ascertain the interviewee’s familiarity, and provide a score for the entire set of answers [44]. I have also added the capability to generate the succeeding question based on the accuracy of the preceding question. I do so by aligning each attribute with an Ontology that encodes the subject matter expertise on that attribute and derive facts from these ontologies using reasoners to generate facts that I turn into questions. I then assign weights to these derivations based on the axioms and rules of derivations used in the proof tree.

1.2 Objective

The primary objective of this dissertation is to show that dynamic dialogs with an objective like access control can be generated using semantic web technologies. I show the feasibility by constructing a dialog system that can be specific to control physical entry points, and that can withstand an attempt to misuse the system by playing back previously recorded conversations.

1.3 Contribution

This dissertation shows that a dynamic dialog system can be constructed using semantic web technologies; specifically, ontology, reasoners and contextual reasoning. Such a dialog system can replace a human or an agent at an access control point such as a physical gate, border entry points of a country or a computing resource. The trustworthiness of a requester’s answers can be measured quantitatively. The generation of these dialogs from large sets of data represented by ontologies and policies is within an acceptable performance.

My dissertation also shows that a theory widely used by psychometricans and in managing and selecting exam questions in standardized tests can be used in dialog managing adaptive and dynamic dialogs.

Lastly, this dissertation shows that contextual reasoning can enhance dialog dynamic management and even IRT itself. It enhances dialog management by adding a sematic dimension to the quantitative parameters utilized in IRT used to generate questions that have quantitative properties capable of estimating user’s ability within an application context.

1.4 Applications

Several categories of people and areas of application can take advantage of my voice based access control system, as listed below.

1.4.1 Physically Disabled

The system developed provides secure access to resources to people who are unable to use their hands or use a visual interface provided by traditional access control systems.

1.4.2 Homeland Security

A system like mine can be used at access control points such as human guarded gates, border control points and embassy counters that issue entry permits and visa that provide rights for entry into facilities or geographical regions. Also, such a system can be used in assessing the ability to hold sensitive security clearance of people, employees or contractors by automating parts of the interviewing process to accelerate it, better assign resources and possibly detect potential issues at an early stage of the evaluation.

1.4.3 Military and First-Responder Applications

It is important to provide hands-free user interfaces to military personnel, especially those participating in combat operations, as well as first responders (e.g., firefighters, police, and paramedics). These personnel have to access support facilities (e.g., gates at a forward operating bases or police sub-stations) and equipment (e.g., fire fighting equipment and trucks).

1.4.4 Hands-free Mobile Computing and Electronic Business

The adaptation of the voice technology and Voice User Interface in mobile computing (e.g., Apple’s Siri, Google’s Android S-Voice, Microsoft Windows Speech

Recognition, Research in Motion's BlackBerry) had introduced challenges in using the technology in order to accomplish more sophisticated tasks such as access control to resources and services, either locally on the devices or remotely at data centers that reside on clouds. I expect this system to be useful for providing ubiquitous access control using mobile devices, where hands-free usage is valued or required.

In general this access control can be used to provide appropriate access to any information system using a Voice User Interface (VUI). This access control approach can be used in performing transactions in e-commerce applications accessed via mobile computing devices.

1.4.5 Physical Access Control

Access control to critical facilities can be automated and access can be granted or denied based on an enterprise's policy. Policy languages such as XACML are standardized to unify policies across enterprises and reduce administrative load. This is all hands-free and it only depends on answers to questions that represent the subject's attributes. A sample dialogue for access control is illustrated in Section 1.6.

1.4.6 Authentication

The access control system using voice can be looked at as a system of identification representing the "what-you-know" part of the famous identification definition in addition to "what-you-have" and "what you are". The system grants or denies access based on what a user knows about facts governing access enforced by enterprise policies. The system can be supplemented by voice authentication using person's voice biometrics and cover the "what-you-have" dimension of this trio.

1.5 Challenges to Voice Dialogs

The use of a voice system in a dialog for access control suffers from discrepancies. Here, I present some of them:

1.5.1 Dialog Length

The natural limitation of human capability to deal with voice versus vision enforces itself on IT applications. Users can receive and submit fewer amount of information through voice than through vision. Consequently, relying on voice to enforce access to resources protected by complex and large policies will require long conversations asking about facts one at a time. This might turn this methodology useless when people questioned may feel furious and maybe abandon the interaction. Another effect of this situation would be the formulation of long physical lines or long waiting time for callers eager to speak with the controller.

1.5.2 Privacy

There are measures that can be taken to limit the visibility of a screen to one person, but there are no such measures for voice. The system has to ask a question loud and clear and the responder is expected to answer it loud and clear, for otherwise it might be recognized incorrectly by voice recognition software. Also, having people in line after the one served might expose the answer (the user's response), which might be very sensitive information like a social security number or a bank account number. But most IVR systems also can take textual inputs.

1.5.3 Item Exposure

A voice based dialog system may expose questions, which may be a security rule or a fact derived from an ontology fact to be enforced by the policy. By standing in line behind a person being questioned, a second person can hear the question and find the answer or come next time prepared.

1.5.4 Relevance

In the developed system, I used a theory accepted by physiometricians to solve the problem of limiting the length of a conversation without compromising the difficulty of the generated questions. This theory, IRT relies on quantitative parameters of items that ignore any semantic sequences. IRT only uses numerical properties of consequent questions, and my work enhances this aspect by generating questions that are relevant and are semantically connected, while following the theoretical guidelines IRT.

1.5.5 Voice Recognition Accuracy

Even with measurable leaps in the progress of voice recognition software, they do not provide 100% accuracy of recognition. Misinterpretations of a user's utterance will occur. Add to that the effect of accent in spoken language which might result in inaccurate recognition. In a traditional voice system that gives permission based on a raw sum, access would be denied. My work does mitigate this difficulty by giving the user another chance to answer other questions correctly and thereby gaining access.

1.5.6 Inadvertent Mistakes

Sometimes, users miss a question or give a wrong answer inadvertently. In a traditional system of access control that would be counted as wrong and access would be denied. This will cause the user either to lose interest in service and cause loss of business or try again from the beginning, but of course, with resentment. My system doesn't rely on raw scores and attempts to accommodate this kind of mistakes, of course after making sure that mistake doesn't really affect the quality of the user's qualification.

1.6 Use Cases

To validate the usability of the IVR access control system, I have implemented a prototype with two Use Cases.

1.6.1 Use Case 1: Physical Access Control

In this use case, an aspiring person is requesting access to a building guarded by an automated voice system. The user doesn't need to provide any biometrics and access is granted or denied according to rules in a policy, which have attributes of subjects that are allowed to have access to this resource. The dialog in Table 1 shows an illustration of the concept implemented in the physical access control IVR system.

Table 1 A Sample policy-based physical access control dialog

	Question/Answer
System	Welcome to the Engineering School. What is your name?
User	Nick
System	What is your e-mail address?
User	nick@gmu.edu
System	What role do you play in the department?
User	Professor
System	What is your department?
User	Computer Science
System	Welcome again. You satisfy access criteria. Please come in.

As can be seen from the Table 1, there are questions asked by the system to an aspiring user to enter the facility. These questions are generated from an access control policy and conveyed verbally to the user. Using all policy rules as they are without being able to differentiate between them would produce dialogs that suffer from problems already introduced in Section 1.5.

1.6.2 Use Case 2: Homeland Security

In this use case, dialogs are generated from ontology built for homeland security purposes. The dialog takes place between an automated system and an aspiring user at a consulate office, border entry point or security clearance interview. The system asks the user multiple questions drawn from asserted or inferred facts about him and the context around that he should know. The access or forwarding decision is made based on an estimation procedure using IRT.

Table 2 Homeland security dialog

	Question/answer
System	What is your name
User	Dias Kadyrbayev
System	How old are you?
User	19
System	I see you have visited Russia, which part?
User	Dagestan
System	Was there any terrorist attacks while you were there?
User	No
System	Do you know Dzhokhar Tsarnaev?
User	Yes
System	Where is he?
User	He is in Georgia
System	Are you still attending UMass Dartmouth?
User	Yes

Table 2 shows questions and answers in the interview. In this Use Case, the major source of questions is an ontology. The knowledge in the ontology is relied on to make the access control decision. By the end of the dialog, the system should be able to estimate a trust or ability value that the system compares against a threshold to grant or deny access.

1.7 Dissertation Organization

The dissertation is organized as follows. Chapter 2 presents background information and related work. Chapter 3 presents the policy-based IVR for access control. Chapter 4 presents the ontology-based access control. Chapter 5 presents the ontology-based access control supported with contextual reasoning. Chapter 6 presents work analyzing the performance of my contextual reasoner. Chapter 7 concludes the dissertation.

CHAPTER TWO: RELATED WORK

In this chapter I describe the theory and technology used in the access control system. I also go through some of what have the others done close to this work.

2.1 CAT

The Computerized Adaptive Tests became possible and flourished in the 1980's mainly after computers found their ways to the desktop outside data centers. Earlier, in the 1970's, computers were used in tests, but maybe only for scoring and keeping records of scores. The core of CATs is Item Response Theory (IRT), which has been conceptualized and developed during the 1950s and 1960s by psychomorticians to measure trait or ability of people, but wasn't used widely because of the intensive computations required that are tedious without a computer. It became possible and practical to conduct adaptive tests for the masses only when the personal computer became available, affordable and easy to use [7, 8].

In a classical test, the examinee gets a fixed form of questions and has to answer all of them. The score is proportional to the number of correct answers. Most of the form items were of a moderate difficulty with fewer items at the extremes to accommodate the fact that examinee communities would fit a bell shaped curve where most would have a median ability with fewer at the high or low ends. Some forms might be easier than the others and a real comparison between two examinees taking two different forms with different questions from the pool can't really be concluded. Equating and scaling techniques were introduced to come up with a ranking criteria. This type of exams has many problems other than equating and scaling such as exam security, which happens when the exam items become exposed because of limited number of static forms and questions, timing, easy or hard forms and dealing with high and low ability.

On the other hand, in a CAT, if an examinee is asked a question that turned out to be too difficult for the examinee and he got it wrong, then the next could be made easier automatically. If the examinee got it right, the next one would be a little harder. The questions will continue until we are able to estimate the examinee's proficiency within some predetermined level of accuracy.

CAT had introduced many advantages and made testing more practical. CAT had increased the efficiency of the tests reducing the time necessary to reach a conclusion of the person's ability level saving resources and reducing costs. It also improved exam security while providing individuals to work at their own pace. The test is scored immediately, pretesting and experimental tests became easier and larger item pool became possible.

The core of CAT is Item Response Theory (IRT), which is the main force driving adaptive testing as we will see in the next section.

2.2 IRT

Item Response Theory (IRT) is also sometimes called latent trait theory [5]. This is the most popular modern testing theory (as opposed to classical testing theory).

Naturally, IRT is based on stronger assumptions than classical test theories and takes a more intuitive approach to measurement. In IRT, the true score is defined on the latent trait of interest rather than on the test in contrast to classical testing theory. IRT is popular because it provides a theoretical justification for its wide applicability [6].

IRT presents a mathematical characterization of what happens when an individual meets an item [5]. Each person is characterized by a proficiency parameter or latent trait (usually denoted theta or θ), which represents his ability and each item by a collection of parameters such as: difficulty (b), discrimination (a) and guessing factor (c). IRT compares the examinees proficiency level and compares it with the item difficulty level and predicts the probability that the person will answer the question correctly. If the person's proficiency level is higher than the difficulty level, then the probability will be large. If the person's proficiency level is lower than the difficulty level, then the probability will be small. We learn the most information about an item when the probability is close to one-half ($p=0.5$). The item selection algorithm tries to select the items that yield the greatest amount of information in the shortest possible time (minimum time) while also satisfying content considerations that are critical for a good test. The final examinee's proficiency is calculated from the difficulty of the items presented to him [5, 8]. The ability or trait usually ranges from $-\infty$ to $+\infty$, but for computational reasons usually acceptable values ranges from -3 to +3. Figure 1 shows a graph of illustrating the relationship between the ability (θ) and the probability of answering a question correctly usually called Item Characteristics Curve (ICC).

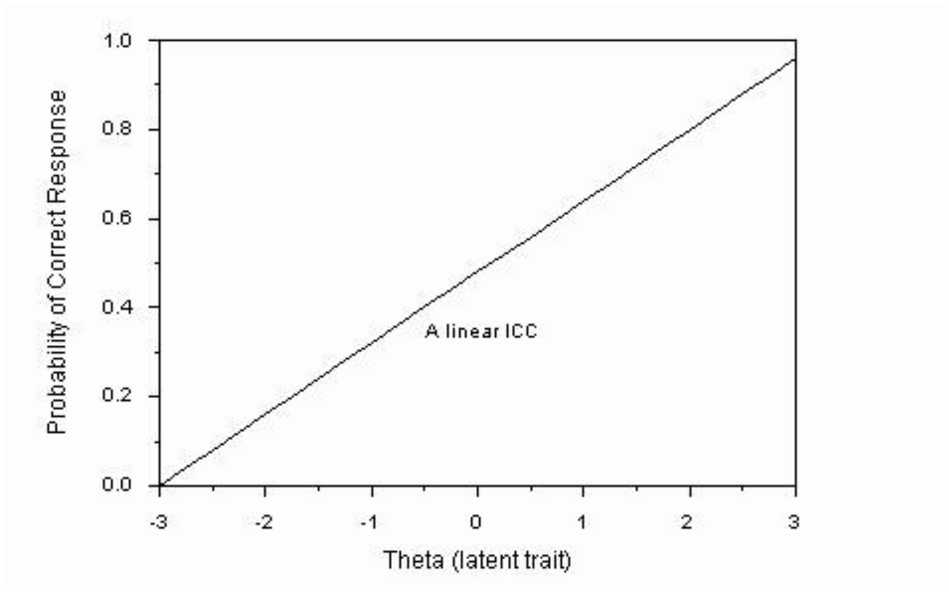


Figure 1 Relationship between trait and correct response

In IRT, the standard mathematical model for item characteristic curve is the cumulative form of the logistic function [5]. The logistic function was first widely used in biological sciences modeling growth of plants and animals. Because of its simplicity it found its way to IRT and is the preferred model. Figure 2 shows the logistic curve. Equation 1 is the most general form of the three-parameter logistic model incorporating difficulty (b), discrimination (a) and guessing factor (c).

Equation 1 IRT probability of answering a question correctly

$$P = c + (1 - c)\,\frac{1}{1 + e^{-a(\theta - b)}}$$

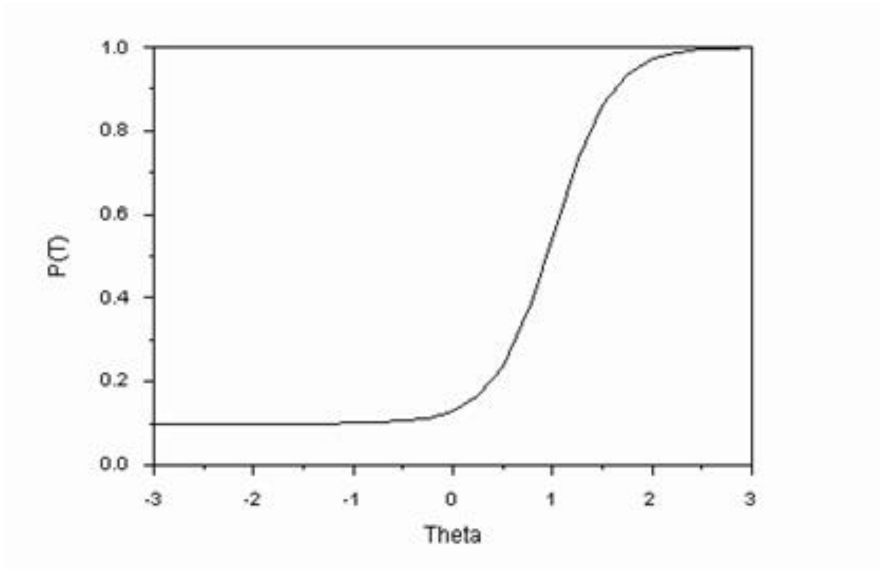


Figure 2 Logistic Curve

In IRT, test items are selected to yield the highest information content about the examinee by presenting items with difficulty parameter values that are closer to his ability value. This reduces time by asking fewer and relevant questions rather wider range ones while satisfying content considerations such as items or rules that are critical for a decision of access or scoring.

In IRT, the most important item parameter is the difficulty parameter (usually denoted by *b*). Figure 3 shows an ICC with different values of difficulty. It can be seen that the harder items require higher ability to get it right.

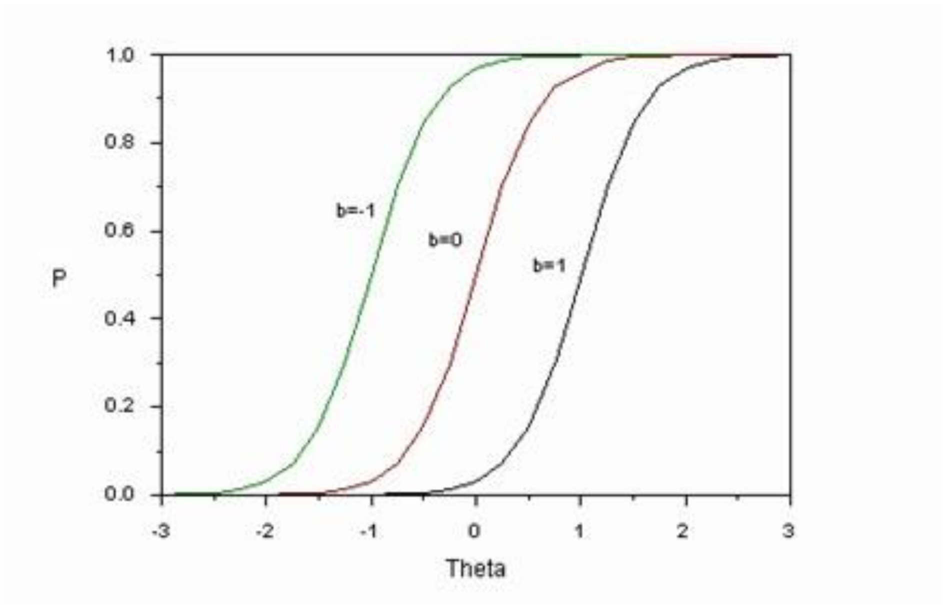


Figure 3 ICC with difficulty parameter (b)

Other IRT models use the discrimination parameter (usually denoted *a*), which represents the ability of an item distinguish between a low-ability and high-ability responder. The effect of the discrimination parameter (*a*) on ICC is illustrated in Figure 4.

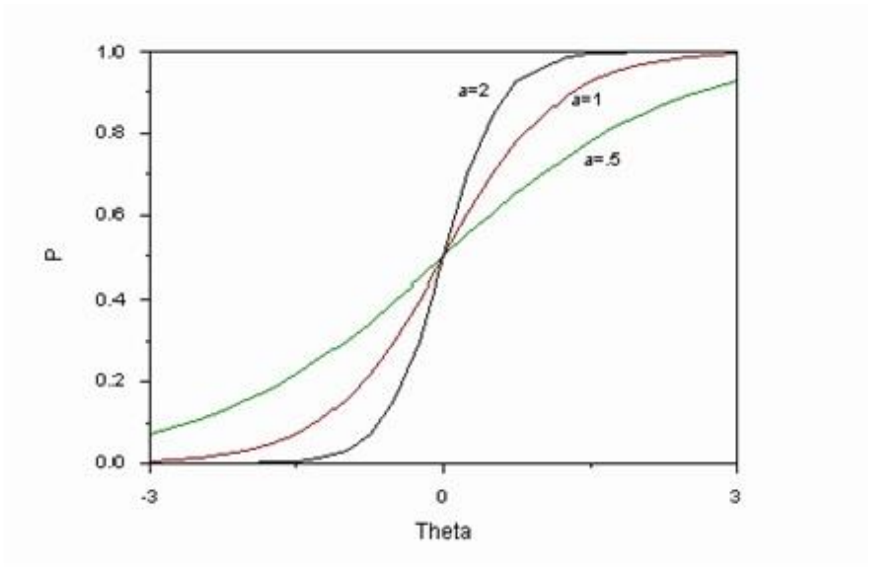


Figure 4 ICC with discrimination parameter (a)

IRT takes guessing into account and implements a guessing factor (usually denoted *c*). In any item-response there is a possibility for the responder to come up with an answer that is not based on his ability, but simply by guessing. The direct effect of this is that it raises the ICC curve a little for lower ability responders. Figure 5 illustrates an ICC with guessing.

