Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 44 (2015) 275 – 284

Procedia Computer Science

2015 Conference on Systems Engineering Research

Developing Ontologies and Persona to Support and Enhance Requirements Engineering Activities – A Case Study

Wee Wee Sim[a]*, Dr. Peggy Brouse[b]

[a]George Mason University, The Volgenau School of Engineering, Fairfax, Virgina, 22039, USA
[b]George Mason University, Department of Systems Engineerinsg and Operations Research, Fairfax, Virgina, 22039, USA

Abstract

This paper provides an insight into incorporating persona concept and developing ontologies to support requirements engineering activities via a university course registration web application system case study. The objectives are to examine (1) how the concept of persona, in the context of the concepts of viewpoint, goal, scenario, task, and requirement, may be integrated in a unified environment to enable stakeholders and developers gain a better understanding of target users' needs and behaviors and identify missing requirements early in the requirements engineering process, and (2) how the concepts and their relationships may be explicitly specified ontologically to help establish a knowledge repository and foster a shared common understanding of target users' needs and behaviors among developers and stakeholders during the requirements analysis and modeling activity. A five-step iterative ontology development process is developed to help guide developers in the process of building the ontologies for the case study. We present the persona and viewpoint documents created and the ontology specifications specified in Protégé-Frames via applying our ontology development process.

Peer-review under responsibility of the Stevens Institute of Technology.

Keywords: Ontology; Persona; User Profile; User Modeling; Requirements Engineering; Systems Engineering, Knowledge Engineering.

1. Introduction

It has been widely acknowledged that one of the contributing factors in requirements engineering projects' failure is the lack of a comprehensive shared understanding of target users' needs and behaviors to achieve projects' requirements[1,2]. There is a lack of semantic agreement among users that hinders the requirements engineering activities. Poor or inadequate understanding of users' requirements increases the chance of not meeting users' needs. Focusing solely on the tasks or functionalities without considering the target users' needs, behaviors, and goals is a recipe for failure. Well understanding, explicit formal specification, and common sharing of users' knowledge and information are crucial in the success of the requirements engineering projects.

The concept of persona, originally presented by Alan Cooper[3] who, focused on the use of personas, their goals

* Corresponding author. *E-mail address*: wsim@gmu.edu.

Peer-review under responsibility of the Stevens Institute of Technology.
doi:10.1016/j.procs.2015.03.060

and scenarios on design, is becoming a promising and an emergent new paradigm in user requirements modeling. Personas are fictitious, specific, and concrete representations of target users[3]. They are constructed to resemble real people, i.e. they contain information such as names, ages, educational backgrounds, occupations, skills, goals, concerns, environments, usage patterns on the system, and so forth. Personas capture rich behavior model of users and can help requirements engineers to obtain deeper understanding of the target users and make better design decisions based on these personas.

The nature of requirements engineering involves capturing knowledge from multiple sources. The field of knowledge representation, commonly known as ontology, is a formal representation of the entities and relationships exist in some domain of interest. According to Gruber[4], an ontology is a formal, explicit specification of a shared conceptualization. An ontology is all about defining the domain vocabularies, the essential concepts in the domain, their classifications, taxonomies (concept hierarchies), relationships among the concepts (including constraints), and domain axioms related to a particular application domain. Ontologies not only offer knowledge representation and interrelating different types of knowledge, but also provide constraint checking on the ontology and inference mechanism to detect inconsistency and incompleteness in requirements description. Ontology-based approach thus offers a good choice to represent knowledge about users, such as users' behaviors, scenarios, tasks, goals, and requirements. Over the past several years, there have been several efforts conducted by researchers on scenarios[5,6,7] and goal[8,9,10] modeling, as well as ontology-based scenarios and goal requirements modeling[11,12]. Few researchers have proposed techniques to identify personas and investigate their relationships with scenarios and goals[13,14,15]. Some researchers have designed Personal or User Profile ontology to represent and model user profiles[16,17]. To the best of our knowledge and as of this writing, there have been no efforts conducted in using an ontology-based approach to provide an explicit specification and representation of personas in the context of viewpoints, scenarios, tasks, goals, and requirements in requirements engineering on web application domain.