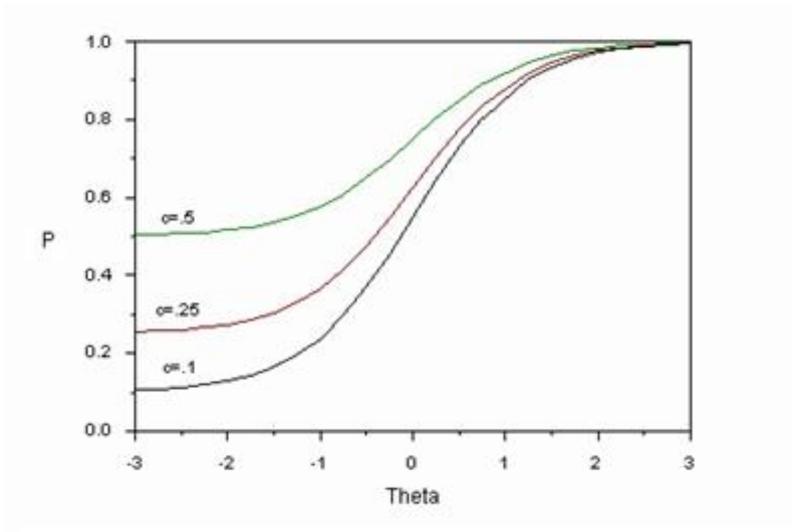


Figure 5 ICC with guessing parameter (c)

2.2.1 IRT parameter estimation

In order to determine the difficulty and discrimination parameters of a test item, IRT uses Bayesian estimates, maximum likelihood estimates (MLE) or similar methods [1, 4]. In the original IRT, an experiment is conducted to estimate these values for each item and at an assumed level of ability for various groups with associated values of IRT parameters using his judgment and experience. In my work, I model attributes in the XACML policy rules as test items and rely on the policy administrator to provide the estimated probabilities.

2.2.2 IRT ability estimation

In IRT, responses to questions are dichotomously scored. That is, a correct answer gets a score of "1" and an incorrect answer gets a score of "0". The list of such results consist an item response vector. For estimating the examinee's ability, IRT utilizes maximum likelihood estimates using an iterative process involving a priori value of the ability, the item parameters and the response vector as shown in equation (2). Here, $\hat{\theta}_s$ is the estimated ability within iteration s. a_i is the discrimination parameter of item i, i=1,2,..,N. u_i is the response of the examine (1/0 for correct/incorrect). $P_i(\hat{\theta}_s)$ is the probability of correct response from equation (1). $Q_i(\hat{\theta}_s)$ is the probability of incorrect response = 1- $P_i(\hat{\theta}_s)$ [5, 6].

Equation 2 IRT ability estimation

$$\hat{\theta}_{s+1} = \hat{\theta}_s + \frac{\sum_{i=1}^{N} -a_i \left[u_i - P_i(\hat{\theta}_s)\right]}{\sum_{i=1}^{N} a_i^2 \; P_i(\hat{\theta}_s)\, Q_i(\hat{\theta}_s)}$$

Then, the ability estimate is adjusted to improve the computed probabilities with the examinee’s responses to items. This process is repeated until the adjustment becomes small enough so that the change becomes negligible. Then the result is considered an estimate of the examinee’s ability parameter and the estimation procedure stops. The ability or trait usually ranges from -∞ to +∞, but for computational reasons acceptable values are limited to the range [-3, +3].

The ability estimate produced also comes with a standard error (SE) value that is a measure of the accuracy of the estimate. Equation (3) presents the formula used for standard error calculation [5]. The standard error can serve as a stopping criteria for a test or a dialog as we will see in the next chapter.

Equation 3 IRT standard error

$$SE(\hat{\theta}) = \frac{1}{\sqrt{\sum_{i=1}^{N} a_i^2 P(\hat{\theta}) Q(\hat{\theta})}}$$

2.3 Interactive Voice Response Systems

The main purpose of an Interactive Voice Response (IVR) system is to interact with humans using a voice stream. An IVR environment consists of a markup language to specify voice dialogues, a voice recognition engine, a voice browser and auxiliary services that allow a computer to interact with humans using voice and Dual Tone Multi-Frequency (DTMF) tones with a keypad enabling hands-free interactions between a user and a host machine [23]. Recently, many applications such as auto attendant, satellite navigation, and personal assistants such as: Apple's Siri [41], Google's Voice [42], Microsoft's Speech [43], etc., have started using IVR systems. The W3C Voice Group is the responsible body for voice standards and technologies.

2.3.1 VoiceXML

I use the IVR language VoiceXML, sometimes abbreviated as VXML [24]. Briefly, Voice XML is a Voice Markup Language (comparable to HTML in the visual markup languages) developed and standardized by the W3C's Voice Browser Working Group to create audio dialogues that feature synthesized speech, digitized audio, recognition of spoken and (DTMF) key inputs, recording of spoken input, telephony, and mixed initiative conversations.

2.4 Policy and XACML

Access control policies specify which subjects may access which resources under which conditions [4]. Being an attribute-based access control policy, XACML specifies subjects, objects and resources using some attributes. For example, a verified combination of (ID, password) pair or some accepted biometric characteristics may be specified as subject attributes. In my work, I use the OASIS standard XML-based eXtensible Access Control Markup Language (XACML) for specifying access control policies [35, 37]. A sample policy rule used in my work to allow a person who has an e-mail of “nick@gmu.edu” to access any resource is shown in Figure 6.

```
<Policy
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
PolicyId="urn:oasis:names:tc:example:SimplePolicy1"
RuleCombiningAlgId="identifier:rule-combining-algorithm:deny-overrides">
<Description>
 GMU VSE Example Secure resource access control policy
</Description>
<Target/>
<Rule RuleId="urn:oasis:names:tc:xacml:2.0:example:SimpleRule1" Effect="Permit">
<Description>
   A subject with an e-mail nick@gmu.edu can perform any action on any resource.
  </Description>
<Target>
<Subjects>
<Subject>
<SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:rfc822Name-match">
<AttributeValue DataType="urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name">
       nick@gmu.edu
      </AttributeValue>
<SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-email"
DataType="urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name"/>
</SubjectMatch>
</Subject>
</Subjects>
</Target>
</Rule>
```
Figure 6 A sample XACML policy rule

2.5 Ontology and Reasoning

An ontology is a specification of a conceptualization [27]. The W3C OWL 2 Web Ontology Language (OWL) is a Semantic Web language designed to represent knowledge about entities, collections of entities and relations between entities [31]. OWL is a computational logic-based language such that knowledge expressed in OWL can be reasoned with using software based proof systems either to verify the consistency of that knowledge or to make implicit knowledge explicit. OWL documents, known as ontologies, can be published in the World Wide Web and may refer to or be referred from other OWL ontologies. OWL is part of the W3C's Semantic Web technology stack, which includes RDF [28] and SPARQL [29].

2.5.1 Reasoning

A reasoner is a key component for working with OWL ontologies. In fact, virtually all querying of OWL ontology (and imported closures) should be done using a reasoner. This is because knowledge in ontology might not be explicit and a reasoner is required to deduce implicit knowledge so that the correct query results are obtained [30]. Some of the most popular reasoners are: FaCT++, JFact, HermiT, Pellet, RacerPro. In addition to making inference, a reasoner is able to explain how that inference happened and provide the axioms used to reach that inference. I will take advantage of this to implement IRT in ontologies.

2.5.2 Contextual Reasoning

A context-aware application is an application that takes into account the context under which it runs. The behavior of the application will depend on factors like: social, physiological, biometric, environmental, hardware, computational, temporal, activity, identity, location, etc. [22].

Contextual reasoning provides the ability to reason or infer contextual information from context models using ontological reasoners. A reasoner can infer implicit facts from asserted contextual ones that affect the behavior of the application. This helps with application customization and with generation of run-time configurations and interfaces based not only on explicit, but also on implicit context [18, 19, 20]. More about contextual reasoning will be presented in chapter 5.

2.6 Related Work

Massie and Wijesekera were among the first to describe a system using IVR for access control [9]. Dismounted soldiers have to be authorized to be eligible to use operational information systems. Many recent works focus on issues related to XACML optimization and analysis. Our work is similar to theirs in attempting to reach a decision in the shortest time possible. But the difference is mainly in trying to reach that decision by relying only on a partial set of rules.

Marouf et al. present an adaptive reordering and clustering-based framework for efficient XACML policy evaluation by using data mining techniques to improve the performance of XACML engine evaluating very large policies [14]. However, our work is different from theirs because we make access control decisions based on a set of the rules and not on all of them.

Liu. A. et al. optimizes XACML policies to on improve the performance of an XACML PDP by numericalization and normalization of XACML Policies [15].

Numericalization is used to convert the character string policies into numbers. The authors suggest that because numerical comparison is more efficient, the process improves performance. Contrastingly, our objective is not to improve the performance of the evaluation engine but to shorten dialogs and minimize the number of attributes necessary to make a good access decision.

Liu, L. et al. presented ontology-based interactive question and answering system. They are presenting traditional question answering systems finding ways to deal with missing information by building domain knowledge and expanding queries on ontologies to search questions and answers. This is different than what we are doing because they only focus on missing information; whereas we use reasoners to infer implicit information and assign weights to this information [16].

Wang presents a domain-specific question answering system based on ontology and question templates. Ontology was employed to model the interesting domain, and properties of concepts are described by a collection of question templates. The system captures a user’s intention by matching his questions to predefined templates, and return answers corresponding to template’s query focus. In my work question selection doesn’t adhere to a manual template and totally dynamic and automated [17].

CHAPTER THREE: POLICY BASED DIALOGS FOR ACCESS CONTROL

Many commercial applications like airline reservation services and credit card payment systems use Interactive Voice Response (IVR) systems to interface with customers in order to save human labor [23]. Most IVR systems provide routine services with a statically designed set of questions. We dynamically generate human-machine dialogs appropriate for some scenarios. For example, consider replacing a security guard that controls access to a secured building. When a person arrives at the door of a secured building, the guard may want some form of acceptable identity document, or ask a few questions to authenticate the person and validate the purpose of the visit etc. and many other purposes. If the guard asks the same set of questions from all respondents, these questions will become common knowledge, and all potential mal-actors may come ready to answer them. But everyone knowing the answers to these questions does not constitute an excuse for the guard to ask a set of unrelated questions, and may even violate institutional policies. Also most accesses are governed by policies. Consequently, I propose to use a policy–based dynamically question generating IVR system to control physical accesses.

A policy-based Interactive Voice Recognition (IVR) system provides hands-free access control service to special-needs people, government, military, fire-fighters etc. [1]. But many policies for physical access control have too many rules. Consequently, blindly converting such a rule base to human-machine dialogue would result in very long conversations with many disadvantages. The first is that human users would become frustrated of being subjected to long machine driven interrogations, and thereby reducing the usability of the system. The second is that long conversations take longer time to arrive at an accept/reject decision, and likely to create long queues at entrances. In addition, having a line of people behind one person in close proximity may leak private information of person answering the questions. Also, others may quickly learn the set of questions and answers that would get them authorized, thereby gain unauthorized access.

One solution is to choose a random number of attributes and generate dialogues from them. Statistically such a process suffers from many problems. The first is that, if the questions are selected randomly following a normal (Gaussian) distribution, the probability of any user answering these questions correctly may not exceed 50%. Secondly, the set of questions generated by using a randomized approach may be unimportant or irrelevant to the requesting permissions. Thirdly, a randomized process may generate a set of questions that may not be answered by most legitimate users. Consequently, I propose to use Item Response Theory that provides the basis for selecting tests from large numbers of potential questions.

Psychmotricans in social sciences and standardized test preparation organizations such as the Educational Testing Services that administer standardized test examinations like SAT, GRE, GMAT etc. have developed methodologies to measure an examinee’s trust or credibility from answers provided to a series of questions. In traditional tests, the ability of the examinee is calculated by adding up the scores of correct answers.

Currently, Computerized Adaptive Testing (CAT) that relies on Item Response Theory (IRT) has been used to better estimate an examinee's ability. It has also been shown that the use of CAT/IRT reduces the number of questions necessary to reach a credible estimation of the examinee's ability by 50%. CAT/IRT can be used to control the number and order of questions to be generated based on examinee's previous answers [7, 8].

I use CAT/IRT in more than one way. The first way is that IRT enables policy administrators to give more weight to more relevant or mandatory rules making it possible to generate questions based on their importance. Secondly, the confidence in estimating of the user's ability is much more than 50%, which is a direct result of the previous benefit. Third, the number of questions necessary to estimate the ability is reduced because the most important and relevant questions are asked. Fourth, when there are fewer questions, less time is needed to reach accept/reject decision. Fifth, using the best set of questions suitable to the user, the next user in line will not get the same set of questions. Sixth, this method of generating questions also minimizes item exposure for people in line listening and learning questions and answers.

The rest of the chapter is written as follows. Section 2 describes the basis of the work. Section 3 describes the main contribution of using IRT to dynamically generate dialogues from policies. Section 4 describes the architecture and the implementation. Section 5 concludes the chapter.

3.1 Using IRT to Manage and Control Dialogues

I use Item Response Theory to manage and control dialogue questions generated from a large pool of policy rules in a way that shortens the length of dialogues while keeping the maximum accuracy in estimating the user's trust. The IRT-based estimated (θ) represents the trust or confidence of the system in the person answering the questions in order to make an access decision given that subject attributes in a policy rule are treated as a test question.

In order for my approach to succeed, I need to differentiate policy rules using IRT parameters. In order to do so, I have extended XACML to comply with IRT requirements, by adding a difficulty (b), discrimination (a) and guessing (c) parameters to every policy rule in XACML. We do so by proposing a Profile for Physical Access Control in XACML. Figure 7 shows our IRT extension to the rule shown in Figure 6.

```
<Rule RuleId="urn:oasis:names:tc:xacml:2.0:example:SimpleRule1">
<IRT>
      <Difficulty>-1.0</Difficulty>
      <Discrimination>1.0</Discrimination>
      <Guessing>0.2</Guessing>
      </IRT>
</Rule>
```

Figure 7 Example XACML IRT extension

In this solution, the parameter values are hard-coded and it is assumed that the initial values are determined by the owner or administrator of the policy. In this implementation and for testing purposes we use a default value of zero for c, which practically neutralizes it. The values of the other two parameters are decided by the administrator.

The solution developed is based on the IRT two-parameter model, which relies on the item's difficulty and discrimination parameters. While my implementation of the IRT algorithm is estimating the ability from these two parameters, it can be easily expanded to accommodate the three-parameter model, which adds the previously neutralized guessing parameter or downgraded to use the one-parameter (Rasch) model, which only uses the difficulty parameter [5]. Figure 8 shows algorithm to estimate ability based on equations 1 and 2.

```
Algorithm 1: IRT Ability estimation
Input: a priori theta, Difficulty, Discrimination, Answer
Output: posteriori theta, standard error
/* calculate theta and standard error*/
1: for (counter < items.length) do
2:    itemDifficulty=parseFloat(difficultyArray[i]);
3:    itemDiscrimination=parseFloat(discriminationArray[i]);
4:    answer=parseFloat(answerArray[i]);
5:    probTheta=calculateProbability(itemDiscrimination,
      aTheta,itemDifficulty); // equation 1
6:    thetaSplus1= claculateTheta(probTheta, thetaS); //equation 2
7: endfor;
8: estimatedTheta = thetaSplus1;
9: return thetaSplus1;
```

Figure 8 IRT estimation algorithm

My system estimates the ability of a user after every answer before selecting and asking the next question. When the ability estimation reaches a predefined level, the system conveys the decision at the end of the dialogue. Consequently, the resultant decision is based on the IRT characteristics of the rule and not on the number or the percentage of correctly answered questions. The ability estimate produced by my implementation also comes with a standard error (SE) value that is a measure of the accuracy of the estimate. Equation (3) presents the formula used for standard error calculation [5].

Higher standard error indicates that the estimate is not very accurate, while lower values indicate higher confidence in the estimation. This too can be used as a means to discontinue the dialogue or use an alternate decision method.

3.1.1 Benefits of using IRT in dialogues

By using IRT to solve the problem of management and evaluation of the policy-based dialogue have the following benefits:

- The intermediate values of the ability estimate and the standard error are used as stopping criteria of the dialogues generating shorter dialogues, instead of the fixed length dialogues.
- Enables the control on order of questions to be asked. The most relevant and important ones, based on their IRT values, can be presented first saving time and leading to faster ability estimation. Most likely, higher ability users will answer harder questions quickly to avoid longer conversations, and others will not need too much time to be disqualified.
- Questions are generated based on the item (rule) parameters and the dynamic estimation of the requester's ability level. Questions generated wouldn't always be the same as the case in a traditional IVR system achieving an important benefit of good distribution and randomization.

- The previous benefit also leads to indirectly improving privacy, especially in a situation where other people are in close proximity to the person interacting with the system. The controlled randomization can produce different questions, but with same difficulty level. So, a bystander person cannot just repeat or record answers to gain access.
- Such a system can compensate for an attribute that is inaccurately recognized because the decision depends on accumulative ability estimation and not on mathematical sum of correct answers.
- The system would be able to reach a decision with a pre-specified accuracy level without asking all the possible questions that can be generated from the policy rules.

This provides a new approach to policy evaluation by accepting some risk in making an access control decision, which might be an interesting area of research.

3.2 The Policy-Based IVR System for Access Control

I integrated two systems that extract attributes from the policy and dynamically generate dialogues.

3.2.1 Architecture

Figure 9 shows the standard XACML data-flow diagram with our IRT extension on the side.

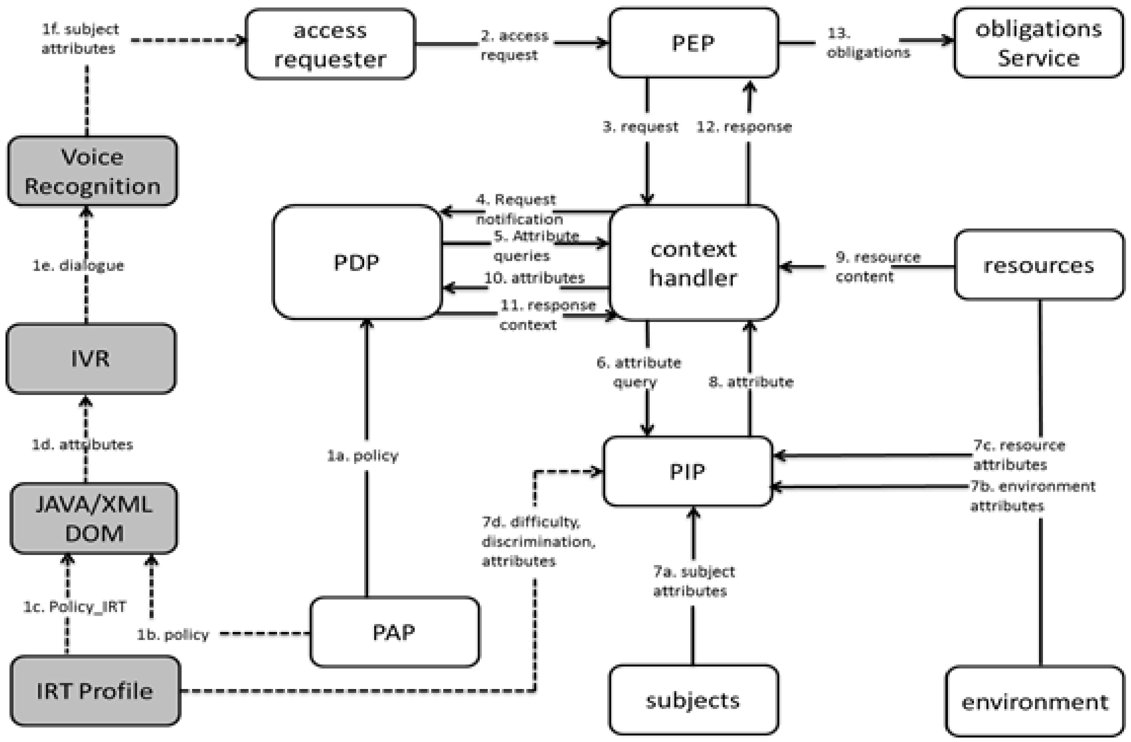


Figure 9 The XACML data-flow with the IRT extension

The flow of data in the XACML architecture is as follows: (1) PAPs write policies and policy sets and make them available to the PDP. These policy sets represent the complete policy for a specified access target. (2) The access requester sends a request for access to the PEP. (3) The PEP sends the request for access to the context handler in its native request format, optionally including attributes of the subjects, resource, action, environment and other categories. (4) The context handler constructs an XACML request context, optionally adds attributes, and sends it to the PDP. (5) The PDP requests any additional subject, resource, action, environment and other categories attributes from the context handler. (6) The context handler requests the attributes from a PIP. (7) The PIP obtains the requested attributes. (8) The PIP returns the requested attributes to the context handler. (9) Optionally, the context handler includes the resource in the context. (10) The context handler sends the requested attributes and (optionally) the resource to the PDP. The PDP evaluates the policy. (11) The PDP returns the response context (including the authorization decision) to the context handler. (12) The context handler translates the response context to the native response format of the PEP. The context handler returns the response to the PEP. (13) The PEP fulfills the obligations. (14) If access is permitted, then the PEP permits access to the resource; otherwise, it denies access [35]. Our XACML-IRT extension to the data-flow is as follows:

1b. the policy is read and parsed by the JAVA/XML/Document Object Model (DOM)

1c. Policy-IRT profile extension is parsed by the JAVA/XML/ Document Object Model (DOM)

1d. extracted attributes (Id, value, difficulty, discrimination) are passed to the IVR system.

1e. IVR generates a dialogue and collect attribute values from the subject.

1f. the voice recognizer converts the successfully recognized utterances to text, formatted as a standard XACML request.

The XACML IRT profile maintains the IRT attributes of the policy rule and is referenced by the PIP for these attributes. An example of these attributes is shown in Figure 7. For the dialogue generation, the XML-based policy rules in the PAP and the IRT profile are both parsed by the Java/XML Document Object Module (DOM), which reads all these attributes and passes them to the IVR system [36, 37]. Figure 10 shows a sample request generated by our dialogues.

```
</Request>
<Subject>
        <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id">
                <AttributeValue> Nick </AttributeValue></Attribute>
        <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-email">
                <AttributeValue> nick@gmu.edu </AttributeValue></Attribute>
        <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-role">
                <AttributeValue> Professor </AttributeValue></Attribute>
        <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-department">
                <AttributeValue> Computer Science </AttributeValue>
        </Attribute>
</Subject>
<Resource/>
<Action/>
<Environment/>
</Request>
```
Figure 10 A Sample XACML Request Generated from a dialogue

The developed system uses a basic PDP, PEP and PIP to show the IRT-extension integration. The PDP is capable of producing a decision of grant or deny access based on rule's IRT parameters and interactive and dynamic answers of the user.

The example XACML rules are concerned of subject's authorization to have access to a physical facility, which is a building with departments and offices. The rules see if someone with name, e-mail, role and department is granted access. These rules are located in the original XACML policy file. The IRT parameters such as difficulty and discrimination are located in the XACML-IRT extension file. The link between the two files is the "Rule Id".

3.2.2 Implementation

I use the Voxeo's Prophecy local server as the voice platform for voice recognition and to run the dialogues. I also use Java, Java Server Pages (JSP), and Java Script (JS) to implement the architecture modules, and communicate between the XACML-PDP (for evaluating access control policies), the IVR module (for asking questions and gathering answers) and the IRT implementation to estimates the user's ability/trust scores.

3.2.3 Voice Platform (Voxeo)

Voxeo's Prophecy is a comprehensive IVR and standards-based platform [25]. Some of the capabilities integrated into the platform are: automatic speech recognition, speech synthesis (Text-to-Speech), Software Implemented Phone (SIP) browser and libraries to create and deploy IVR or VoIP applications using VXML and CCXML. It supports most of server side languages, application servers and has a built-in web server.

3.2.4 Inter-operation between the Three Runtimes

The dialogue starts when the system is engaged by a person in front of the building gate. The conversation starts with a menu in VoiceXML hosted on the local Voxeo Prophecy web server. The voice browser connects to the web server and converts text to speech and speech to text. Figure 11 shows a sample VoiceXML code.

```
<form id="Begin">
 <block>
          <prompt bargein="true">
                    Welcome to G M U Volgenau School of Engineering
          </prompt>
          <prompt baregin="true">
                    Only authorized people can have access to this facility
          </prompt>
          <assign name="xacmlResource" expr="'V S E Main Gate'"/>
          <goto next="#Resource"/>
 </block>
</form>
```
Figure 11 VoiceXML main form

A script embedded in the VoiceXML page fetches the policy, parses the policy into DOM, iterates through the rules one by one to extract the subject's attribute names and their value and populate an array of questions and answers. Attribute (name, value) pairs are passed to VXML to dynamically convert to questions. The system now waits for the user's answers. If provided the VXML voice recognition engine converts the answer to text. This textual value is matched with the attribute value from the rule. If it matches a value of "1" is inserted in the response vector and if it doesn't a value of "0" is inserted.

After collecting all required values, the resultant vector of answer pairs, a priori θ, the difficulty and discrimination parameters from rules are used to estimate a posteriori θ, and the standard error (SE). This θ, is the estimated user's ability and is compared to a threshold value specified by the policy administrator, say θ'. Access is granted if $(\theta > \theta')$ and denied otherwise. Figure 12 shows my algorithm integrating policy, IVR and IRT.

```
    Algorithm 2: dialogue access decision evaluation
    Input: a priori theta, Difficulty, Discrimination, Answer
    Output: access control decision
    /* make an access control decision based on policy*/
1:  domDoc=parse(policy); // use DOM and XPath to locate nodes
2:  irtDoc=parse(irt_profile);
3:  difficultyArray=IRT_node.difficulty;
4:  discriminationArray=IRT_node.discrimination;
5:  attribArray=”//Subject/SubjectMatch/SubjectAttributeDesignator@attributeId”;
6:  valueArray=”//Subject/SubjectMatch/AttributeValue”;
7:  /*use voiceXML and JSP to generate the dialog*/
8:  for (counter < items.length) do
9:     <vxml:Prompt> “What is” + attribArray[i] + “?”;
10:    <vxml:Field>= user_utterance;
11:    response[i] = Field.voiceRecognition(user_utterance);
12:    if response[i]= valueArray[i]
13:       resultVector[i]=1;
14:    else
15:       resultVector[i]=0;
16: endfor;
17: theta = IRT_algorithm(resultVector, difficulty, discrimination, aPrioriTheta);
18: if theta > thetaThreshold
19:    permit;
20: else
21:    deny;
```

Figure 12 Policy-IVR-IRT algorithm

Recognizable utterances have to conform to a grammar [40]. In this example, I used a simple hard-coded inline grammar, as shown in Figure 13. In a next phase, this grammar would be generated automatically as “grxml”.

```
<field  modal="true">
     <prompt>  What is your<%out.println(subjectAttributeValue);%>
     </prompt>
        <grammar xml:lang="en-US" root = "MYRULE">
            <rule id="MYRULE" scope = "public">
            <one-of>
                 <item> Nick Clark</item><item> Nick</item>
                 <item> nick@gmu.edu </item><item> Professor </item>
                 <item> Computer Science </item><item> Student </item>
            </one-of>
        </grammar>
</field>
```

Figure 13 VoiceXML grammar

3.3 Summary

I have designed and implemented a system that can dynamically generate efficient interactive voice dialogs for physical access control from XACML polices. I have used IRT to generate shorter dialogues between the system and a human speaker. It is also useful in compensating for inaccurate voice recognition of attributes collected during dialogs. The access control decisions are made on an estimated level of trust based on the importance or relevance of rules in the access control polices. This approach also enables the reordering of questions with the purpose of preserving privacy in IVR systems. Using this new approach in IVR systems will enable and facilitate new types and usage of access control technology in enterprise systems and make them more ubiquitous and match advancements in mobile, cloud and voice technologies.

CHAPTER FOUR: ONTOLOGY BASED DIALOGS FOR ACCESS CONTROL

Physical control points such as human guarded gates, border control points and visa counters provide entry into facilities or geographical regions to those that can be admitted legitimately. Legitimacy is usually determined by rules, regulations or policies known to entry control personnel whose duty is to ensure that these policies are enforced while admitting people. In order to do so, they hold an interview, in which an aspiring entrant is asked a series of questions, and possibly show some documents and demonstrate some knowledge about the contents of the documents or attributes contained in them. Successful interviews should have questions that are relevant, of a reasonable level of difficulty (i.e. not too difficult or common knowledge) and not to have been asked in prior interviews for the same purpose without drawing accusations of bias from rejected entrants. Ideally, a successful interview should accommodate differences in accents and provide assurance that it is unbiased against similar attributes.

Given the recent success of interactive voice response (IVR) systems such as auto attendants, satellite navigation, and personal assistants such as Apple’s Siri, Google’s Voice, Microsoft’s Speech, I investigated the possibility of specializing IVR systems for access control such as: Visa interviews, entry point interviews, biometric enrollment interviews, password reset, etc.

Although IVR systems have come a long way in recognizing human voice, and responding to human requests as if responses come from another human, most of the existing IVR systems are pre-programmed with questions and their acceptable answers, and consequently have limited capability in satisfying the Use Case at hand.

The first minor limitation of current IVR systems comes from the fact that, the human starts and drives the conversation. The second limitation is that most IVR systems have a finite number of pre-programmed conversations. Therefore the set of questions generated by such a system are the same for every conversation. This limitation may expose the set of questions so that aspiring entrants may come with prepared question-answer pairs, even if the subject matter of the questions may be unfamiliar to them. Consequently, having the ability to select questions from a large pool may resolve this limitation. The third limitation is that when selecting a random set of questions from a large pool, the set of questions asked may not have the desired overall level of difficulty to challenge the user. Solving this issue is relevant because all aspiring entrants expect to have a fair interview. The forth limitation is that questions must be able to discriminate between someone that knows the subject matter from someone who guesses an answer.

As a solution I created an ontological inference based IVR system that uses item response theory (IRT) to select the questions [23, 5]. The system uses the XACML language as a base to establish entry policies that consist of rules to specify the attributes that must be possessed by permitted entrants [35]. The IVR system has the responsibility of determining access by asking questions generated using ontological inferences and IRT.

In previous chapter, I introduced a policy-based IVR system for use in access control to resources and an enhancement that uses IRT to select queries from a large set of attributes present in a policy. Here I introduce ontology-aided access control system by including questions related to the base attributes in order to ascertain the interviewee's familiarity, and provide a score for the entire set of answers. I do so by aligning each attribute with an ontology that encodes the subject matter expertise on that attribute and derive facts from these ontologies using reasoners to generate questions. I then assign weights to these derivations based on the axioms and rules of derivations used in the proof tree.

Usually ontologies have a large number of axioms and assert even more facts when using reasoners. Consequently, blindly converting such an axiom base to human-machine dialogue would result in very long conversations with many disadvantages. The first is that human users would become frustrated of being subjected to long machine driven interrogations, and thereby reducing the usability of the system. The second is that long conversations take longer time to arrive at an accept/reject decision, and likely to create long queues at points of service, such as Airports and guarded doors. In addition, having a line of people behind one person in close proximity may leak private information of the interviewee. Also, others may quickly learn the set of questions and answers that would get them mistakenly authorized, thereby gaining unauthorized access.

My goal in this work is to demonstrate and build an access control system using dialogues of questions and answers generated from a suitable collection of ontologies. Table 3 shows a sample dialogue that is generated for Use Case 2.

Table 3 Use Case 2 Sample dialog

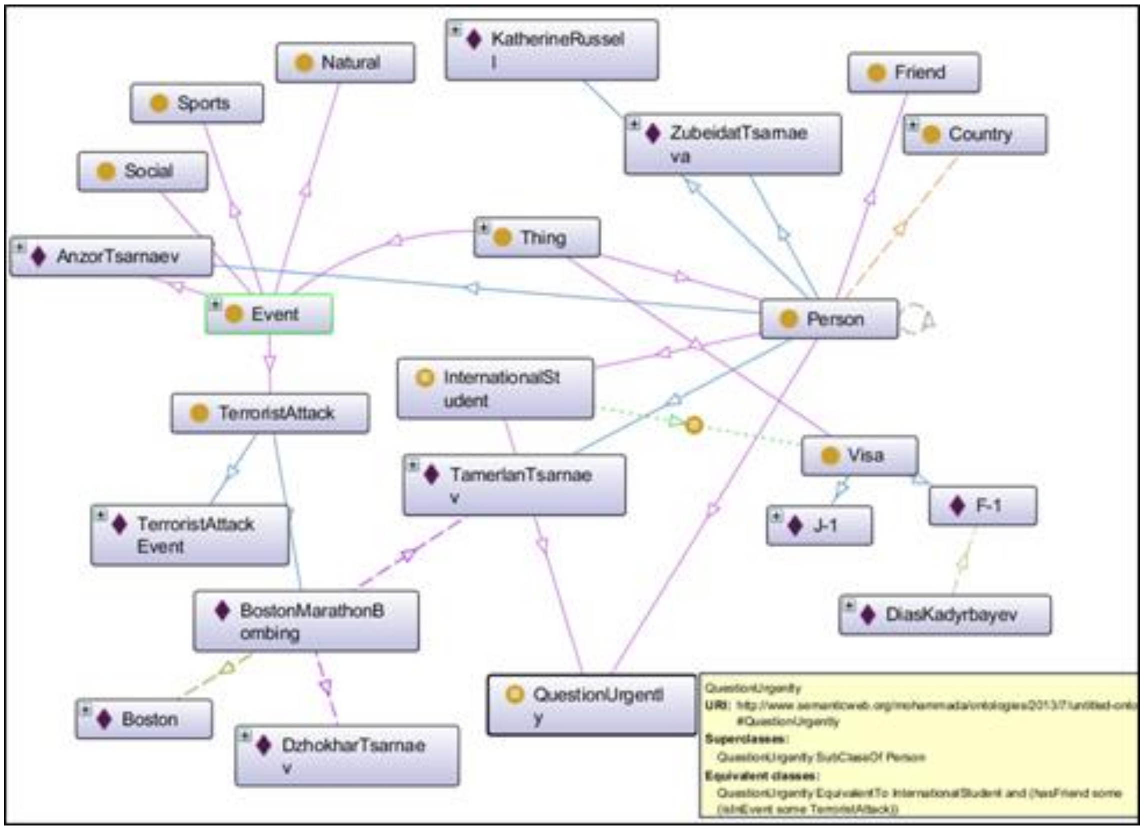
	Question/answer	IRT Question Difficulty	Ability estimate (All answers correct)	Ability estimate (All answers wrong)
			1.0	1.0
System	What is your name	1.0		
User	Dias Kadyrbayev		1.1	0.9
System	How old are you?	1.5		
User	19		1.25	0.85
System	I see you have visited Russia, which part?	1.75		
User	Dagestan		1.4	0.75
System	Was there any terrorist attacks while you were there?	2.0		
User	No		1.75	0.5
System	Do you know Dzhokhar Tsarnaev?	3.0		
User	Yes		2.5	0.0
System	Where is he?	2.5		
User	He is in Georgia		2.75	-0.75
System	Are you still attending UMass Dartmouth?	2.0		
User	Yes		3.0	-0.5

The prototype automated IVR system can help immigration enforcement at a border control point making a decision to permit or deny a person asking for entry. Through a dialogue of questions and answers, the interviewee will be assigned a numerical score that will then serve as a threshold in the decision making process. This score is calculated using IRT, which takes into account the correctness of the user’s responses and the weight of the individual questions.

The rest of the chapter is written as follows. Section 2 describes an ontological use case. Section 3 describes the response theory. Section 4 describes the system architecture. Section 5 describes our implementation. Section 6 is about results and summary.

4.1 Motivating Use Case

In this section, a description of an example ontology used in Use Case 2 to generate efficient dialogues of questions and answers that are used in assigning a numerical value to an interviewee's ability or trust level. Figure 14 illustrates a class diagram of example ontology for homeland security.



QuestionUrgently
URI: http://www.semanticweb.org/mohammada/ontologies/2013/7/untitled-onto
#QuestionUrgently
Superclasses:
QuestionUrgently SubClassOf Person
Equivalent classes:
QuestionUrgently EquivalentTo InternationalStudent and (hasFriend some (isInEvent some TerroristAttack))

Figure 14 The Homeland Security Ontology in Protégé

The purpose of this ontology is to collect, organize and infer information that can help deterring possible attacks, enforcing strict entry and enabling faster reach to suspects. The ontology defines classes, individuals, properties and relationships using OWL Ontology Language. The major entities in the ontology are:

- **Person:** defines humans in general and has subclasses like; International Student and Friend.
- **Event:** defines an event that has a location, date, time and type like terrorist attack
- **International Student:** is a person who is on an F-1 or J-1 Visa type

- **University:** defines a university. Some of its current members are MIT and GMU
- **City:** defines a city like Boston
- **Country:** defines a country like USA, Russia, Dagestan, Kazakhstan, etc.
- **State:** defines a state like Massachusetts
- **Visa:** defines visa types like F-1 and J-1 student visas and maybe others.

This ontology represents many kinds of data classes and relationships between these major classes and individuals. For example, we define the “Boston Marathon Bombing” as a “Terrorist Attack” that happened in “Boston”, which is a city in “Massachusetts” state. Another fact is that “Dzhokhar Tsarnaev” is an “Event Character” in the “Boston Marathon Bombing” “Terrorist Attack”. Also we have an “International Student” who is a friend to “Event Character” in the “Boston Marathon Bombing”.