In our earlier research efforts, we have developed a Concept Development Process (CDP) model[18] and an Ontology-Based Persona-Driven User Requirements Modeling (OntoPersonaURM) model[19], with the goals of examining (1) how the concept of persona, in the context of the concepts of viewpoint, goal, scenario, task, and requirement, may be integrated in a unified environment to enable stakeholders and developers gain a better understanding of target users' needs and behaviors and identify missing requirements early in the requirements engineering process and (2) how the concepts and their relationships may be explicitly specified using an ontology approach to help establish a knowledge repository and foster a shared common understanding of target users' needs and behaviors among developers and stakeholders during the requirements analysis and modeling activity. The CDP model is developed to help guide requirement engineers and developers in the development of the concepts and the integration of concepts into the requirements engineering process. The CDP model consists of four main processes: 1–Personas Construction, 2–Viewpoints Identification and Construction, 3–Concepts Modeling, and 4–Analysis and Evaluation. In the Concepts Modeling process of the CDP model, the OntoPersonaURM model is developed to provide insights and help guide ontology engineers and developers into the construction of ontologies for explicit specifications of the concept of persona in representing users' characteristics, and the concepts of viewpoint, goal, scenario, task, and requirement. The OntoPersonaURM model is composed of three generic interrelated ontologies: (1) Persona Ontology: covers general concepts pertaining to person characteristics including education, abilities, interests, knowledge, viewpoints, environments, and so forth. (2) Behavioral-GST (**G**oal-**S**cenario-**T**ask) Ontology: captures and defines the needs and behaviors of the personas and the system-to-be, i.e. viewpoint, goal, scenario, and task concepts. (3) Requirements Ontology: specifies general concepts for the representation of the requirements and their properties. We chose Protégé-Frames ontology editing tool as a concept representation environment for the construction of ontologies, as Protégé-Frames (1) has an intuitive and easy-to-use graphical user interface that does not demand too much learning curve, and (2) provides our research needs of defining classes (and sub-classes), describing properties and relationships of classes, populating classes with instances, and performing queries to check constraints on the classes. Within the OntoPersonaURM model, a five-step iterative ontology development process has also been developed to help guide engineers in the process of building the ontologies. We strongly encouraged readers to review both of our papers[18,19] to get to know more about the CDP and OntoPersonaURM models. This paper demonstrates the application of these models on a course registration web application system case study.

The paper is organized as follows: section 2 summarizes the ontology development process and provides walkthroughs of the application of the ontology development process steps in the chosen case study. Section 3 demonstrates constraints checking on the developed ontologies. Finally, section 4 addresses the conclusion and highlights future research directions.

2. Ontology Development Process

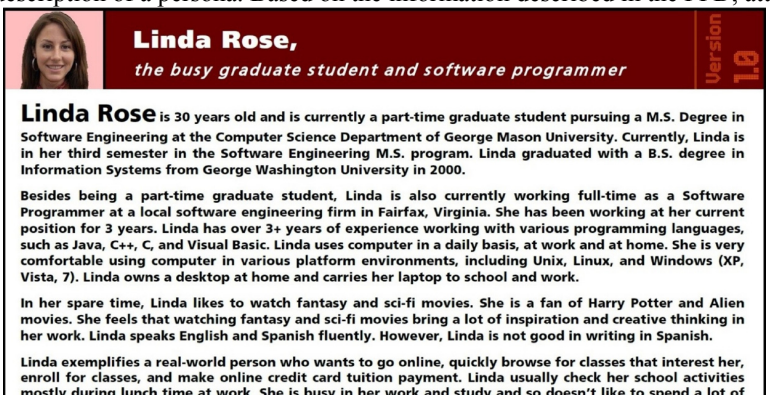
It is to be emphasized that developing a new ontology is often tedious and time consuming; it normally requires engineers and developers to have sufficient knowledge in ontology specifications and familiar with ontology development environment. There is no single correct ontology for any domain[20]. In building our ontologies for the OntoPersonaURM model on the case study, we consulted with the guidelines suggested in [20]. We outline our five-step iterative ontology development process in Table 1. We provide walkthroughs of the application of the ontology development process on the chosen case study in the subsequent sub-sections.

Table 1. Ontology Development Process

Ontology Development Process	Description
Step 1: Synthesize Information Collected	Information gathered and described in the persona and viewpoint documents[18] created through collaboration with marketing analysts, ontology engineers, and requirement engineers during the requirements elicitation process are analyzed and synthesized. Terms extracted from these documents are candidates for the definition of classes and properties in the ontology(ies).
Step 2: Consult Existing Ontologies	There are extensive libraries of reusable ontologies available on the Web. For examples, the Protégé ontology library[21] maintains a good collection of ontologies, the DAML ontology library[22], user profile ontology[17], personal ontology[16], and so forth. As building a new ontology from scratch is a time consuming process, if an existing solution ontology is available and is relevant to the application domain in hand, then it is suggested to consult with the existing ontology to determine if we can reuse, refine, or extend existing classes and properties.
Step 3: Define Classes and Properties	A top-down approach is adopted to define the class hierarchy (super-sub-class), i.e. from most general concepts to specialized concepts. A set of potential classes, class hierarchy, and class properties (i.e. attributes, cardinalities, and relationships with other classes) are identified, defined, and specified in the Protégé-Frames editor tool. This step occupies the most time.
Step 4: Create Instances	Instances of the classes are created in the Protégé-Frames editor. Creating class instances can help to correct mistakes and fine-tune the classes and properties in the ontology.
Step 5: Combine Ontologies	If one or more relationships exist between classes of two ontologies, the ontologies are combined by including the related ontology into the current ontology via Protégé-Frames' Manage Included Projects menu[23]. For example, in the OntoPersonaURM model, the Persona Ontology contains classes that have relationships with classes of the Behavioral-GST Ontology, thus the Behavioral-GST Ontology is included in the Persona Ontology. Combining related ontologies help ontology engineers better understand the relationships of classes between ontologies, identify conflicts, and make necessary changes. Ontology engineers may need to revisit one or more previous steps to refine the ontologies.