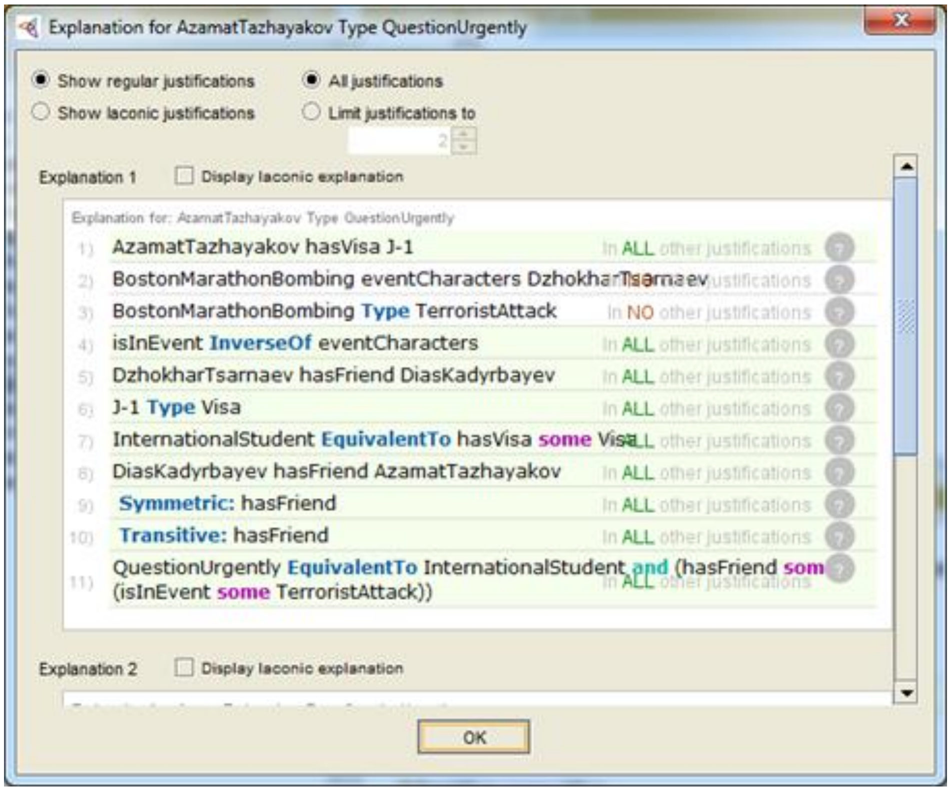
I use this ontology in my work because it serves as a good example showing the strength of the system. First, it shows the possibility of generating valuable questions from asserted or inferred facts. Second, it enables the implementation of the theory under consideration to generate efficient and secure dialogs that are used in: (1) making entry control decisions, (2) assigning numerical values to ability or trust in the shortest time possible and (3) load distribution among interviewers and diverting people for further investigation.

The use of ontology in such an application provides many benefits. The most important amongst them is reasoning. Using a reasoner we are able to derive facts from asserted ones. These facts are used to generate questions to measure the knowledge or ability level of an interviewee on a subject under questioning. In IRT, better item selection and ability estimation happens when a large set of items is available to draw questions from. Using ontology, the large number of derivable facts provides us with the ability to increase the number of questions, and also control the quality and difficulty of questions.

Although there are many reasoners such as FaCT++, JFact, Pellet, RacerPro, I use HermiT [30] in my work. Given an OWL file, HermiT can determine whether or not the ontology or an axiom is consistent, identify subsumption relationships between classes and deduce other facts. Most reasoners are also able to provide explanations of how an inference was reached using the predefined axioms or asserted facts.

One such fact derived from asserted ones in our ontology, is finding the friends that hold a student visa of a person involved in a terrorist attack. To explain this, I have “dzhokhar is friend of Dias”, “Dias is friend of Azamat”, “Dias has F-1 visa”, “Azamat has a J-1 visa”, “dzhokhar is an “Event Character” in the “Boston Marathon Bombing”, “Boston Marathon Bombing” is a “Terrorist Attack”. Thus the system infers (using the HermiT reasoner) that Azamat and Dias are the friends of the Boston Bomber and therefore need to be questioned at any entry point. I use this chain of derivations to generate specific questions from them.

Reasoners and the explanations that they provide are very important components in this work to generate relevant and critical questions from ontology that measure knowledge and estimate ability from a response in order to grant access or assign trust. In the example above, the reasoner provided an explanation of the inference using 11 axioms. My system uses such a number in defining the difficulty of questions generated from such inferences. Figure 15 shows the HermiT reasoner’s explanation of our inferred fact.


Explanation for AzamatTazhayakov Type QuestionUrgently

[● Show regular justifications] ○ Show laconic justifications

[● All justifications] ○ Limit justifications to 2

Explanation 1 ☐ Display laconic explanation

Explanation for: AzamatTazhayakov Type QuestionUrgently

1)	AzamatTazhayakov hasVisa J-1	In ALL other justifications
2)	BostonMarathonBombing eventCharacters DzhokharTsarnaev	In ALL other justifications
3)	BostonMarathonBombing Type TerroristAttack	In NO other justifications
4)	isInEvent InverseOf eventCharacters	In ALL other justifications
5)	DzhokharTsarnaev hasFriend DiasKadyrbayev	In ALL other justifications
6)	J-1 Type Visa	In ALL other justifications
7)	InternationalStudent EquivalentTo hasVisa some Visa	In ALL other justifications
8)	DiasKadyrbayev hasFriend AzamatTazhayakov	In ALL other justifications
9)	Symmetric: hasFriend	In ALL other justifications
10)	Transitive: hasFriend	In ALL other justifications
11)	QuestionUrgently EquivalentTo InternationalStudent and (hasFriend som (isInEvent some TerroristAttack))	In ALL other justifications

Explanation 2 ☐ Display laconic explanation

OK

Figure 15 The Homeland Security Ontology in Protégé

4.2 Using IRT to Manage and Control Dialogues from Ontologies

Figure 16 shows the overall architecture of our system. The system uses derived or axiomatic facts of the ontology to create questions asked by our IVR system. Given that a large number of facts can be derived from our ontology, but only few questions can be asked during an interview, we use IRT to select the facts that are used to generate questions.

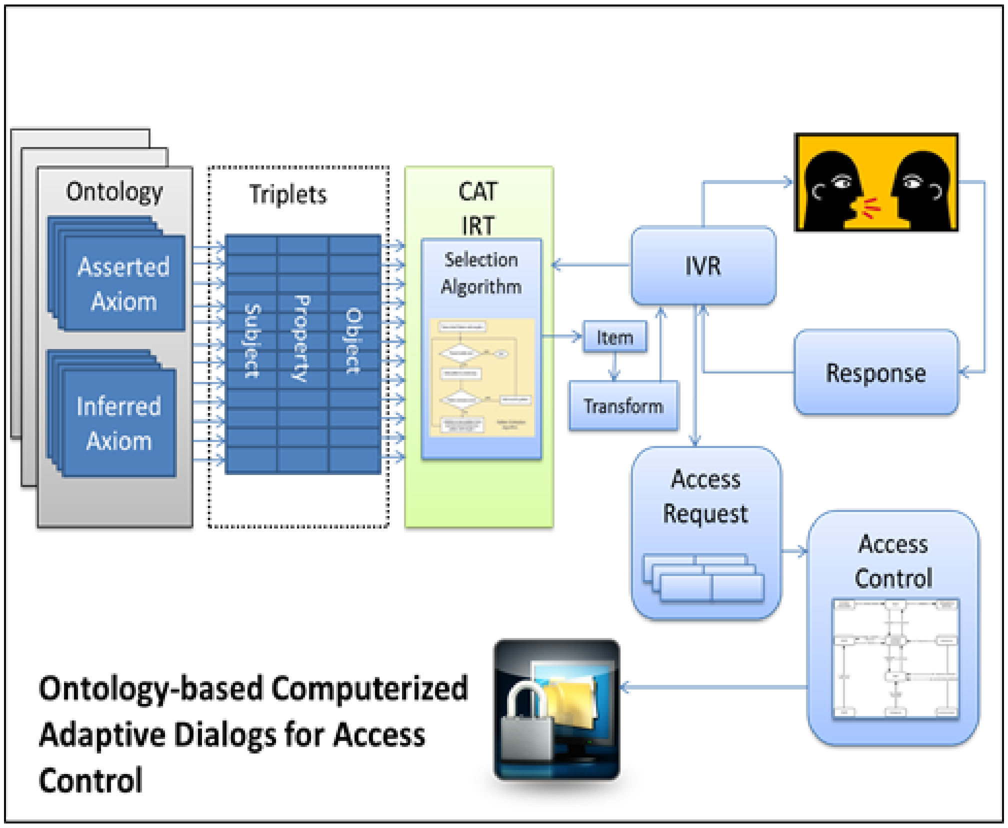


Figure 16 Ontology-based IVR using IRT

The questions are automatically created without human involvement by combing English words or phrases such as “Does” or “Is-a” with ones chosen from the ontology of (subject, property, object) triples. The expectation is a dichotomous answer of either (yes, no) or (true, false). The ontological property names such as “is-a”, “has-something” are prime candidates for creating true/false questions. The system transforms the question into VoiceXML and plays to the user. Then the system waits for the user's utterance, and if the user provides one, the system's voice recognition software attempts to recognize the input and checks the correctness of the answer. Based on the answer, the IRT estimation procedure either increases a priori ability score or decreases it. The process continues until a predetermined level of ability or accuracy specified according to the application is reached.

Because ontologies produce a large number of facts, it would be impractical to run a dialogue that lasts hours in order to estimate user's ability. My homeland security ontology uses 167 axioms. The reasoner was able to infer 94 facts raising the total number of axioms and candidate to generate questions to 273.

IRT is used to manage and control dialogue questions generated from a large pool of ontologically derived facts in a way that shortens the length of dialogues while keeping the maximum accuracy in estimating the user's trust. The IRT-based estimated (θ) represents the trust or confidence of the system in the person answering the questions in order to make an access decision.

I use OWL's annotation property to assign IRT parameters to axioms. Annotations were selected in order to keep the semantics and structure of the original ontology intact. Every asserted axiom in the ontology is annotated with IRT parameters; difficulty (b), discrimination (a) and guessing (c). All asserted axioms are assumed to have the same default degree of difficulty and discrimination values of 1. The code snippet in Figure 17 illustrates the annotation using Java with OWL API. An improvement to this approach would be to assign different values for difficulty and discrimination by using domain experts.

```
OWLAnnotationProperty irtDifficultyAP = df.getOWL
AnnotationProperty(IRI.create("#irt_difficulty"));
OWLAnnotation irtAnnotation = df.getOWLAnnotation(
   irtDifficultyAP, df.getOWLLiteral(1.0));
for (OWLAxiom axiom : axioms) {
     OWLAxiom axiom2 = axiom.getAnnotatedAxiom
        (Collections.singleton(irtAnnotation));
     manager.addAxiom(ontology, axiom2);
}
```

Figure 17 Ontology IRT Annotation

Inferred facts are weighed more during the estimation process. The parameter values are calculated from the number of explanation axioms used in each individually inferred fact. The scheme of difficulty value assignment is shown in Table 4; where higher values or weights are assigned according to the number of explanation axioms used to infer a fact, and consequently the question generated from it is considered to be more difficult than one generated from an asserted fact. Figure 18 illustrates a code snippet for inferred axiom annotation.

Table 4 IRT Difficulty Assignment Based on Number of Axioms in Explanation

Number of explanations	IRT Difficulty	
1	0	Easy
2-3	1	
4-5	1.5	Moderate
6-7	2	
8-9	2.5	
>=10	3	Hard

```
Set<OWLAxiom> inferredAxioms=inferredOntology.getAxioms();
DefaultExplanationGenerator explanationGenerator =new
DefaultExplanationGenerator(manager, factory, ontology, reasoner,
new SilentExplanationProgressMonitor());
for (OWLAxiom axiom : inferredAxioms) {
    Set<OWLAxiom> explanation =
explanationGenerator.getExplanation(axiom);
//Annotate inferred axioms using the number of explanation
OWLAxiom tempAxiom =
axiom.getAnnotatedAxiom(Collections.singleton(irtAnnotation));
manager.addAxiom(inferredOntology, tempAxiom);
```

Figure 18 Java code for inferred axiom annotation

In this work and for testing purposes I use a default value of “1.0” for discrimination and “0.0” for guessing, which practically neutralizes them leaving the difficulty parameter as the sole factor in estimating ability using equation 2. However, the solution and algorithm are based on the IRT two-parameter model, which relies on the item’s difficulty and discrimination parameters. The algorithm to estimate ability is the same one introduced in Figure 8.

The system estimates the ability of a user after every answer to a question generated from an axiom before selecting and asking the next question. If the ability estimate exceeds the threshold then access is granted. If the threshold is not reached then additional questions are offered. If the estimated ability doesn’t reach the threshold the dialog stops and access is denied. Depending on the application, the dialog might be run again giving a second chance. When the ability estimation again reaches a predefined threshold, the system concludes the dialog and conveys the decision.

The resultant decision is based on the IRT characteristics of the axiom and not on the number or the percentage of correctly answered questions as in traditional testing. Also, the ability estimate produced comes with a standard error (SE) value that is a measure of the accuracy of the estimate. Equation 3 presented the formula used for standard error calculation.

Higher standard error indicates that the estimate is not very accurate, while lower values indicate higher confidence in the estimation. This too can be used as a means to discontinue the dialogue or use an alternate decision method.

4.3 Implementing the Ontology-Based IVR System for Entry Control

A prototype of the system showing the major components using Use Case 2 is presented here.

4.3.1 Item Bank

In this work, I start with ontology, annotate every axiom with an "irt_difficulty" property of value "1". Then the HermiT reasoner is used to infer implicit axioms and their explanations. The inferred facts are themselves annotated with "irt_difficulty" property and values calculated by factoring the number of explanation axioms using the schema stated in Table 4.

For example, when annotating the inferred fact "the friends of the Boston Attack Bomber", which has an explanation that includes 11 axioms shown in Figure 15, the irt_difficulty annotation would be "3.0"; which is the highest value on the scale of IRT difficulty parameter values in Table 4. I assume that answering a question generated from a high-valued fact is a difficult task. Consequently, if the answer to a question derived from this fact is correct, the ability estimate would be impacted more positively than a correct, but easy one and more negatively if the opposite happens. An example is the asserted axiom that “Boston is located in Massachusetts”. Because this is an asserted fact, it is annotated with value “1.0”; which makes a question generated from it an easy one and thus not affecting the ability estimate greatly.

This process is basically generating the item bank in CAT/IRT terminology. Each item in the item bank contains a question, an answer and IRT parameters. In addition to saving it as ontology in any of the supported formats, this item bank can also be supported by using a more specialized CAT/IRT platform like Cambridge University’s Concerto [39].

4.3.2 Generating dialogues from ontology

The conversation starts with a menu in VoiceXML hosted on the local Voxeo Prophecy web server. The voice browser connects to the web server and converts text to speech and speech to text. Figure 19 shows a sample VoiceXML code.

```
<form id="Begin"><block>
<prompt bargein="true">
   Welcome to the United States. To accelerate
your entry, we will appreciate your responses to
some questions to verify your identity and
eligibility  </prompt>
<assign name="xacmlResource" expr="’point of
entry"/>
<goto next="#Resource"/></block>
</form>
```

Figure 19 A sample Homeland security VoiceXML greeting form

Figure 20 shows our algorithm integrating ontology, IVR and IRT. This algorithm was successfully implemented using JavaScript and Java Server Pages (JSP) embedded in VoiceXML pages.

```
Algorithm 2: dialogue access evaluation
Input: a priori theta, Difficulty, Discrimination, Answer
Output: access control decision
/* make access control decision from ontology*/
domDocument=parse(ontology); // DOM
subjectArray=getAxiomSubject(axiom);
propertyArray=getAxiomProperty(axiom);
objectArray=getAxiomObject(axiom);
difficultyArray=getAxiomDifficulty(axiom);
/*use voiceXML , JSP to generate dialog*/
for (counter < items.length) do
   <vxml:Prompt> '[auxiliary verb]' +propertyArray[i] + "
   " + objectArray[i] +" "+ subjectArray[i];
   <vxml:Field>= user_utterance;
      response[i] =
   Field.voiceRecognition(user_utterance);
      if response[i]= 'Yes' or 'true'
         resultVector[i]=1;
      else
         resultVector[i]=0;
   endfor;
   theta = IRT_algorithm(resultVector, difficulty,
   discrimination,aPrioriTheta);
   if theta > thetaThreshold
      permit;
   else
      deny;
```

Figure 20 Ontology-IVR algorithm with IRT

The main steps are as follows:

- Load the ontology and parse the XML into Document Object Model (DOM).
- Extract the axiom's triplet (subject, property, object)
- Extract the axiom's IRT difficulty value from the annotation

- Establish a VoiceXML "For" loop that synthesizes a question from string or text values to speech (TTS). The question consists of an auxiliary verb, object, property and subject to test the correctness of an axiom.
- The system waits for a response. If there is one it converts it to text and recognizes it. If it adheres to grammar then a value is assigned as an answer.
- If there was no answer then VXML re-prompts the question up to a programmed number of times. If exceeded then an appropriate VXML is executed.
- The vector of binary answers is used to estimate the IRT ability.
- The loop continues until a threshold of θ or the maximum number of questions is reached.
- The IRT ability estimation algorithm, as illustrated in Fig. 6, takes the variables: answer vector, a priori θ, difficulty, discrimination and calculates a posteriori $\theta^{\prime}$.
- If the answer is correct ("yes" or "true"), a value of "1" is assigned. If not, a "0" is assigned.

The last posteriori $\theta^{\prime}$ in the loop is the estimated user's ability θ and can be compared to a threshold value set by an administrator. Access is granted if (θ > threshold) and denied otherwise.

4.4 Results and Summary

The prototype implementation shows that efficient dialogs could be generated from ontologies that have been enhanced with IRT attributes. The successful implementation of the IRT in dialogues of questions and answers shortens the number of questions necessary to reach an accurate estimation of subject’s ability, knowledge or trust by at least 50% as it has already been proved by the IRT literature [7, 8]. This reduction of the number of questions necessary to estimate the ability produces shorter dialogs without losing accuracy. Also, the use of IRT enables the use of multiple stopping criteria such as: fixed length number of questions or time, ability threshold and standard error confidence interval. The availability of large number of ontology axioms enables generating a set of questions different from another set to be generated immediately after the current user preserving privacy and protecting against question exposure, especially in voice systems. The success of dialog system depends upon multiple timing factors and scalability of supporting multiple users. Our on-going research addresses these two aspects.

IRT is useful in compensating for inaccurate voice recognition of answers during dialogs or accidental mistakes. The access control decisions are made based on an estimation of a level of trust in a subject derived from the importance or relevance of axioms in ontology. With the advancement in the fields of mobile, cloud and cloud based voice recognition such systems become important in defense and physical security applications [41, 42, 43].

CHAPTER FIVE: DIALOGS SUPPORTED BY CONTEXTUAL REASONING

Questions that may be asked to a person that is facing an interview for the purpose of gaining entry into a controlled region may depend on the context and vary in detail, but asked with the objective of identifying the person in a broader sense within a limited timeframe. One of the issues that interviewers face is to ask relevant questions that would enable them to either accept or reject based on this broader identity and apparent truthfulness of the attributes used to substantiate that identity. Repeating questions asked at entry point interviews may render them useless because most interviewees may come prepared to answer common questions.

As a solution, I present an interactive voice response system that can generate a random set of questions that are contextually relevant, of the appropriate and equal level of difficulty across all interviewees without repeating in successive question answer sessions. Furthermore, the system will have the ability to limit the number of questions based on the available time, degree of difficulty of generated questions or the desired subject concentration. The solution uses Item Response Theory to ensure an equal level of difficulty of chosen questions without repeating the same question using a large federated question bank generated by making inferences over multiple ontologies consisting of data taken from large distributed sources.

5.1 Contextual Reasoning

There are multiple definitions of context, but a widely accepted one as in [21]: "Context is any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and the application themselves." Context-aware application is an application that takes into account the context under which it runs. The behavior of the application will depend on factors like: social, physiological, biometric, environmental, hardware, computational, temporal, activity, identity, location, etc. [22].

Context modeling is a way to represent context [20]. In order to decide on a good context model the authors in [18] outline the following criteria: applicability, comparability, traceability, history, logging, quality, satisfiability and inference. Multiple modules have been proposed as ideal to model context, among them; database modules, Context Modeling Language [32], W3C Delivery Context Model [33], Eclipse Organization Persona, Context Data Model and the Higgins Data Model [34], etc. With the evolvement of Sematic Web and ontology and the increasing power of expressivity and reasoning capabilities, many researchers had started to model context using the Web Ontology Language (OWL). This modeling choice not only provides representation, accessibility, but also, flexibility and standardization provided by the ubiquitous use of XML technologies. All this comes in addition to the aforementioned benefit of support for logical reasoning.