2.1. Step 1: Synthesize Information Collected

This step is closely related to the Concept Development Process (CDP) model proposed in our earlier work[18], which we recommend readers to review to get to know more detail about the CDP model. For our case study, the primary target users are students that use the system to browse and register for courses, check course grades, review financial aid information, pay tuition and fees, and so forth. The secondary users are the application developers and site administrator. In this paper, we focus on the primary users. A primary persona, "Linda Rose, the busy graduate student and software programmer" is created. We also created three documents: Persona Profile Document (PPD), Persona Definition Document (PDD), and Viewpoint Document (VPD)[18]. The PPD is a one-page narrative description of a persona. Based on the information described in the PPD, attributes are extracted to form a PDD that defines the persona. From the PDD created, a VPD is created that contains information defining one or more views at a certain level of detail and addressing certain design concerns by the persona playing a particular role in a particular environment. The attributes are potential candidates for the definition of classes in the ontologies of the OntoPersonaURM model. The PPD, PDD, and VPD are shown in Fig. 1, 2, and 3 respectively.


Linda Rose,
the busy graduate student and software programmer
Version 1.0

Linda Rose is 30 years old and is currently a part-time graduate student pursuing a M.S. Degree in Software Engineering at the Computer Science Department of George Mason University. Currently, Linda is in her third semester in the Software Engineering M.S. program. Linda graduated with a B.S. degree in Information Systems from George Washington University in 2000.

Besides being a part-time graduate student, Linda is also currently working full-time as a Software Programmer at a local software engineering firm in Fairfax, Virginia. She has been working at her current position for 3 years. Linda has over 3+ years of experience working with various programming languages, such as Java, C++, C, and Visual Basic. Linda uses computer in a daily basis, at work and at home. She is very comfortable using computer in various platform environments, including Unix, Linux, and Windows (XP, Vista, 7). Linda owns a desktop at home and carries her laptop to school and work.

In her spare time, Linda likes to watch fantasy and sci-fi movies. She is a fan of Harry Potter and Alien movies. She feels that watching fantasy and sci-fi movies bring a lot of inspiration and creative thinking in her work. Linda speaks English and Spanish fluently. However, Linda is not good in writing in Spanish.

Linda exemplifies a real-world person who wants to go online, quickly browse for classes that interest her, enroll for classes, and make online credit card tuition payment. Linda usually check her school activities mostly during lunch time at work. She is busy in her work and study and so doesn't like to spend a lot of

Fig. 1. Persona Profile Document (PPD) sample (partial view) for primary persona Linda Rose

Linda Rose,
the busy graduate student and software programmer

Primary Persona | PRIORITY

DEMOGRAPHIC PROFILE

NAME				GENDER	AGE	MARITAL STATUS	HIGHEST EDUCATION LEVEL	OCCUPATIONS				SOCIAL LEVEL
First	Last	Nickname	Title					Job Title	Work Status	Length of Employment	Salary	
Linda	Rose	Rosie		Female	25-34	Single	B.S. degree	Software programmer	Full-time	3 years	$50K - $80K	Middle-class

LANGUAGE PROFICIENCY			INTERESTS
Listening	Speaking	Writing	
• English – excellent • Spanish – good	• English – excellent • Spanish – good	• English - good • Spanish – fair	• Listen music • Play guitar • Surf the web • Shop online

ABILITIES

Physical				Cognitive				
Motor	Sensory-Hearing	Sensory-Speech	Sensory-Visual	Attention	Memory	Problem-Solving	Reading-Comp.	Visual-Comp.	Spatial
Average					Average	High	Average	Average	

KNOWLEDGE & SKILLS PROFILE

COMPUTER
- Software (office applications) – excellent
- Hardware – good
- Programming (Java, C++, Visual Basic) – good
- Operating systems (Windows XP/Vista/7, Linux, Unix) – good

WEB
- Internet – good
- Social media – good

DOMAIN
- UML – good
- Ontology – good
- Requirements engineering – good

INTERACTION PROFILE

ROLES	GOALS	CONCERNS
Part-time graduate student	• Browse courses with fast response • Enroll courses with ease • Make online tuition payment with ease	• Will the system be available for service when needed? • Will the public WiFi connection be available when needed? • Will the public WiFi connection be secured? • Will the website interface, layout, and information be properly presented on cell phone? • Will the online payment function properly?