Contextual reasoning provides the ability to reason or infer contextual information from context models using ontological reasoners. A reasoner can infer implicit facts from asserted contextual ones that affect the behavior of the application. This helps with application customization and with generation of run-time configurations and interfaces based on not only explicit, but also implicit context [18, 19, 20].

Contextual reasoning is used in our work to enhance our identity and access control system by including questions that are not only satisfying IRT requirements of having the necessary parameters such as: difficulty, discrimination or guessing factor, but also related to the base attributes, assertions, inferences, current context and the historical context of the same requester with the system in order to ascertain the interviewee’s ability or trustworthiness, and provide a score for the entire set of answers. Using contextual reasoning we are able to ask related subsequent questions to individuals or groups. The system should be able to ask a question with a subject or object value that was used in a previous question in the current session as well as it should be able to ask one starting from where a previous session has stopped or one that the interviewee got wrong. It is also capable of asking a requester a question with subject or object values that were asked to someone he is related to through an ontology object property either asserted or inferred. All this, while keeping the use of the IRT annotated weights of axioms and using them through the IRT ability estimation procedure to assign ability or trust score with a desired level of accuracy.

5.2 Selecting Related Questions Using Contextual Reasoning

In chapter 4, an ontological inference driven dialogs were generated. In this chapter, I enhance that system with contextual extensions by introducing two contextual reasoning algorithms for selecting related questions that have the capability to continue asking questions based on current or past context to further enhance the identification and authorization of a user.

5.2.1 Selecting Next Question Based on the Context

In the IRT/CAT only selection criteria, the next item is selected only by its IRT parameters, which has to be in a range of values close to the interim ability estimation ($\hat{\theta}$). This can result in asking questions that are inhomogeneous and unrelated. For example, the first question might be: [is it true (a isBrotherOf b)?], and the next question might be: [is it true (Obama isPresidentOf United States)?].

Here, I am expanding the IRT/CAT-based item selection by asking related question [44]. We are not only concerned about the IRT parameters of a question rendered from an annotated ontology axiom, but I also attempt to continue asking questions related to each other that test the user on a branch of knowledge. In the above example, we believe that it would have been better to ask [is it true (a isBrotherOf c)] followed by [is it true (b isFriendOf c)] because the two relations isBrotherOf and isFriendOf are contextually related by a social relationship context between two people. In order to capture such notions of context, I use the subject, predicate (property) and object of one question to generate the next question. Multiple strategies can be used to achieve this objective selecting the next question based on subject all the time; predicate all the time, object all the time or a combination of the three.

I introduce the use of a reasoner executing queries in the selection algorithm. A reasoner query using the subject, property or object of a question/axiom is executed in order to further filter the axioms in the item bank with the desired IRT parameters as well as be related to current subject, property or object. Figure 21 provides an overview of the system's architecture.

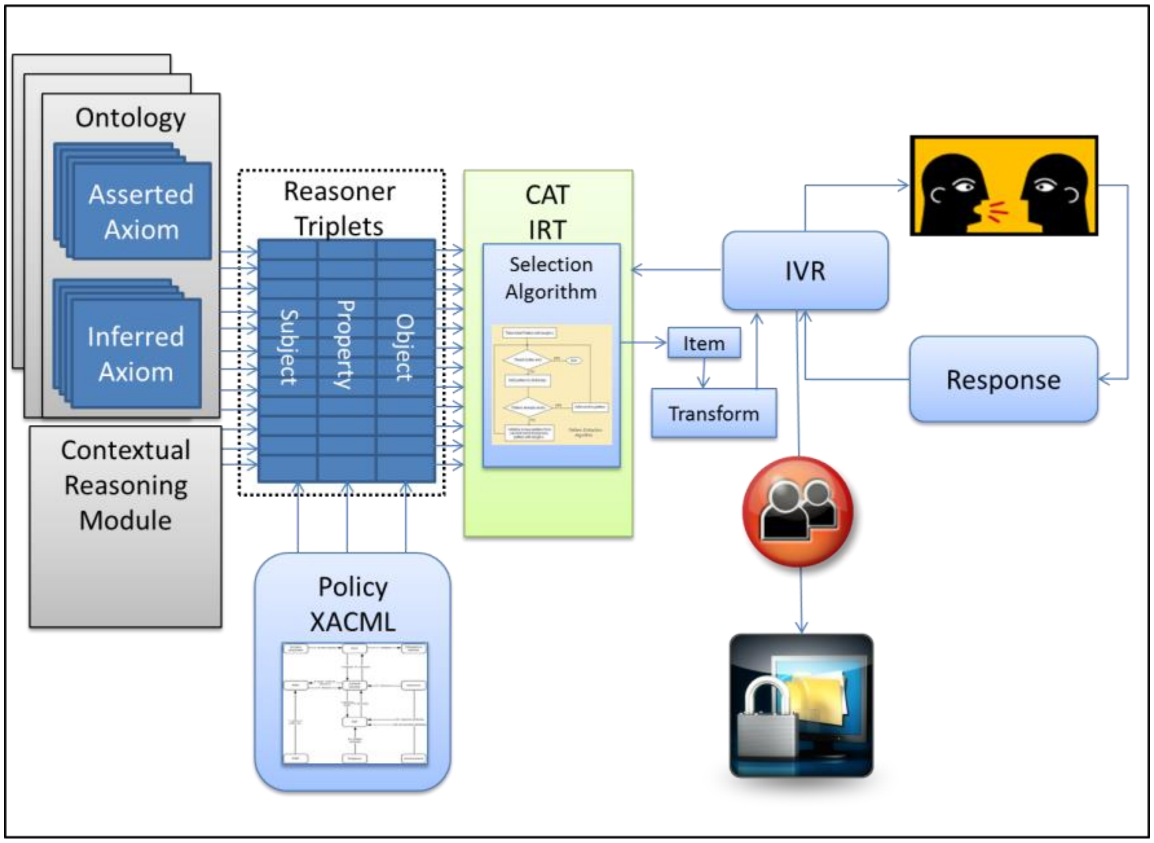


Figure 21 Contextual reasoning enhancement architecture

This architecture extends what have already been built in the system architecture of the ontology based dialogs presented in Figure 16 of the previous chapter. The main addition here is a contextual reasoning module.

In this architecture, the main source of knowledge and thus questions is still the ontology. The asserted axioms and the preferable inferred ones are decomposed to their triplet components (i.e. subject, property and object). The ontology is annotated with IRT parameters to facilitate the compatibility with IRT algorithm. Reasoning is used in the same way introduced in the previous chapter to give more weight to the harder questions, which in this case means the ones with the more explanation axioms.

The main difference between this architecture and the previous one is the item selection enhancement. In the previous one, IRT parameters were the only factor to decide which question to select among the item bank and present to the user. Here, we are adding to this, the ability to add questions related to the dialog context.

In this module, I am using the current question's context of subjects, properties and objects and add it to the original ontology when running reasoning queries to select most appropriate; probably hard questions, and related to the context. An example of the result of this contextual reasoning is the selection of a question whose subject is an object of the current question such as: [Q1: Does Bob know *Alice*?], [Q2: Does *Alice* know John].   This addition of this query further filters the triplets originally asserted or inferred by the reasoner.

The selected question is rendered verbally through Text-To-Speech standards to the user who responds with an utterance. If the utterance is recognized then the answer is evaluated. If it is correct then the IRT ability estimation algorithm increases its a priori value for the user commensurate to the difficulty.   If incorrect then the IRT ability estimation algorithm decreases its a priori value for the user commensurate to the difficulty. Another question is asked and the cycle goes again until a threshold or a fixed number of questions is reached. A final decision can now be made to grant or deny access.

5.2.2 History Based Contextual Reasoning for Item Selection

With historical contextual reasoning I expand the reasoning about current context presented in the previous section by saving the user's session questions and answers into ontology. I run a reasoner over the closure of these session ontology with the axioms in the item bank ontology. The goal is to be able to ask questions related to the ones presented in previous sessions. Multiple selection strategies can be followed like:

- Asking the questions related to the ones that the user got wrong in a previous session, but not the same question.
- Asking a question that tests deeper knowledge building over a correctly answered question in a previous session.
- Asking a question related to questions that were asked of other users that are known (by inference) to be related to the current user under questioning, such as co-workers, family members, friends, colleagues, etc.

This capability brings benefits, especially when such a system is used to identify, interview and evaluate individuals multiple times or groups of related people, for example, in immigration or security clearance interviews. It also gives us the ability to compare and be able to detect any changes in the user's behavior or trust over time and possibly detect masquerading of personalities.

To implement historical contextual reasoning, I save a user's ID, question ID and a session ID and other situational data such as the location, time, time spent answering questions, purpose of the previous interview, etc. These historical and contextual data are modeled using ontologies, giving us the great advantage of reasoning support. Figure 22 illustrates our historical contextual reasoning based IVR system.

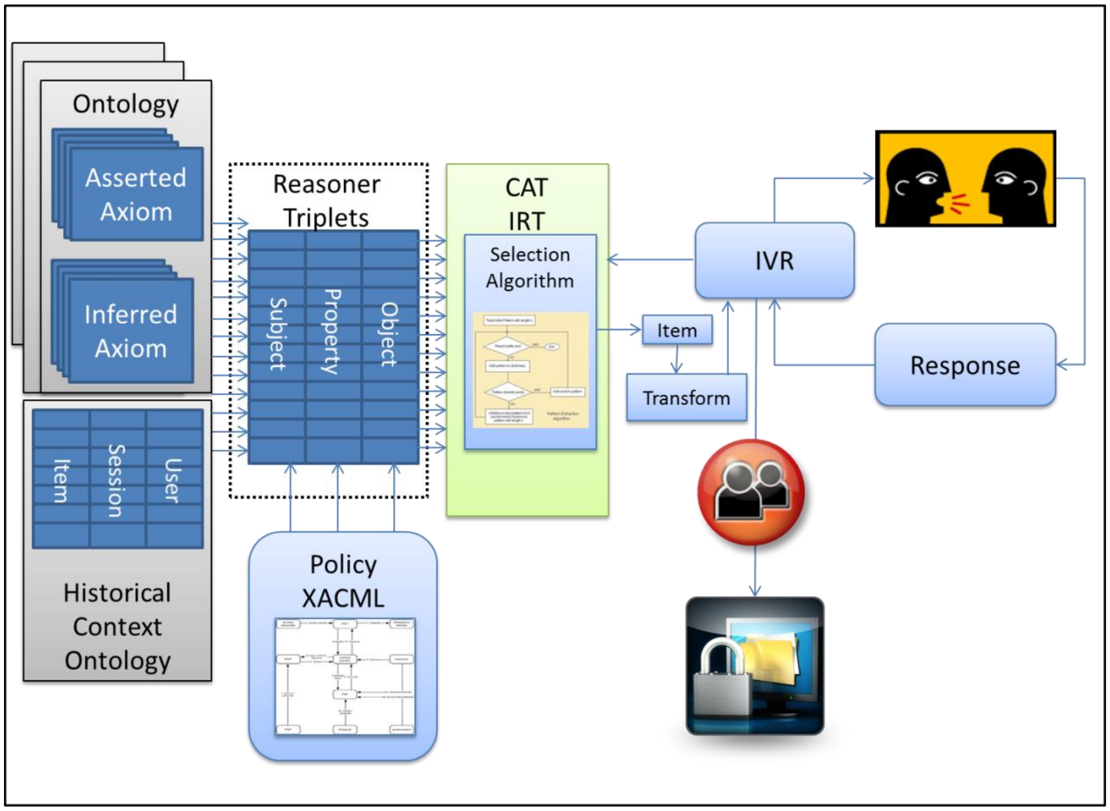


Figure 22 Historical contextual reasoning enhancement architecture

A module for historical context ontology is added to our system architecture. This ontology has mainly item, session and user information collected from each user interaction with the IVR. An ontological reasoner is called to execute a query after the user answers a question from the ontology item bank. The reasoner will run over the closure of the historical contextual ontology and the item bank ontology. The result of the query will be transformed to a question and presented to the user.

In the historical context reasoning, we execute a reasoner query with the original ontology, session (context) ontology and use the result to generate questions. Line 15 of the algorithm in Figure 23 in the next section illustrates this.

5.3 Implementing Dialogs from Ontology and Contextual Reasoning

The conversation starts with a menu in VoiceXML hosted on the local Voxeo Prophecy web server. The voice browser connects to the web server and converts text to speech and speech to text. Figure 23 shows the algorithm integrating ontology, IVR and IRT.

```
Algorithm 2: dialogue access decision evaluation
Input: a priori theta, Difficulty, Discrimination, Answer
Output: access control decision
/* make access control decision from ontology*/
1:  domDocument=parse(ontology); // DOM
2:  subjectArray=getAxiomSubject(axiom);
3:  propertyArray=getAxiomProperty(axiom);
4:  objectArray=getAxiomObject(axiom);
5:  difficultyArray=getAxiomDifficulty(axiom);
    /*use voiceXML and JSP to generate the dialog*/
6:  for (counter < items.length) do
7:     <vxml:Prompt> 'auxiliary verb' +propertyArray[i] + " " +
    objectArray[i] +" "+ subjectArray[i];
8:     <vxml:Field>= user_utterance;
9:     response[i] =    Field.voiceRecognition(user_utterance);
10:    if response[i]= 'Yes' or 'true'
11:       resultVector[i]=1;
12:    else
13:       resultVector[i]=0;
    //Contextual reasoning
14: Reasoner.Query([subject][object][property], itemBank);
    //Historical contextual reasoning
15: Reasoner.Query([subject][object][property], itemBankOntology,
    sessionOntology);
16: endfor;
17: theta = IRT_algorithm(resultVector, difficulty, discrimination,
    aPrioriTheta);
18: if theta > thetaThreshold
19:    permit;
20: Else
21: deny;
```

Figure 23 Ontology-IVR algorithm with IRT and context

The main steps are as follows:

- **Line 1:** Load the ontology and parse the XML into Document Object Model (DOM) [24].
- **Lines 2-4:** Extract the axiom's triplet (subject, property, object)
- **Line 5:** Extract the axiom's IRT difficulty value from the annotation
- **Lines 6-7:** Establish a VoiceXML "For" loop that synthesizes a question from string or text values to speech (TTS). The question consists of an auxiliary verb, object, property and subject to test the correctness of an axiom.
- **Lines 8-9:** The system waits for a response. If there is one it converts it to text and recognizes it. If it adheres to grammar then a value is assigned as an

answer. If there was no answer then VXML re-prompts the question up to a programmed number of times. If exceeded then an appropriate VXML is executed. An upper level VXML routine is taking care of no input and number of trials allowed.

- **Lines 10-13:** If the answer is correct ("yes" or "true"), a value of "1" is assigned. If not, a "0" is assigned
- **Line 14**: For implementation with contextual reasoning and in order to ask the next question, a reasoner query is executed with one of the subject, property or object as a parameter that filters the items in the item bank. (Line 15 of the algorithm)
- **Line 15:** For implementation with historical contextual reasoning and in order to ask the next question, a reasoner query is executed on the session history and item bank ontologies with one of the subject, property or object as a parameter.
- **Line 16:** The end of the VXML for loop
- **Line 17:** The vector of binary answers is used to estimate the IRT ability using algorithm in Figure 8. It takes the variables: answer vector, a priori θ, difficulty, discrimination and calculates a posteriori θ'.
- **Lines 18-21:** The last a posteriori θ' is an estimation of the user's ability θ and can be compared to a threshold value set by an administrator. Access is granted if ($\theta >$ threshold) and denied otherwise.

Recognizable utterances have to conform to a grammar [40]. For our direct implementation, we use a simple hard-coded inline grammar that enforces yes, no, true, false as shown in Figure 24.

```
<grammar xml:lang="en-US" root ="MYRULE"
mode="voice">
    <rule id="MYRULE" scope = "public" >
       <one-of>
          <item> yes </item>
          <item> no </item>
          <item> true </item>
          <item> false </item>
        </one-of>
    </rule>
</grammar>
```

Figure 24 An inline grammar used

5.4 The Context Ontology

To implement the historical contextual reasoning we need to persist context such as: user's ID, question ID, session ID, location, time, purpose of, etc. These historical and contextual data are modeled using ontologies enabling reasoning support. Figure 25 shows sample context ontology with the following classes:

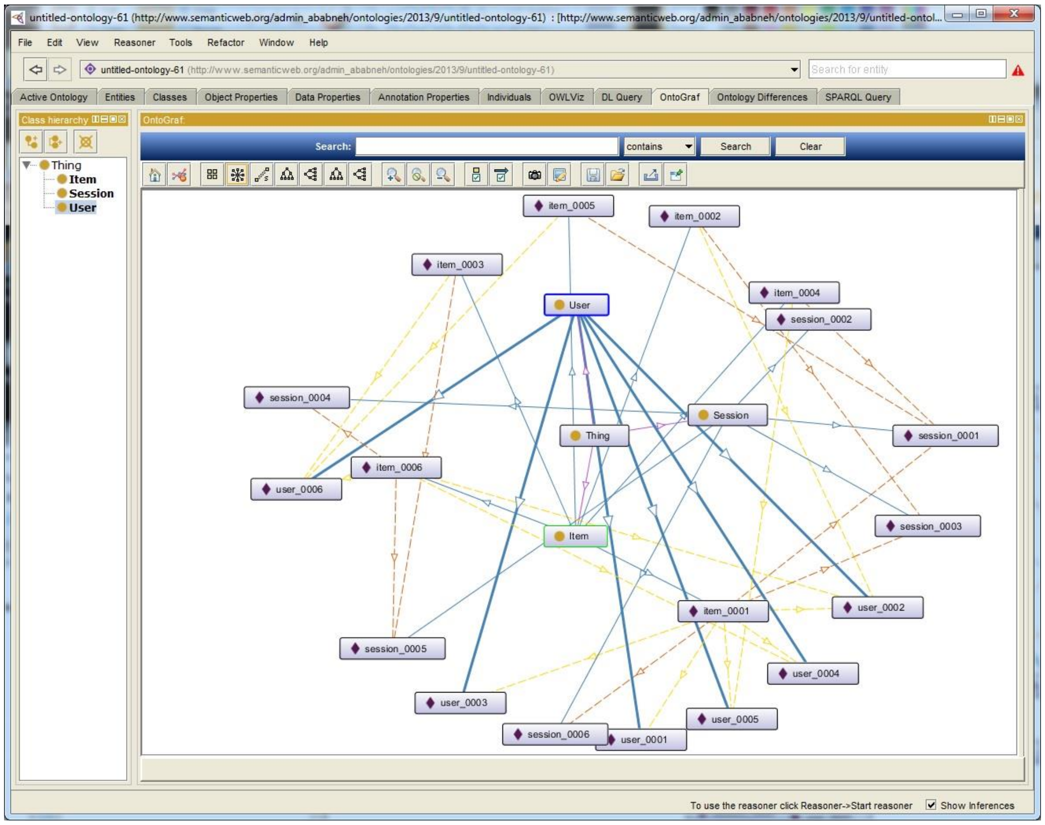

untitled-ontology-61 (http://www.semanticweb.org/admin_ababneh/ontologies/2013/9/untitled-ontology-61) : [http://www.semanticweb.org/admin_ababneh/ontologies/2013/9/untitled-ontol...
File Edit View Reasoner Tools Refactor Window Help
untitled-ontology-61 (http://www.semanticweb.org/admin_ababneh/ontologies/2013/9/untitled-ontology-61)
Search for entity
Active Ontology Entities Classes Object Properties Data Properties Annotation Properties Individuals OWLViz DL Query OntoGraf Ontology Differences SPARQL Query
Class hierarchy
Thing
Item
Session
User
OntoGraf
Search:
contains
Search
Clear
item_0005
item_0002
item_0003
item_0004
session_0002
User
session_0004
Session
session_0001
Thing
item_0006
user_0006
session_0003
Item
item_0001
user_0002
session_0005
user_0004
user_0003
user_0005
session_0006
user_0001
To use the reasoner click Reasoner->Start reasoner
Show Inferences

Figure 25 The contextual reasoning ontology

- **Item:** defines a question generated from an axiom of: subject, object, property, irt_difficulty annotation and ID annotation
- **Session:** defines a session of interaction between the IVR and the user. The session ID is extracted from the IVR.
- **User:** defines a user. In our application the user will be a "sameAs" and individual in the item bank ontology.