ENVIRONMENTS

Location	Time of Day	Duration	Frequency (low: 0-5 times/wk, medium: 6-10 times/wk, high: >10 times/wk)	Attitudes (eager, skeptical, cautious, indifferent, impatient, resentful, curious, trusting, others)
Starbucks	Mid-work day	15 mins	Medium	Eager, skeptical, cautious, impatient
Home	Evening	40 mins	Medium	Eager, trusting

USAGE PATTERNS (scenarios titles)	TOOLS	USABILITY PREFERENCES (accuracy, attractiveness, efficiency, learnability, reliability, comprehensibility, clarity, rememberability)
• Browse courses • Register courses • Check student schedule • Review student records • Review financial aid • Make online payment	• Laptop • Desktop • Cell phone • Tablet	• Accuracy – important • Attractiveness – not at all important • Efficiency – important • Learnability – somewhat important • Comprehensibility – important • Clarity – important • Rememberability – somewhat important

Fig. 2. Persona Definition Document (PDD) sample for primary persona Linda Rose

<Linda Rose, Part-Time Graduate Student, Starbucks (mid-work day)> | **VIEWPOINT (VP) NAME**

SOURCES	STAKEHOLDERS	HISTORY (VP-version, date, author, changes)	ROLE
• Persona Profile Document (PPD) • Persona Definition Document (PDD)	• Administrators • Billing system • Instructors	VP-v1.0, 10-27-2013, John Deer	Part-time graduate student

ENVIRONMENT (location, time of day, duration, tools, etc): • Starbucks coffee shop • Mid-day (lunch work time) • Limited available time (~15 mins) • Cell phone (internet WiFi capable) • Heavy human traffic • Rely on WiFi connection • Security and privacy issues

GOALS
- Browse courses with fast response
- Enroll courses with ease

CONCERNS
- The accessibility of internet (WiFi) connection
- The security and privacy of public internet (WiFi) connection
- The availability of the system when needed
- The presentation of website interface, layout, and information on cell phone

SCENARIOS, TASKS

Scenarios (titles)	Tasks (titles)
• Browse courses	• Find a specific course • Browse course registration catalog
• Register courses	• Find a specific course • Browse course registration catalog • Add course to schedule • Select specific course
• Check student schedule	....
• Review student records	....

REQUIREMENTS

Functional	• The system shall enable student to query for information based upon course name, course code, instructor name, and department • The system shall display available courses to the student based on student's query criteria • The system shall accept student's course registrations • The system shall validate based upon course availability • The system shall enable student to change course registration information • The system shall enable student to view her class schedule • The system shall display message if student's request does not succeed • The system shall display message if student's course registration does not succeed
Non-Functional	• The system shall be easy-to-use with understandable navigational design and self-explanatory instructions • The system shall display page layout and information appropriately for smaller screen size devices such as mobile phones and tablets • The system shall ensure that all confidential information be encrypted to maintain security • The system shall conform to the security standards for mobile devices as published in "NIST Special Publication 800-124 Revision 1" • The system shall be capable of providing frequent auto-save backup and recovery information in the event of lost network connection • The system shall be compatible with different operating systems (Windows, Mac) and platforms (PC, Mac, mobile phone, tablets)

MODEDLING

Modeling Techniques	Concepts Representation	Models, Ontologies
• UML Class Diagram • UML Instance Diagram	• Ontology tool (Protégé-Frames)	• Persona Class Diagram, Persona Instance Diagram, Behavioral-GoalScenarioTask Class Diagram, Behavioral-GoalScenarioTask Instance Diagram, Requirements Class diagram, Requirements Instance Diagram • Persona Ontology, Behavioral-GoalScenarioTask Ontology, Requirements Ontology

Fig. 3. Viewpoint Document (VPD) sample for primary persona Linda Rose

For our case study, it is worth highlighting that the primary persona ("Linda Rose, the busy graduate student and software programmer") uses the course registration system to browse and register for courses in two different situations: 1) In a public place like Starbucks coffee shop in mid-day lunch hour during work using her cell phone with WiFi internet connection. 2) At home during evening time using her desktop with strong and reliable internet connection. Therefore, we can identify two viewpoints, VPs (or Viewpoint Block, VPB[18]): VP1 as <Linda Rose, Part-Time Graduate Student, Starbucks (mid-work day)> and VP2 as <Linda Rose, Part-Time Graduate Student, Home (evening)>. These two viewpoints have slightly different goals, concerns, scenarios, goals, and requirements, as the environments are different: a public place where internet connection depends on the availability and reliability of a strong WiFi connection, security and privacy issues for payment using a credit card, and the constraint of time (during lunch work time); and a comfortable home environment where internet connection is available and reliable, less concern for security and privacy, and there is no constraint on time. Thus, the development team can identify crucial requirements that may not have been included in the original requirement sets, e.g. non-functional requirements (Fig. 3) pertaining to mobile phone information presentation layout, security, and privacy. The VPD for viewpoint VP1 < Linda Rose, Part-Time Graduate Student, Starbucks (mid-work day)> is shown in Fig. 3.

2.2. Step 2: Consult Existing Ontologies

In constructing the ontologies for our OntoPersonaURM model on the case study, there appear to be no existing ontologies that are similar and are readily available for reusable purpose. However, we have consulted some existing ontologies[11,16,17] on the approaches in specifying some of the class properties, for examples, Persona class, Interest class, Goal class, and Requirement class.

2.3. Step 3: Define Classes and Properties

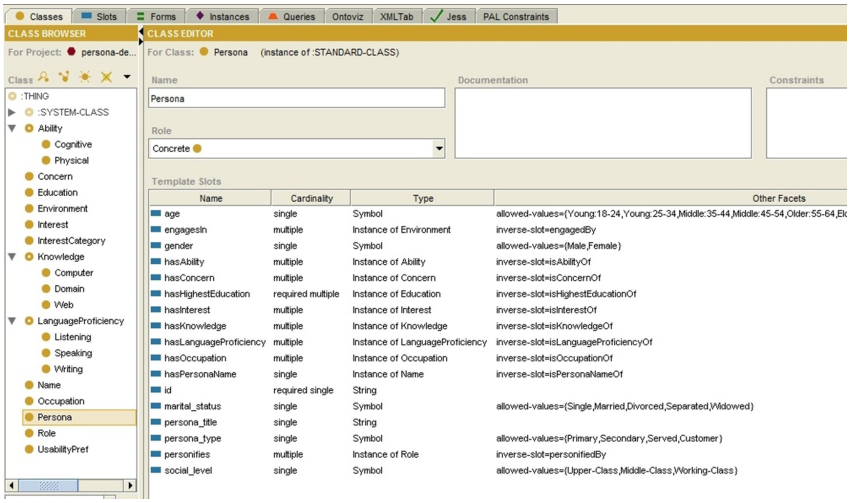


Fig. 4. Persona Ontology (Persona concept) – Protégé-Frames

***Persona Ontology*:** The Persona Ontology provides a comprehensive set of general concepts pertaining to person characteristics and environment. The concepts in the Persona Ontology not only capture the basic characteristics and preferences of a person such as age, gender, name, education, occupation, abilities, expertise, interests and so forth, but also the relationships to the environment in which the person engages in. For a detailed description of the concepts, we suggest that readers refer to our earlier paper[19]. Due to space limitation, we have omitted showing the UML Class Diagram, which can be found in our earlier paper[19]. A representative of the specification of the Persona concept in Protégé-Frames is shown in Fig. 4. In Protégé-Frames tool, classes are specified in the "Class Browser" (left pane of the tool) and properties are specified in the "Class Editor" (right pane of the tool).

There are three areas to highlight in regards to the Persona Ontology:

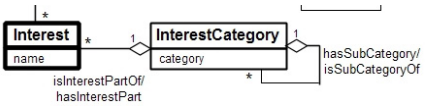


Fig. 5. Interest Class

(1) The Interest class (Fig. 5) is associated with the InterestCategory class via aggregation, i.e. an InterestCategory class is an aggregation of an Interest class. As it is common that there are various interest names that may belong to a same interest category, we chose to place a one-to-many relationship between InterestCategory and Interest classes, i.e. an interest category hasInterestPart one or more interests. For example, an interest category "*Entertainment*" hasInterestPart "*Listening music*", "*Playing guitar*". For Simplicity, we chose to specify that an interest is part of one and only one interest category. For example, an interest "*Listening music*" isInterestPartOf "*Entertainment*". The cardinalities between Interest and InterestCategory classes (and vice versa) is a design choice and thus may be modified by the ontology designer.

(2) Some entities in the Persona class are represented as separate distinct classes (via associations) rather than attributes as these entities have internal structures or complex data types that may be useful for the ontology designer to apply validation or formatting rules to be recognized by the ontology reasoner. For examples, a person's name is represented as a Name class, since it contains internal structures such as first_name, last_name, middlename, title, and nickname; a person's education is represented as an Education class as it captures internal structures such as degree_year, degree_title, highest_education_level. If these entities (person's name, education, occupation) were represented as attributes of String type in the Persona class, then the internal structure of these complex data values and their semantics could be lost and thus could not be made available in the ontology for further processing, filtering, sorting, etc. The decision to represent an entity as a class or an attribute is a design choice to be decided by the ontology designer, based on the application domain in hand.

(3) The Environment class is represented in the Persona Ontology rather than the Behavioral-GST Ontology, since the Environment class is directly related with several classes in the Persona Ontology, namely, Persona, Role, and Concern classes. Constraint check and query execution on the classes in the Persona Ontology can thus be executed easily with an appropriate plugin tool such as PAL[24].

***Behavioral-GST Ontology*:** The Behavioral-GST Ontology captures the behavior of the system-to-be. The main concepts are the viewpoint, goal, scenario, and task concepts. The Behavioral-GST Ontology is related to the Persona Ontology via the Viewpoint class (Fig. 6) of the Behavioral-GST Ontology and the Requirements Ontology via the Goal and Scenario classes (Fig. 7) of the Behavioral-GST Ontology. For a detailed description of the properties of the classes, we suggest that readers refer to our earlier paper[19]. Due to space limitation, we have omitted showing the UML Class Diagram, which can be found in our earlier paper[19]. A representative of the

specification of the Goal concept in Protégé-Frames is shown in Fig. 8.