For each question asked, a set of axioms are added to the context ontology. Some examples are:

- User_0001 hasQuestion item_0002

- Item_0001 wasOfferedIn Session_ dd552fcdc5fccef412f96d38818a1c25
- Item_0002 wasOfferedIn Session_ dd552fcdc5fccef412f96d38818a1c25
- Session_ dd552fcdc5fccef412f96d38818a1c25 timeDateIs “Jun 30, 2009 7:03:47 AM”
- Session_ dd552fcdc5fccef412f96d38818a1c25 callerLocation “Fairfax, VA”
- Item_0001 outcome “correct”
- Item_0002 outcome “incorrect”
- Session_ dd552fcdc5fccef412f96d38818a1c25 abilityEstimate “2.9”
- Session_ dd552fcdc5fccef412f96d38818a1c25 outcome “grant”
- User_0001 sameAs “IRI:IRT: DiasKadyrbayev”
- User_0001 hasSession Session_ dd552fcdc5fccef412f96d38818a1c25
- Session_ dd552fcdc5fccef412f96d38818a1c25 warning “None”
- Session_ dd552fcdc5fccef412f96d38818a1c25 recommendation “Interview upon entry”

The existence of such axioms makes contextual aware reasoning possible. Queries to select questions of the current user in this session or in a previous one can be executed. Also, selecting questions that were asked to someone linked somehow to the user becomes possible.

5.5 Summary

The prototype implementation shows that efficient dialogs could be generated from policies and ontologies that have been enhanced with contextual reasoning and IRT attributes. The use of contextual reasoning adds to the previous benefit by narrowing the scope of the questions and enables questioning on related and continuous content. Also, using historical and contextual reasoning to generate questions that are in a sense a continuation to previous conversation or related to sessions that other users related to this person provides a better sense of identification and possible derive a more complete personality trait or changes in personality traits that may occur over a longer period of time. Based on the track of questions in the current session or previous ones, our selection algorithm can select harder, but related questions that are more efficient in evaluating the user's ability or credibility with the purpose of identification or authorization.

CHAPTER SIX: PERFORMANCE ANALYSIS

In this chapter I evaluate the performance of question generation using contextual reasoning introduced in the previous chapter. Generating relevant questions requires frequent querying after each question using its context to generate the next one. This process takes time and might lead to a situation where the interviewee waits for a question for an unacceptably long time. The objective of this work is to show that this process doesn't cause delays that might turn the dialogs useless.

There has been much work evaluating the performance of DL reasoners and ontology repositories since the technology started to prevail [46, 47, 48]. There have been benchmarks introduced by reputable institutions that can be used in such evaluations [45, 52]. Also, with cloud computing and cloud based services and repositories, more performance experiments have been conducted and benchmarks were upgraded for more scalability testing [49, 50].

There are different types of delays in the system cycle while generating questions from policies and ontologies such as; annotation, converting text to speech, waiting for response, voice recognition, ability estimation and decision making. My focus here is on the performance of frequent querying of contextual reasoning.

6.1 Performance Analysis of Ontological Inferences

There are several factors that can affect the performance of reasoning:

- The size of the ontology: the larger the ontology the more time it will require to traverse all the triples to check applicability and return results.
- The kind of the query: the purpose of reasoning varies according to the application. Consistency checks are different than entailment checking etc. ABox reasoning checks if an individual is an instance of a class and the relationships between individuals. TBox reasoning checks if a class is subsumed by another. Consequently, an application may use one or both of ABox and TBox derivations.
- The expressiveness of the ontology: the OWL ontology standard and RDF have multiple profiles (DL, lite, full, etc.) that reflect various degrees of expressiveness required to model application needs. Most of the time, ontologies are built using some of these variations. This affects the reasoning results themselves and the performance of this reasoning.
- The algorithm used by the reasoner: different reasoners use different kinds of algorithms to infer or perform the same task. Some examples are Rule based derivations, Tableau based derivations and Resolution Theorem provers, etc.
- The computing resources: The CPU, memory, space, virtualization, Java heap space, etc. all affect performance.

6.2 Most Popular Benchmarks

6.2.1 LUBM

The Lehigh University Benchmark (LUBM) was developed to facilitate the evaluation of Semantic Web repositories in a standard and systematic way [45]. The benchmark is intended to evaluate the performance of those repositories with respect to extensional queries over a large data set of a single real-life ontology. It consists of a domain ontology for universities with customizable and repeatable synthetic data, a set of test queries, and several performance metrics. Appendix 1 shows the ontology design.

6.2.2 University of Berlin's BSBM

The Berlin SPARQL Benchmark (BSBM) defines a suite of benchmarks for comparing the performance of these systems across architectures [51]. The benchmark is built around an e-commerce use case in which a set of products is offered by different vendors and consumers have posted reviews about products. The benchmark query mix illustrates the search and navigation pattern of a consumer looking for a product.

6.3 Contextual Reasoning Performance Analysis

For the performance analysis I use a benchmark, SPARQL queries, owl ontology through a repository and conduct two experiments; one using the LUBM synthetic ontology using queries simulating the ones used for contextual reasoning in the second one, which using my homeland security ontology, but with larger number of synthetic axioms.

6.3.1 Objective of My Experiment

The objective of my experiment here is to ensure that the generation of questions from my ontology using contextual reasoning with frequent querying is not affecting the overall performance of the system through any delays.

6.3.2 Experiment 1

In order to test the performance of queries used for contextual reasoning, I used the LUBM benchmark package. I generated synthetic ontology samples using the generator software. I have generated ontologies for 1, 2, 5, 10, 20, 50 and 100 universities. Table 5 illustrates statistics about our sample. It shows the number of universities, number of axioms and size in gigabytes.

Table 5 LUBM Sample

	OWLIM_1	OWLIM_2	OWLIM_5	OWLIM_10	OWLIM_20	OWLIM_50	OWLIM_100
University#	1	2	5	10	20	50	100
Axiom#	0.08M	0.2M	0.5M	1.15M	2.2M	5.5M	11M

6.3.2.1 Sub Experimental Environment

The machine used for testing was has a CPU of Intel i7, 8GB of memory, 1TB of storage. The LUBM generator and the tester were run under Java 1.6 with a heap space of 256MB. The operating system was Windows 7. LUBM loaded the ontology into an OWLIM repository. The tester application executes the query on that repository.

6.3.2.2 Experiment Execution

To test the performance of the contextual reasoning queries, I used the LUBM tester package which executes the targeted set of ontologies using the standard set of 14 queries testing various levels of complexity. I have added three queries to this set that return all predicates and objects participating in an axiom with a designated subject in

Query 15. Return all predicates and subjects participating in an axiom with a designated target in Query 16. And lastly, return a union of previous two queries in Query 17. Appendix 2 lists the original queries and Appendix 3 lists the added ones.

Table 6 shows the SPARQL query execution time on my test ontology sample. Queries from 0 to 14 are the standard LUBM queries and I added queries 15 to 17 to test contextual reasoning.

Table 6 LUBM SPARQL query execution time in milliseconds over ontologies in OWLIM

Query	OWLIM_1	OWLIM_2	OWLIM_5	OWLIM_10	OWLIM_20	OWLIM_50	OWLIM_100
query0	113	1201	131	199	180	197	644
query1	61	821	144	147	173	318	697
query2	38	2987	64	87	181	662	2946
query3	1	2	1	1	1	1	1
query4	9	6	6	5	5	5	5
query5	26	11	15	10	8	8	10
query6	58	512	103	64	151	242	482
query7	4	3	3	3	2	2	2
query8	231	2262	165	237	416	926	1893
query9	44	5675	219	454	932	2411	5336
query10	1	1	1	1	1	1	1
query11	2	1	2	2	2	2	2
query12	7	133	9	13	25	58	157
query13	1	3	1	1	1	2	3
query14	2	324	12	22	51	119	295
query15	114	309	102	108	104	95	433
query16	2	2	2	2	1	2	4

query17	5	5	6	5	5	6	6

Table 7 shows the SPARQL query execution results or number of axioms return after each successful execution on our test ontology sample. Again, Queries from 0 to 14 are the standard LUBM queries and Queries 15 to 17 were created by me to test the performance of contextual reasoning.

Table 7 LUBM SPARQL execution number of results

Query	OWLIM_1	OWLIM_2	OWLIM_5	OWLIM_10	OWLIM_20	OWLIM_50	OWLIM_100
query0	540	72302	3373	6843	14457	35973	72302
query1	4	4	4	4	4	4	4
query2	0	264	9	28	59	130	264
query3	6	6	6	6	6	6	6
query4	34	34	34	34	34	34	34
query5	719	719	719	719	719	719	719
query6	7790	1048532	48582	99566	210603	519842	1048532
query7	67	67	67	67	67	67	67
query8	7790	7790	7790	7790	7790	7790	7790
query9	208	27247	1245	2540	5479	13639	27247
query10	4	4	4	4	4	4	4
query11	224	224	224	224	224	224	224
query12	15	15	15	15	15	15	15
query13	1	472	21	33	86	228	472
query14	5916	795970	36682	75547	160120	393730	795970
query15	6	6	6	6	6	6	6
query16	17	17	17	17	17	17	17

query17	23	23	23	23	23	23	23

Figure 26 illustrates a graph of the SPARQL query performance of execution time on our test ontology sample. Queries from1 to 14 are the standard LUBM queries and 15 to 17 were created by me to test the performance of are contextual reasoning.

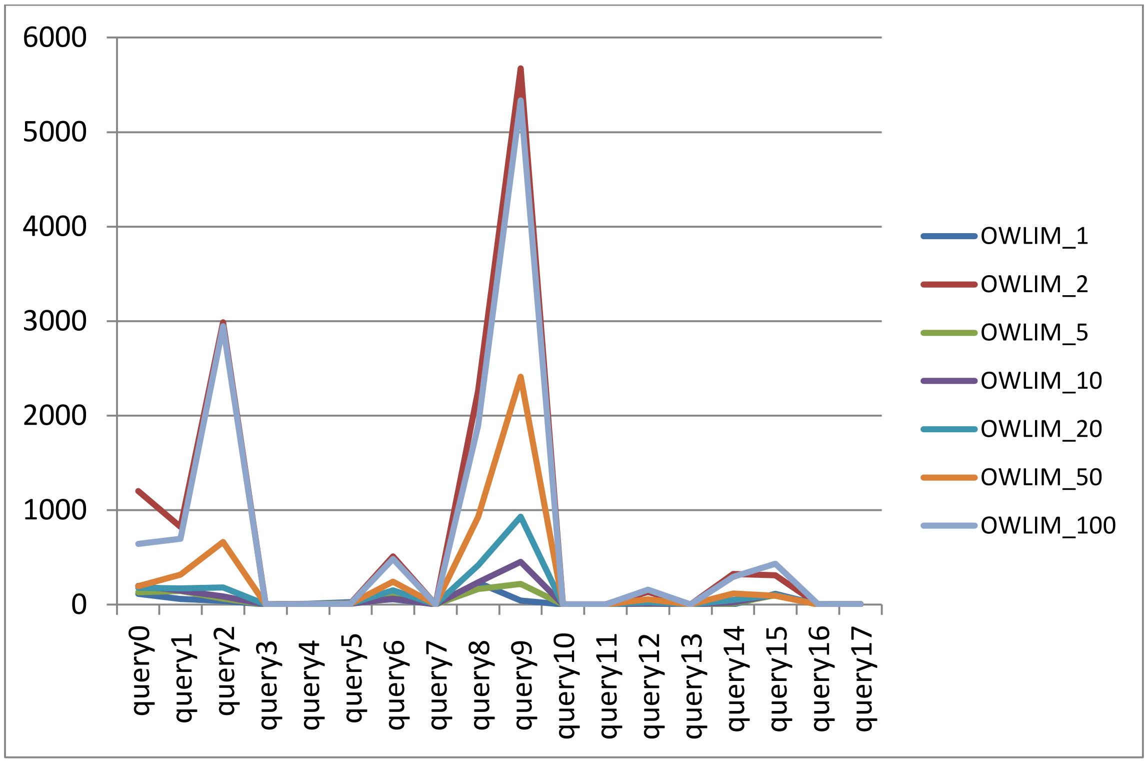


Figure 26 LUBM SPARQL query performance in milliseconds

Figure 27 illustrates a graph of the SPARQL query number of results returned after execution on our test ontology sample. Queries from1 to 14 are the standard LUBM queries and 15 to 17 are our added queries from contextual reasoning.

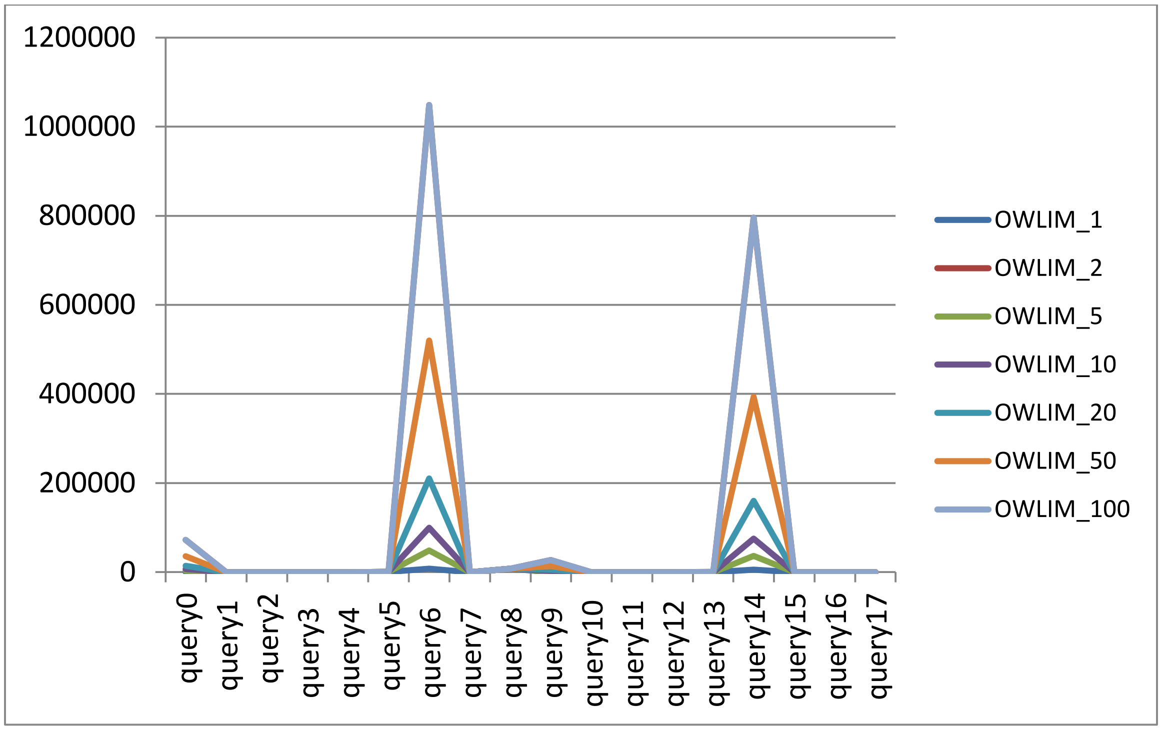


Figure 27 LUBM SPARQL query number of results

6.3.3 Experiment 2

In order to test the performance of queries used for contextual reasoning on an ontology used in my Use Cases, I used the homeland security ontology, but with larger samples of random individuals instantiating ontology classes. I generated synthetic ontology samples using Java and OWLAPI. I have generated ontologies of sizes related to number of students in foreign student database. Axioms like [Student_(randomNumber) isA Student], [Student_(randomNumber) hasVisa F-1], [Student_(randomNumber) isFromCountry Country_(randomNumber)] were added to produce the ontology simulating the LUBM ontology. Specifically, the ontology has the following structure:

- There are a number of student individuals of type Student
- There are 200 countries
- There are two types of student visas; F-1, J-1
- Every student in this ontology has a visa; either F-1 or J-1
- Every student in a national of one of the 200 countries
- Every student has random number of 1 to 20 friends who are also students
- Every student has participated in a session with the system for a random number of 1 to 20 times.

Table 8 illustrates statistics about the sample. It shows the number of axioms and size in megabytes.

Table 8 Homeland Security Ontology Sample of Foreign Students

HS_(number of students)	**HS_100**	**HS_1000**	**HS_5000**	**HS_10k**	**HS_50k**	**HS_75k**	**HS_100k**	**HS_200k**	**HS_250k**
Students#	100	1000	5000	10000	50000	75000	100,000	200,000	250,000
Axiom#	2500	23000	0.1M	0.2M	1.1M	1.7M	2.2M	4.6M	5.7M

6.3.3.1 Experiment Environment

The machine used for testing was with a CPU of Intel i7, 8GB of memory, 1TB of storage. The LUBM generator and tester were run under Java 1.6 with heap space of 256MB. The operating system was Windows 7. The ontology was loaded into an OWLIM repository to maintain compatibility with experiment 1. The tester application executes the query on that repository.

As can be seen from Table 8 above, this test environment didn't allow us to generate a very large ontology to test on. One of the reasons for this is the limited computing power of the desktop machine available to run the experiment. The other reason is the ontology repository used, which is the lite, free and limited version. The ontology of 300,000 students could be generated, but the file write failed because of the lack of memory. Anything more than that number failed in the generation step. For ontologies of around 100,000 students, which translated to around 2,000,000 axioms, the repository loader also failed. But even with this, it was sufficient to get to our conclusions using available readings.

6.3.3.2 Experiment Execution

To test the performance of contextual reasoning queries, I have used the synthesized homeland security ontology. Three contextual queries executed on the data set return all predicates and objects participating in an axiom with a designated subject in Query 15. Return all predicates and subjects participating in an axiom with a designated target in Query 16. Query 17 returns a union of previous two queries. One more query 18 returns all axioms to compare the performance of the previous three with. Also, a limited version of the 4 queries that limits the result retrieval to 100 axioms were added.

Table 9 shows the SPARQL query execution results or number of axioms returned after each successful execution on our test ontology sample.

Table 9 SPARQL query execution results over homeland security ontology

Query	HS_100	HS_1000	HS_5000	HS_10k	HS_50k	HS_75k	HS_100k
query15	21	21	44	27	30	29	27
query16	0	0	9	7	16	2	3
query17	21	21	53	34	46	31	30
query18	835	3018	24535	239566	1199949	n/a	n/a

Table 10 shows the SPARQL query execution time on this test ontology sample.

Table 10 SPARQL query execution time in milliseconds over homeland security ontology

Query	HS_100	HS_1000	HS_5000	HS_10k	HS_50k	HS_75k	HS_100k
query15	8	7	13	9	9	10	10
query16	6	3	8	7	6	7	8
query17	9	7	10	9	10	9	8
query18	66	142	1353	12133	58918	n/a	n/a
query15_100	8	7	13	9	9	10	10
query16_100	6	3	8	7	6	7	8
query17_100	9	7	10	9	10	9	8
query18_100	20	19	17	18	21	18	23

Figure 28 illustrates a graph of the SPARQL query number of results after executing queries 15 to 18 on our test ontology sample. My results show that it would be impractical to retrieve large number of results to generate questions from them quickly.