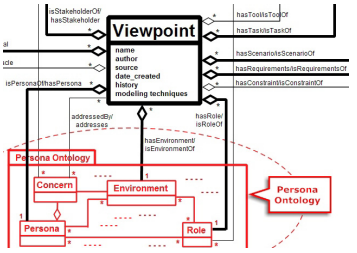


Fig. 6. Viewpoint Class

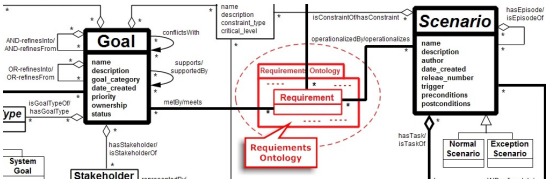


Fig. 7. Goal and Scenario Classes

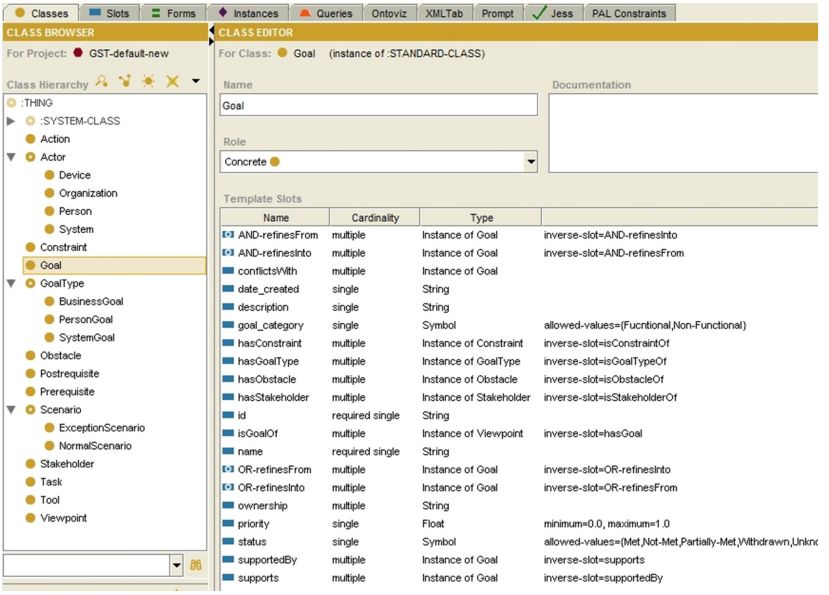


Fig. 8. Behavioral-GST Ontology (Goal concept) – Protégé-Frames

***Requirements Ontology*:** The Requirements Ontology contains general concepts that are considered applicable to most domains for the representation of the requirements and their properties. The central class is the Requirement class which defines typical yet comprehensive set of requirement properties. Supporting classes include RequirementCategory class and SRS (Systems or Software Requirements Specification) class. The attributes of the Requirement class are by no means exhaustive. For a detailed description of the classes, we suggest that readers refer to our earlier paper[19]. Due to space limitation, we have omitted showing the UML Class Diagram, which can be found in our earlier paper[19]. A representative of the specification of the Requirement concept in Protégé-Frames is shown in Fig. 9.