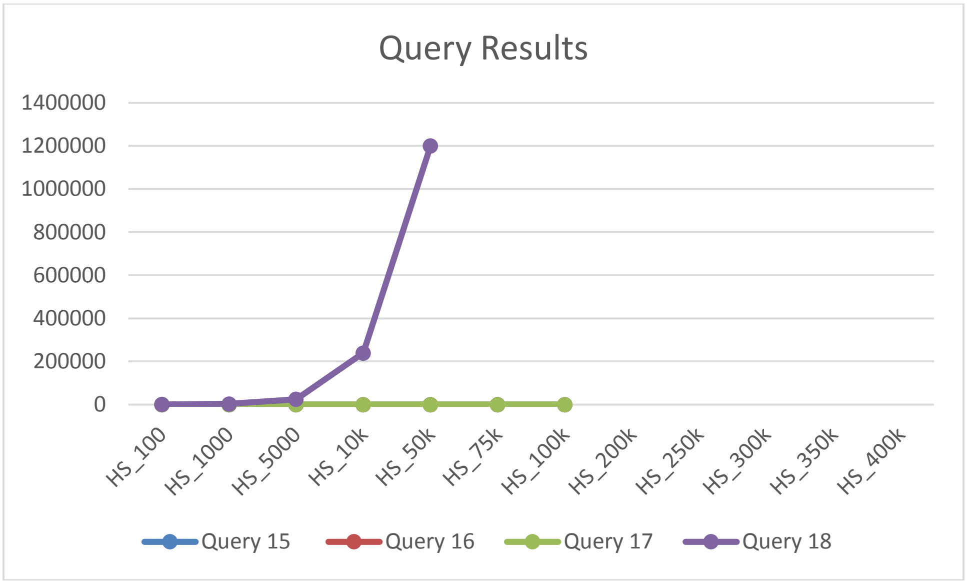


Figure 28 Homeland Security Ontology SPARQL query results

Figure 29 illustrates a graph of the SPARQL query number of results after executing queries 15 to 17 to give a better idea about the difference between the three contextual queries and the non-contextual returning all results as seen in Figure 28. The number of results is relatively small and the expectation that the time needed to retrieve them would be also small as it is proved in the next graph.

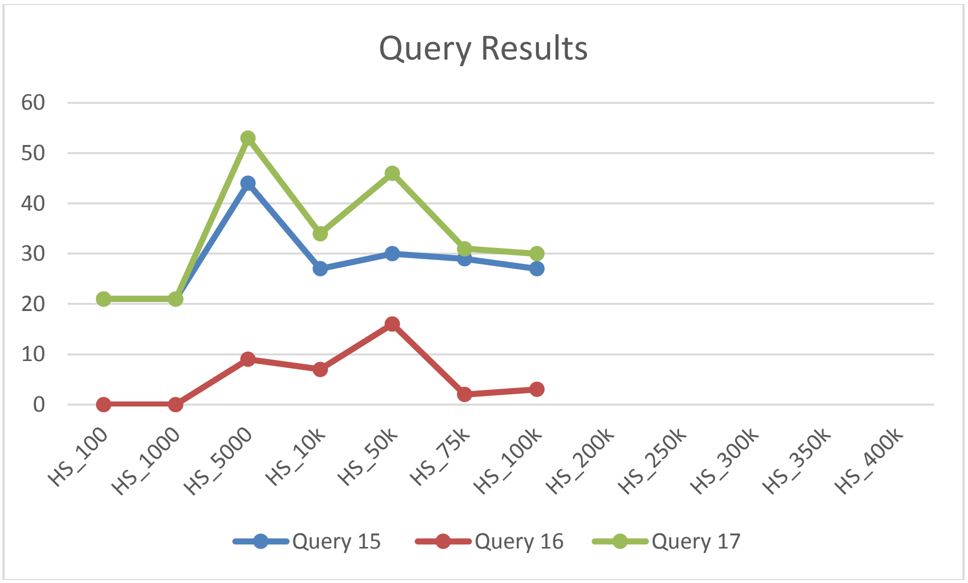


Figure 29 Homeland Security Ontology SPARQL contextual query results

Figure 30 illustrates a graph of the SPARQL query performance after executing queries 15 to 18 on our test ontology sample. Starting from ontology with 10,000 students, query 18 was taking more than 10 seconds to retrieve results, which is undesirable. But the more important ones 15 to 17 are still performing reasonably

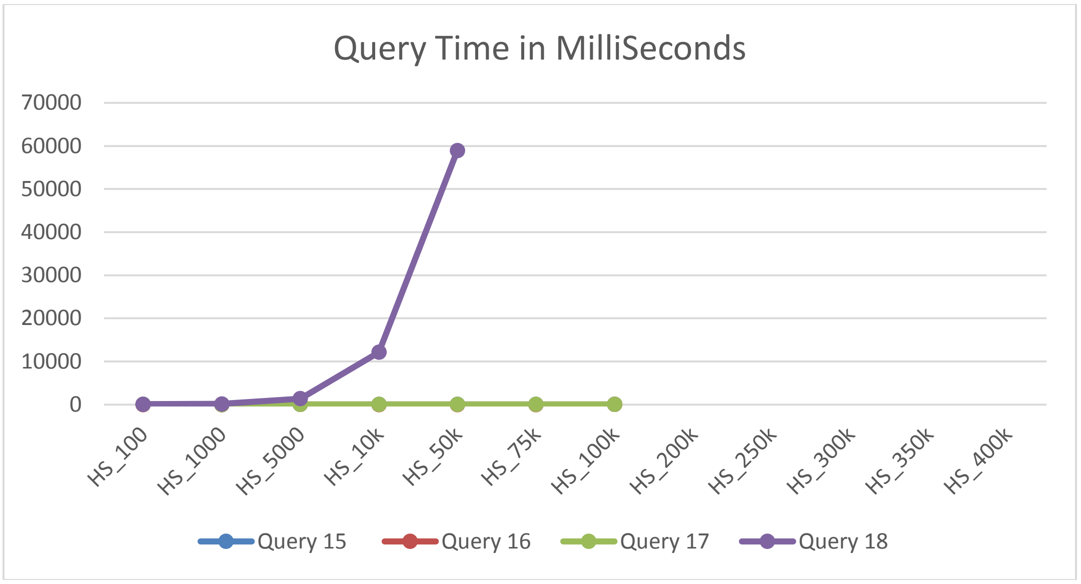


Figure 30 Homeland Security Ontology SPARQL query performance

Figure 31 illustrates another graph of the SPARQL query execution time in milliseconds after executing queries 15 to 17 to give a better idea about the difference between the three contextual queries and the non-contextual returning all results in Figure 30.

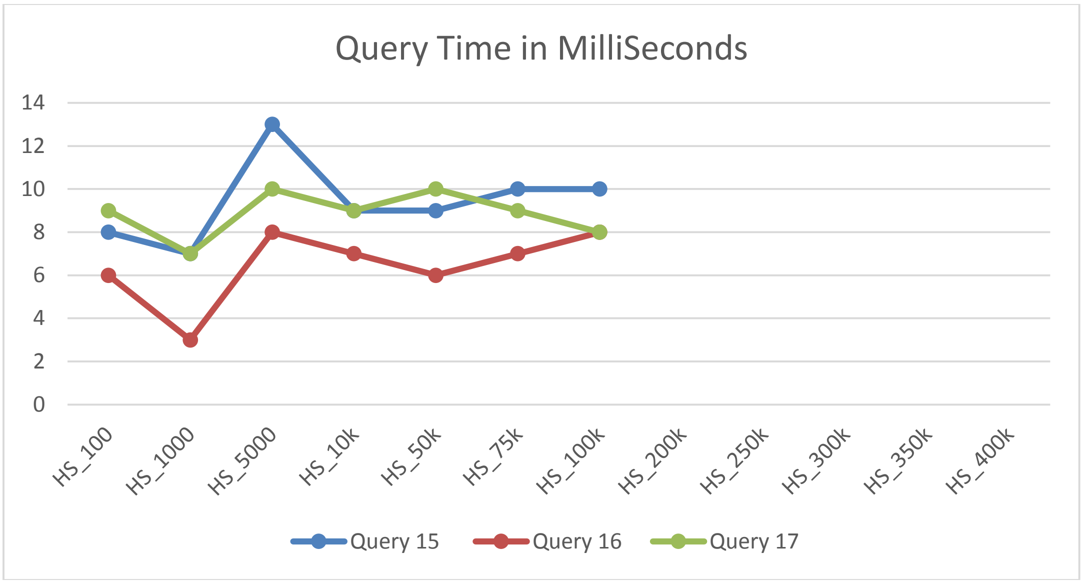


Figure 31 Homeland Security Ontology SPARQL contextual query performance

Figure 32 shows another graph of the SPARQL query execution time in milliseconds after executing queries 15 to 18 and limited versions of them. The limitation is imposed by returning only 100 relevant results and not all of them. It can be seen that query 18 is taking 58918 milliseconds, which is considered too long time to return a result. Figure 33 illustrates the same graph excluding query 18, which returns all axioms and leaving its limited version to give a better idea about the difference between the original three contextual queries and the limited version. It can be seen that it takes from 3 to 25 milliseconds to return related results.

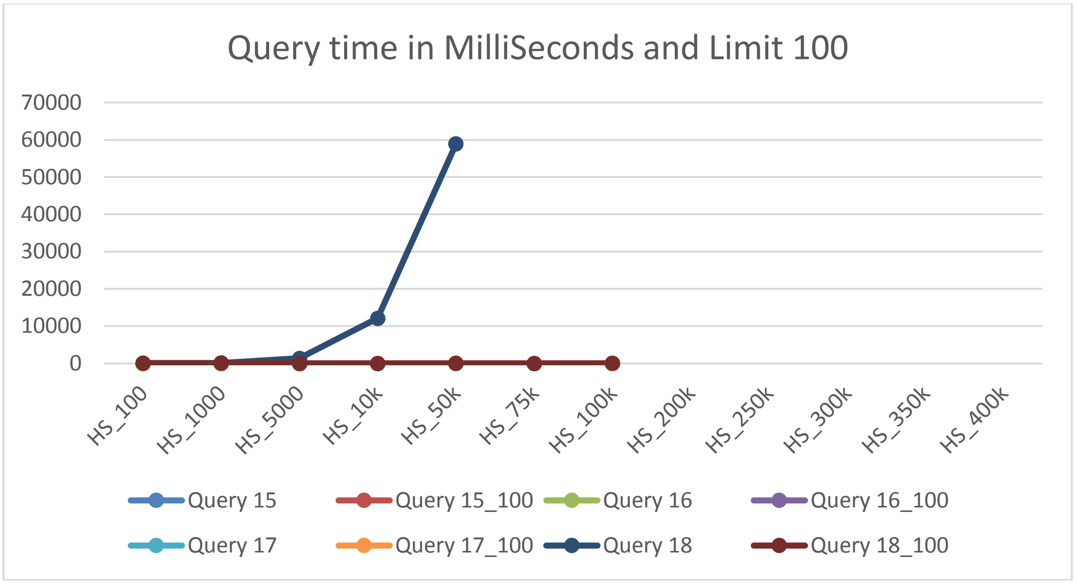


Figure 32 Homeland Security Ontology SPARQL query performance 1

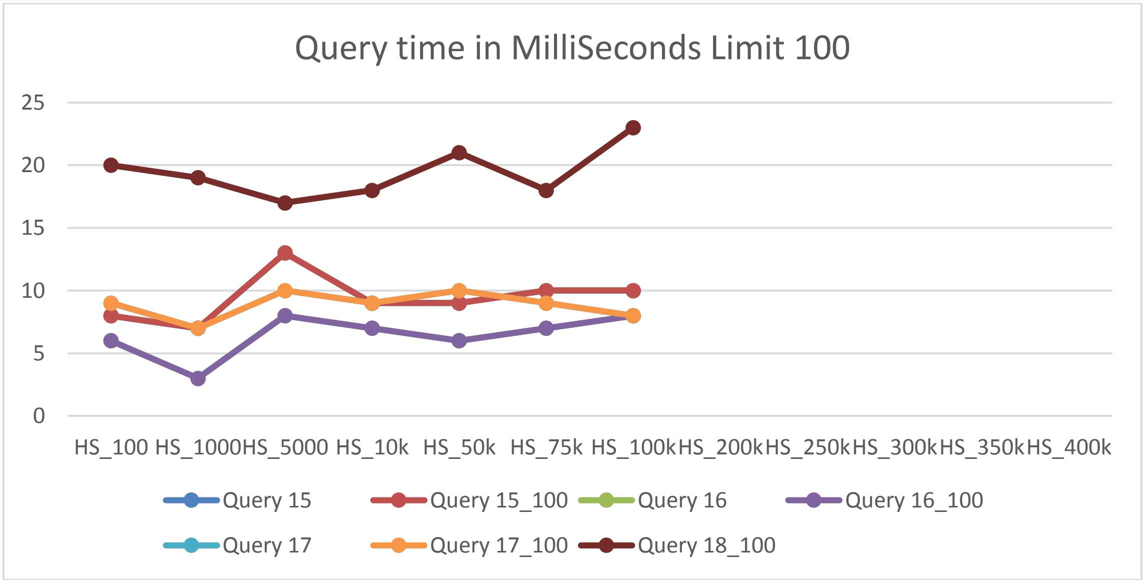


Figure 33 Homeland Security Ontology limited query performance 2

Figure 34 is a better illustration of the performance of the original and limited versions of queries and excluding query 18.

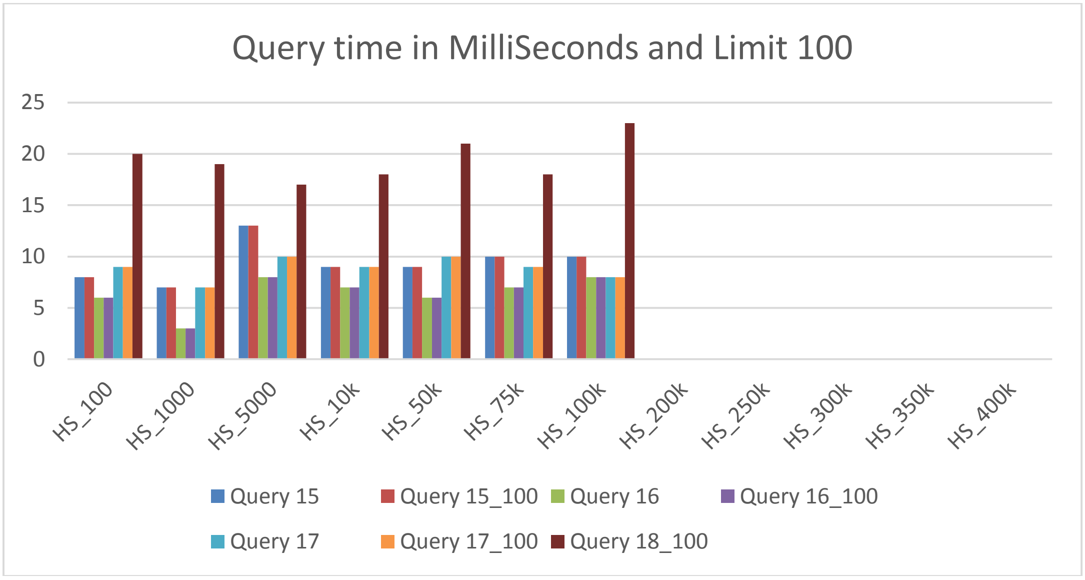


Figure 34 Homeland Security Ontology limited query performance 3

6.4 Experimental Results

The output from experiment 1 as shown in Tables 6, Table 7, Figure 26 and Figure 27 show that the execution time of the contextual reasoning queries 15, 16 and 17 uses between 1 and 433 milliseconds compared to other queries from the original LUBM set of queries, that takes a maximum of up to 6 seconds (as in the case of Query 9). The output from experiment 2 as shown in Table 9, Table 10, and Figures 28 through 34 show that the execution time of the contextual reasoning queries 15, 16 and 17 uses between 3 and 23 milliseconds. This shows that this type of queries used for contextual reasoning can be executed relatively quickly, especially when retrieving a limited number of related results. It also shows that no significant delay is expected when executing the contextual reasoning queries.

From experiment 2, we notice that the execution time depend on the size of the output. The execution time can be minimized by returning back only a limited number of results, because our goal is to select an appropriate number of axioms that are relevant, not all of them. This is useful in the case if large number of results is returned as seen in Queries 6, 14 and 18 and longer time is required to materialize them causing delay in generating the questions.

6.5 Summary

The contextual reasoning we have introduced doesn’t have a negative impact on performance and the overall ability to generate questions without delay that may cause frustration or unwillingness of the user to cooperate with the system. In case, a delay happens we can always ask to return a reasonable number of results will reduce time of result retrieval and improve performance. But that comes at the cost of limiting the retrieved results that can be used to generate questions.

CHAPTER SEVEN: CONCLUSION

It is feasible to develop a dynamic dialog system using semantic web technologies. An implementation of prototype using ontologies, item response theory and contextual reasoning had shown this feasible. The use of IRT theory provided the means to quantitatively characterize dialog items and measure trust or ability. The use of ontology and reasoning was critical in developing dialogs with items differentiated quantitatively using undisputable logic (number of explanation facts). Contextual reasoning provided the means to not only select the most appropriate question quantitatively, but also semantically relevant to the domain and subject at question. Specifically,

- **IRT**

The IRT ability estimation algorithm was successfully implemented in this work. IRT parameters were also successfully assigned to policies and ontologies. For policies, I have used the profile extension mechanism supported by the standard. For the ontology, I have used annotation to assign IRT parameters.

- **Ontology**

I have successfully implemented ontology based and supported IVR system for access control. The use of ontology enabled us to supplement any missing attributes from a limited policy and strengthen it by asking about attributes that are not in the policy. The nature of ontology of having large number of axioms enabled us to generate a huge set of questions that can be varied for different users. This is important in a voice system. The annotation feature enabled us to make it IRT compatible.

- **Reasoning**

The reasoning feature enabled us to deduce inferred facts and generate questions from them. These are virtual facts that can avoid attacks and cannot be stolen because they are not persisted physically.

- **Contextual reasoning**

The contextual reasoning feature enabled us to generate sets of question that are related to the user, domain, environment or history. In a pure IRT implementation like standardized tests, it only depends on the numerical IRT values. My implementation also depends on the semantics. Generated questions will take into account the subject, property or object of the previous one, the identity of the user or people related to him, and the history of interactions.

- **Performance**

Through the experiments conducted, I was able to show that the techniques used in my work will produce practical dialogues. The reasoning, inference and frequent querying will not have any negative effect on the overall system performance.

7.1 Future Work

- **Abductive reasoning**

Traditional DL reasoners used in my work are all deductive reasoners. They deduce (infer) facts from existing or inferred ones. It would be interesting to see how we can generate dialogs using abductive reasoning, when there is some uncertainty in the basic facts used as an inference or using hypothesis that are not completely known, but do not contradict known facts.

- **Access control using Social Network Information as Context**

The work presented in this dissertation focuses on formal types of policy or knowledge. The rules or axioms (at least the asserted ones) have to be retained in a document. Although not done in this dissertation, it is possible to build an access control system using knowledge or rules that exist in informal data representations such as social networks. This hope is based on the observation that trails, properties, links, photos in someone's, social network accounts like Facebook, Twitter, Google+, etc., can be converted to and interfaced using an ontology and thereafter used to create dialogs for authentication and authorization. I compare this to an electronic fingerprint versus biometric fingerprint. I realize that obtaining access to social networks data is a challenge by itself, but it would be interesting to prove the concept and then address the concerns.

- **Policy and ontology evaluation**

IRT used in my work is built around the individual test item and thus can be used to evaluate policy item (rule) or ontology item (axiom) used as source of questions or knowledge in a dialog. IRT provides procedures to evaluate items themselves while the item is continuously served in tests (dialogs). Item properties can be collected and reviewed for their effectiveness. If it has been noticed that a certain item is always answered right then that item might be considered easy or compromised and thus has to be retired, strengthened or replaced.

- **Risk-based access control**

I have used the ability estimates calculated by IRT to solve problems that has be found to accompany voice systems for access control. I still rely on standard XACML implementation to enforce access policy. The numerical estimate can be used to make more granular access control instead of the grant/deny decisions. A full access decision can be made when the ability estimate is over a predetermined threshold, if not, an appropriate level of access is given. If the user is still asking for more privileges then he has to answer more questions or be redirected to another enforcement mechanism that might involve a human or a traditional access control methodology.

APPENDIX 1: LUBM ONTOLOGY

DATA PROFILE

(Class and property names are underlined. Class names are capitalized and property names in italic.)

In each University

- 15~25 Departments are *subOrgnization* of the University

In each Department:

- 7~10 FullProfessors *worksFor* the Department
- 10~14 AssociateProfessors *worksFor* the Department
- 8~11 AssistantProfessors *worksFor* the Department
- 5~7 Lecturers *worksFor* the Department
-
- one of the FullProfessors is *headOf* the Department
-
- every Faculty *is teacherOf* 1~2 Courses
- every Faculty is *teacherOf* 1~2 GraduateCourses
- Courses taught by faculties are pairwise disjoint
-
- 10~20 ResearchGroups are *subOrgnization* of the Department
-
- UndergraduateStudent : Faculty = 8~14 : 1
- GraduateStudent : Faculty = 3~4 : 1
-
- every Student is *memberOf* the Department
-
- 1/5~1/4 of the GraduateStudents are chosen as TeachingAssistant for one Course
- The Courses the GraduateStudents are TeachingAssistant of are pairwise different
- 1/4~1/3 of the GraduateStudents are chosen as ResearchAssistant
-
- 1/5 of the UndergraduateStudents have a Professor as their *advisor*
- every GraduateStudent has a Professor as his *advisor*
-
- every UndergraduateStudent *takesCourse* 2~4 Courses
- every GraduateStudent *takesCourse* 1~3 GraduateCourses
-
- every FullProfessor is *publicationAuthor* of 15~20 Publications

- every <u>AssociateProfessor</u> is *<u>publicationAuthor</u>* of 10~18 <u>Publication</u>s
- every <u>AssistantProfessor</u> is *<u>publicationAuthor</u>* of 5~10 <u>Publication</u>s
- every <u>Lecturer</u> has 0~5 <u>Publication</u>s
- every <u>GraduateStudent</u> co-authors 0~5 <u>Publication</u>s with some <u>Professor</u>s
-
- every <u>Faculty</u> has an *<u>undergraduateDegreeFrom</u>* a <u>University</u>, a *<u>mastersDegreeFrom</u>* a <u>University</u>, and a *<u>doctoralDegreeFrom</u>* a <u>University</u>
- every <u>GraudateStudent</u> has an *<u>undergraduateDegreeFrom</u>* a <u>University</u>

APPENDIX 2: LUBM STANDARD SPARQL QUERIES

```
# Query1
# This query bears large input and high selectivity. It queries about
just one class and
# one property and does not assume any hierarchy information or
inference.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X
WHERE
{?X rdf:type ub:GraduateStudent .
  ?X ub:takesCourse
http://www.Department0.University0.edu/GraduateCourse0}

# Query2
# This query increases in complexity: 3 classes and 3 properties are
involved. Additionally,
# there is a triangular pattern of relationships between the objects
involved.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X, ?Y, ?Z
WHERE
{?X rdf:type ub:GraduateStudent .
  ?Y rdf:type ub:University .
  ?Z rdf:type ub:Department .
  ?X ub:memberOf ?Z .
  ?Z ub:subOrganizationOf ?Y .
  ?X ub:undergraduateDegreeFrom ?Y}

# Query3
# This query is similar to Query 1 but class Publication has a wide
hierarchy.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X
WHERE
{?X rdf:type ub:Publication .
  ?X ub:publicationAuthor
        http://www.Department0.University0.edu/AssistantProfessor0}

# Query4
# This query has small input and high selectivity. It assumes
subClassOf relationship
```

```
# between Professor and its subclasses. Class Professor has a wide
hierarchy. Another
# feature is that it queries about multiple properties of a single
class.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X, ?Y1, ?Y2, ?Y3
WHERE
{?X rdf:type ub:Professor .
  ?X ub:worksFor <http://www.Department0.University0.edu> .
  ?X ub:name ?Y1 .
  ?X ub:emailAddress ?Y2 .
  ?X ub:telephone ?Y3}

# Query5
# This query assumes subClassOf relationship between Person and its
subclasses
# and subPropertyOf relationship between memberOf and its
subproperties.
# Moreover, class Person features a deep and wide hierarchy.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X
WHERE
{?X rdf:type ub:Person .
  ?X ub:memberOf <http://www.Department0.University0.edu>}


# Query6
# This query queries about only one class. But it assumes both the
explicit
# subClassOf relationship between UndergraduateStudent and Student and
the
# implicit one between GraduateStudent and Student. In addition, it has
large
# input and low selectivity.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X WHERE {?X rdf:type ub:Student}


# Query7
# This query is similar to Query 6 in terms of class Student but it
increases in the
# number of classes and properties and its selectivity is high.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X, ?Y
WHERE
{?X rdf:type ub:Student .
  ?Y rdf:type ub:Course .
  ?X ub:takesCourse ?Y .
  <http://www.Department0.University0.edu/AssociateProfessor0>,
        ub:teacherOf, ?Y}
```

```
# Query8
# This query is further more complex than Query 7 by including one more
property.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X, ?Y, ?Z
WHERE
{?X rdf:type ub:Student .
  ?Y rdf:type ub:Department .
  ?X ub:memberOf ?Y .
  ?Y ub:subOrganizationOf <http://www.University0.edu> .
  ?X ub:emailAddress ?Z}


# Query9
# Besides the aforementioned features of class Student and the wide
hierarchy of
# class Faculty, like Query 2, this query is characterized by the most
classes and
# properties in the query set and there is a triangular pattern of
relationships.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X, ?Y, ?Z
WHERE
{?X rdf:type ub:Student .
  ?Y rdf:type ub:Faculty .
  ?Z rdf:type ub:Course .
  ?X ub:advisor ?Y .
  ?Y ub:teacherOf ?Z .
  ?X ub:takesCourse ?Z}


# Query10
# This query differs from Query 6, 7, 8 and 9 in that it only requires
the
# (implicit) subClassOf relationship between GraduateStudent and
Student, i.e.,
#subClassOf rela-tionship between UndergraduateStudent and Student does
not add
# to the results.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X
WHERE
{?X rdf:type ub:Student .
  ?X ub:takesCourse
<http://www.Department0.University0.edu/GraduateCourse0>}


# Query11
# Query 11, 12 and 13 are intended to verify the presence of certain
OWL reasoning
# capabilities in the system. In this query, property subOrganizationOf
is defined
# as transitive. Since in the benchmark data, instances of
ResearchGroup are stated
# as a sub-organization of a Department individual and the later
suborganization of
# a University individual, inference about the subOrgnizationOf
relationship between
# instances of ResearchGroup and University is required to answer this
query.
# Additionally, its input is small.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X
WHERE
{?X rdf:type ub:ResearchGroup .
  ?X ub:subOrganizationOf <http://www.University0.edu>}


# Query12
# The benchmark data do not produce any instances of class Chair.
Instead, each
# Department individual is linked to the chair professor of that
department by
# property headOf. Hence this query requires realization, i.e.,
inference that
# that professor is an instance of class Chair because he or she is the
head of a
# department. Input of this query is small as well.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X, ?Y
WHERE
{?X rdf:type ub:Chair .
  ?Y rdf:type ub:Department .
  ?X ub:worksFor ?Y .
  ?Y ub:subOrganizationOf <http://www.University0.edu>}


# Query13
# Property hasAlumnus is defined in the benchmark ontology as the
inverse of
# property degreeFrom, which has three subproperties:
undergraduateDegreeFrom,
# mastersDegreeFrom, and doctoralDegreeFrom. The benchmark data state a
person as
# an alumnus of a university using one of these three subproperties
instead of
# hasAlumnus. Therefore, this query assumes subPropertyOf relationships
between
# degreeFrom and its subproperties, and also requires inference about
inverseOf.
```

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X
WHERE
{?X rdf:type ub:Person .
  <http://www.University0.edu> ub:hasAlumnus ?X}


# Query14
# This query is the simplest in the test set. This query represents
those with large input and low selectivity and does not assume any
hierarchy information or inference.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?X
WHERE {?X rdf:type ub:UndergraduateStudent}
```

APPENDIX 3: LUBM CONTEXTUAL REASONING SPARQL QUERIES

```
# queries expressed in SPARQL
# [query ID]
# query
# modified by Mohammad Ababneh, 11/12/13
# Contextual reasoning performance test

[query15]
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?s ?p ?o
 WHERE {
?S ?p <http://www.Department0.University0.edu/GraduateStudent61> .
}

[query16]
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?s ?p ?o
 WHERE {
<http://www.Department0.University0.edu/GraduateStudent61> ?p ?o .
}

[query17]
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?s ?p ?o
 WHERE {{
?s ?p <http://www.Department0.University0.edu/GraduateStudent61> .
}
UNION
{
<http://www.Department0.University0.edu/GraduateStudent61> ?p ?o  .
}}
```

APPENDIX 4: HOMELAND SECURITY CONTEXTUAL REASONING SPARQL QUERIES

```
# queries expressed in SPARQL
# [query ID]
# query
# modified by Mohammad Ababneh, 12/22/13
# Contextual reasoning performance test

[query15]
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?s ?p ?o
 WHERE {
?S ?p <http://www.Department0.University0.edu/GraduateStudent61> .
}

[query16]
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?s ?p ?o
 WHERE {
<http://www.Department0.University0.edu/GraduateStudent61> ?p ?o .
}

[query17]
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#>
SELECT ?s ?p ?o
 WHERE {{
?s ?p <http://www.Department0.University0.edu/GraduateStudent61> .
}
UNION
{
<http://www.Department0.University0.edu/GraduateStudent61> ?p ?o  .
}}
```

REFERENCES

[1] Ababneh, M.,Wijesekera, D., Michael, J. B. (2012). A Policy-based Dialogue System for Physical Access Control. The 7th International Conference on Semantic Technologies for Intelligence, Defense, and Security (STIDS 2012), October 24-25, Fairfax, VA

[2] Ababneh, M.,Wijesekera, D. (2013). Dynamically Generating Policy Compliant Dialogues for Physical Access Control. CENTERIS2013 - Conference on Enterprise Information Systems – aligning technology, organizations and people, Lisbon, Portugal. October 23-25, 2013.

[3] Ababneh, M.,Wijesekera, D. (2013). An Ontological Inference Driven IVR System. The 8th International Conference on Semantic Technologies for Intelligence, Defense, and Security (STIDS 2013), Fairfax, VA, November 12-15, 2013.

[4] Bishop, M. (2002). Computer Security: Art and Science. Addison Wesley.

[5] Baker, F. B. (2001). The basics of item response theory. ERIC Clearinghouse on Assessment and Evaluation.

[6] Baker, F. B. (2004). Item response theory: Parameter estimation techniques, vol. 176, CRC.

[7] Weiss, D. J., Kingsbury, G. G. (1984). Application of computerized adaptive testing to educational problems. Journal of Educational Measurement.

[8] Wainer, H. (2000). Computerized Adaptive Testing: A Primer. Second Edition, Lawrence Erlbaum Associates Publishers

[9] Massie, T., Wijesekera, D. (2008). TVIS: Tactical Voice Interaction Services for Dismounted Urban Operations. Proceeding of Military Communications Conference, IEEESan Diego, Calif., Nov. 17-19,pp. 258-264.

[10] Faresi, A., Wijesekera, D., (2011). Preemptive Mechanism to Prevent Health Data Privacy Leakage. Proceedings of International Conference on Management of Emergent Digital EcoSystems, ACM, pp. 17-24, San Francisco, CA.

[11] Chongshan Ran and Guili Guo, "Security XACML Access Control Model Based On SOAP Encapsulate," International Conference on Computer Science and Service System, IEEE, pp. 2543-2546, Nanjing, China, 27-29 June 2011.

[12] Ardagna, C.A., De Capitani di, Vimercati, S., Paraboschi, S., Pedrini, E., Samarati, P., Verdicchio, M. , "Expressive and Deployable Access Control in Open Web Service Applications," IEEE Transactions on Services Computing, Volume 4, Issue 2, pp. 96-109, April-June 2011.

[13] Security Assertion Markup Language (SAML), SAML 2.0 profile of XACML v2.0, available at URL: http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-saml-profile-spec-os.pdf

[14] Marouf, S.,Shehab, M., Squicciarini, A., Sundareswaran, S. (2011). Adaptive Reordering and Clustering-Based Framework for Efficient XACML Policy Evaluation. IEEE Transactions on Services Computing, Vol. 4, No. 4.

[15] Liu, A., X., Chen, F., Hwang, J., Xie T. (2008), XEngine: A Fast and Scalable XACML Policy Evaluation Engine. Proceeding ACM SIGMETRICS Int'l Conference, Measurement and Modeling of Computer Systems, pp. 265-276

[16] Liu, L., Qi, Q., Li, F., (2010). Ontology-Based Interactive Question and Answering System, Internet Technology and Applications, International Conference on, pp. 1–4.

[17] Wang, D.S., (2010).A Domain-Specific Question Answering System Based on Ontology and Question. 11th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing.

[18] Krummenacher, R., Strang, T. (2007.). Ontology-Based Context-Modeling. In Third Workshop on Context Awareness for Proactive Systems (CAPS'07).

[19] Pessoa, R.M., Calvi, C.Z., Filho J.P., de Farias C.G., Neisse,R., (2007). Semantic context reasoning using ontology based models, Dependable and Adaptable Networks and Services, 13th Open European Summer School and IFIP TC6.6 Workshop (EUNICE), vol. 4606, Springer-Verlag, Germany, pp. 44–51

[20] Wang, X.H, Gu, T., Zhang, D.Q. Pung, H.K. (2004). Ontology based context modeling and reasoning using OWL, Proceedings of Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, IEEE Computer Society

[21] Dey, A.K. (2001). Understanding and Using Context. Personal and Ubiquitous Computing, 5(1):4–7.

[22] Beamon, B., Kumar, M., (2010). HyCoRE: Towards a generalized hierarchical hybrid context reasoning engine. Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on , vol., no., pp.30,36,

[23] W3C Voice Browser Working Group, URL: http://www.w3.org/Voice, accessed September 30, 2013.

[24] W3C, Voice Extensible Markup Language (VoiceXML)(VXML), URL: http://www.w3.org/Voice/, accessed September 30, 2013.

[25] Voxeo web site, URL: http://www.Voxeo.com, accessed September 30, 2013.

[26] W3C, Web Ontology Language (Primer), http://www.w3.org/TR/owl2-primer/, accessed September 30, 2013.

[27] Gruber, T. R. (1993). A translation approach to portable ontologies. Knowledge Acquisition, 5(2):199-220.

[28] W3C, The Resource Description Framework (RDF), URL:http://www.w3.org/TR/rdf-primer/ accessed September 30, 2013.

[29] W3C, SPARQL Protocol and RDF Query Language, URL: http://www.w3.org/2009/sparql/, accessed September 30, 2013.

[30] Hermit reasoner, University of Oxford, URL: http://hermit-reasoner.com/, accessed September 30, 2013.

[31] The Web Ontology Language OWLAPI, URL: http://owlapi.sourceforge.net/reasoners.html, accessed September 30, 2013.

[32] Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A. (2010). A survey of context modelling and reasoning techniques, Pervasive and Mobile Computing, 6, pp. 161–180.

[33] W3C, Delivery Context Ontology, URL: http://www.w3.org/TR/dcontology/, accessed September 30, 2013.

[34] Eclipse organization, Higgins Personal Data Service, URL: http://www.eclipse.org/higgins/, accessed September 30, 2013.

[35] XACML, OASIS, URL: https://www.oasis- open.org/committees/tc_home.php?wgabbrev = xacml, accessed September 30, 2013.

[36] Document Object Module, URL: http://www.w3.org/DOM/, accessed September 30, 2013.

[37] Extensible Markup Language (XML), URL: http://www.w3.org/XML/, accessed September 30, 2013.

[38] OASIS, Security Assertion Markup Language (SAML), SAML 2.0 profile of XACML v2.0, available at URL: http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-saml-profile-spec-os.pdf, accessed September 30, 2013.

[39] Concerto IRT Platform, URL: http://www.psychometrics. cam.ac.uk/page/338/concerto-testing-platform, accessed September 30, 2013.

[40] W3C, Speech Grammar Recognition Specification, URL: http://www.w3.org/TR/speech-grammar/, accessed September 30, 2013.

[41] Apple Siri, URL: http://www.apple.com/ios/siri, accessed September 30, 2013.

[42] Google Android Mobile Search, URL: http://www.google.com/mobile/search/, accessed September 30, 2013.

[43] Microsoft Windows Phone Speech, URL: http://www.windowsphone.com/en-us/how-to/wp7/basics/use-speech-on-my-phone, accessed September 30, 2013.

[44] Ababneh, M.,Wijesekera, D., Costa, C.G., Paulo. (2014). Interactive Voice Response based Identity Establishment. Special Issue on Identity Protection and Management, Elsevier Information Security Technical Report (ISTR) Journal, March, 2014, (under review).

[45] Lehigh University BenchMark (LUBM), (2013), URL:http://swat.cse.lehigh.edu/projects/lubm/profile.htm

[46] Guo, Yuanbo, Pan, Zhengxiang and Heflin, Jeff . LUBM: A Benchmark for OWL Knowledge Base Systems. Web Semantics. 3(2) July 2005. pp.158-182.

[47] Wang, Sui-Yu, Guo, Yuanbo, Qasem, Abir and Heflin, Jeff . Rapid Benchmarking for Semantic Web Knowledge Base Systems. In Proc. of the 4th International Semantic Web Conference (ISWC2005). Galway, Ireland. 2005. pp.758-772.

[48] Guo, Yuanbo, Pan, Zhengxiang and Heflin, Jeff . An Evaluation of Knowledge Base Systems for Large OWL Datasets. Third International Semantic Web Conference, Hiroshima, Japan. Springer. 2004. pp.274-288.

[49] OWLIM Benchmarking, (2013), URL: http://www.ontotext.com/owlim/ benchmark-results/lubm.

[50] OWLIM Performance on the Amazon Cloud, (2013), URL: http://www. ontotext. com/owlim/ benchmark-results/cloud-performance

[51] Berlin SPARQL Benchmark (BSBM) Specification, (2013), URL: http://wifo5-03.informatik.uni-mannheim.de/bizer/berlinsparqlbenchmark /spec/index.html, Germany.

[52] BSBM Results, (2009), BSBM Results for Virtuoso, Jena TDB, BigOWLIM URL: http://wifo5-03.informatik.uni-mannheim.de/bizer/berlinsparqlbenchmark /results/V5/index.html

BIOGRAPHY

Mohammad Ababneh finished his high school diploma from Irbid Secondary School, Irbid, Jordan in 1990. He had earned a bachelor degree in Computer Science from Mu’tah University, Jordan in 1994. He had earned a dual Masters degree in Computer Science and Information Technology Management from the Naval PostGraduate School, Monterey, CA in 2000. He had been a Ph.D. Student at George Mason University since 2009. He is graduating with a Ph.D. in Information Technology in 2014. He had been working for The Royal Jordanian Air Force since 1994. While pursuing his Ph.D., he had continuously been a research assistant investigating various topics in the areas of Information Security and Assurance, Command and Control, Semantic Web and Information Systems.