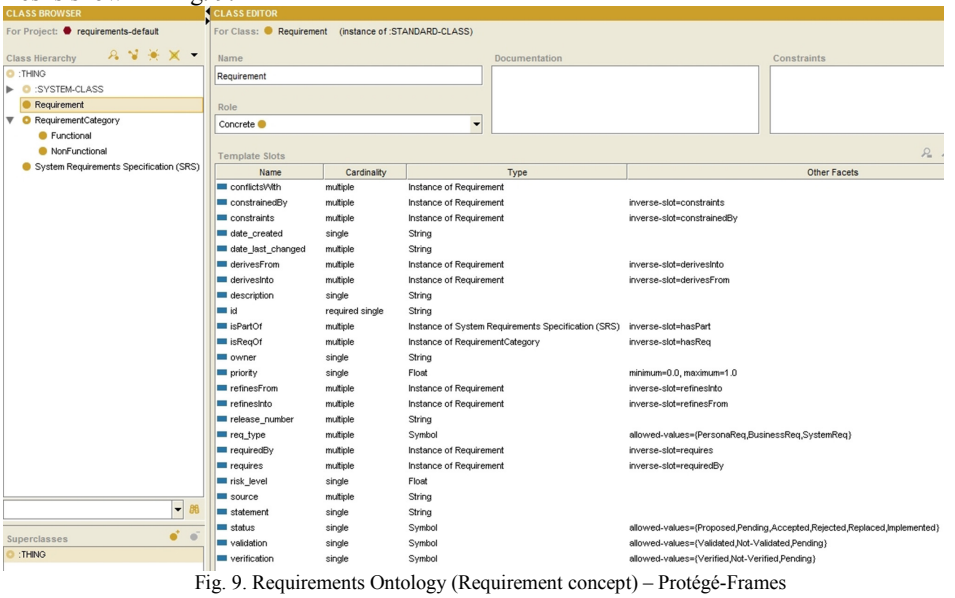


Fig. 9. Requirements Ontology (Requirement concept) – Protégé-Frames

2.4. Step 4: Create Instances

After the classes are defined and specified in Protégé-Frames, instances of the classes are created in Protégé-Frames tool. A representative name for an instance of a class is chosen and displayed in the "Instance Browser". The instance name may be selected from any one or combination of the values of the properties (or slots in Protégé-Frames). Values for the properties are filled in in the "Instance Editor" of the Protégé-Frames tool. One or more instances may be created for a class. In our case study, for example, we created an instance for the Persona class and selected the value of the attribute persona_title ("Linda Rose, The Busy Graduate Student & Software Programmer") as the representative instance name displayed in the "Instance Browser". An alternative for the instance name could be the id attribute. Due to space limitation, we show in Fig. 10 the instance created for the Persona concept of the Persona Ontology in Protégé-Frames. The instances created in Protégé-Frames for the Behavioral-GST Ontology and the Requirements Ontology can be found in [19].

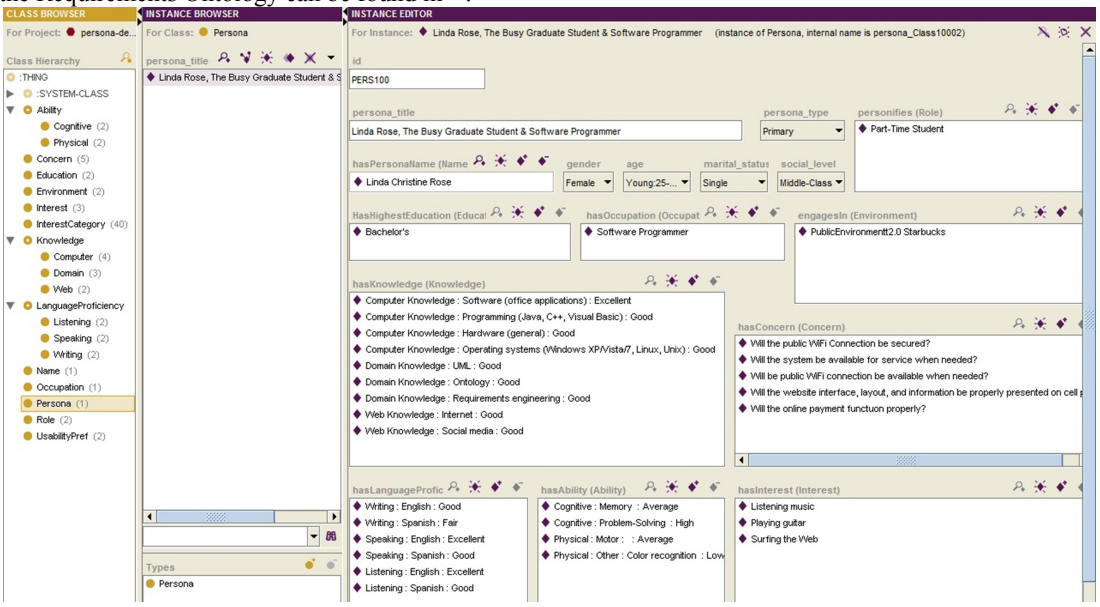


Fig. 10. Persona Ontology – Persona instance in Protégé-Frames

2.5. Step 5: Combine Ontologies

The final step in our ontology development process is to combine related ontologies to help to better understand the relationships of classes between ontologies and make necessary corrections to refine the ontologies. If there exist one or more relationships between classes of two ontologies, then the ontologies are combined by including the related ontology into the current ontology via the "Manage Included Projects" selection of the "Project" menu in the Protégé-Frames tool (Fig. 11a, 11b). The included classes and properties are displayed in Protégé-Frames as pale icons to distinguish from the classes in the current ontology. An included ontology may also be merged with the current ontology to form a single ontology via the "Merge Included Projects" and all classes and properties of the merged ontologies are then displayed as solid icons.

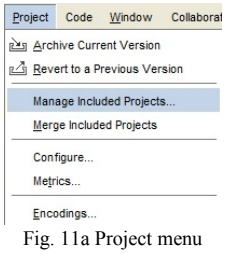


Fig. 11a Project menu

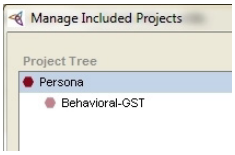


Fig. 11b. Manage Included Projects

In our case study, for the Persona Ontology, since it contains classes that have established relationships with classes of the Behavioral-GST Ontology, we decided to include the Behavioral-GST Ontology in the Persona Ontology. In a similar fashion, as the Behavioral-GST Ontology is related with the Persona Ontology and the Requirements Ontology, we included the Persona Ontology and Requirements Ontology in the Behavioral-GST Ontology. For the Requirements Ontology, we included the Behavioral-GST Ontology in the Requirements Ontology. Fig. 12, 13, and 14 are snapshots of the combined ontologies for Persona Ontology, Behavioral-GST Ontology, and Requirements Ontology respectively.

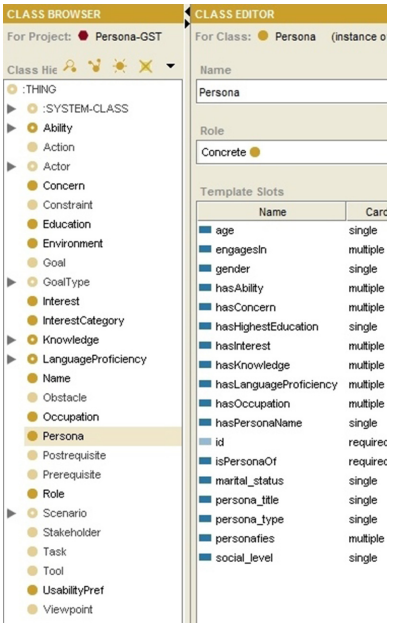


Fig. 12. Combined Persona Ontology and Behavioral-GST Ontology – Protégé-Frames

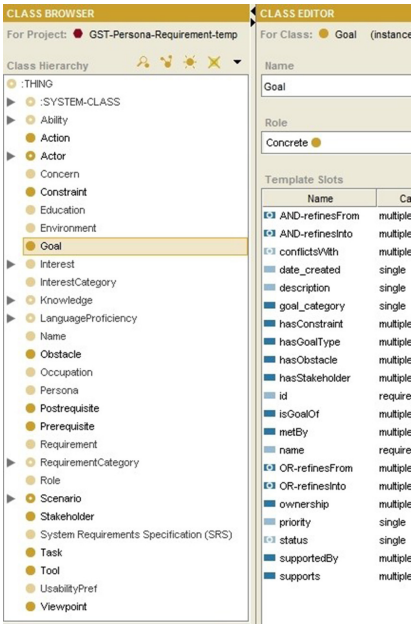


Fig. 13. Combined Behavioral-GST Ontology, Persona Ontology, and Requirements Ontology – Protégé-Frames

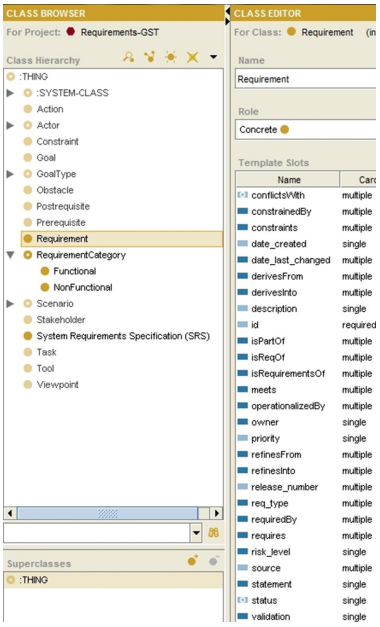


Fig. 14. Combined Requirements Ontology and Behavioral-GST Ontology – Protégé-Frames

3. Ontology Constraints Checking

In order to check for integrity constraints in the developed ontologies of the OntoPersonaURM model, we employed the capability of Protégé-Frames' PAL Constraint tab[24] plugin to help us enforce semantic properties of our ontologies specified in Protégé-Frames. In this paper, for the case study, we show selected samples of PAL constraint statements created for the Persona Ontology and the Behavioral-GST Ontology.

Persona Ontology - PAL Constraint 1: *Each name must have at most one persona.*

```
(defrange ?name :FRAME Name)
(forall ?name (=> (and (own-slot-not-null first_name ?name)
                       (own-slot-not-null last_name ?name)
                       (own-slot-not-null middle_name ?name))
                  (own-slot-not-null isPersonaNameOf ?name)))
```

Explanation: This constraint concerns all instances of the Name class. For all names, there must have at most one persona for each associated name. Note that this constraint is also equivalent to setting the cardinality of the isPersonaNameOf relationship property of the Name class to at least one. For illustration example, Fig. 15a shows that the isPersonaNameOf relationship property of a name instance (NAM1) is empty, and Fig. 15b shows that a red circle warning is displayed next to the constraint title in the "Choose Constraint" pane of the PAL Constraints tab and the violating instance (NAM1) is displayed in the "Query Responses" pane of the PAL Constraints tab.

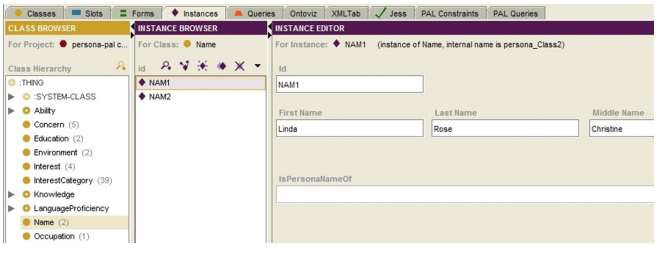


Fig. 15a. Constraint Violation – Name instance (NAM1)

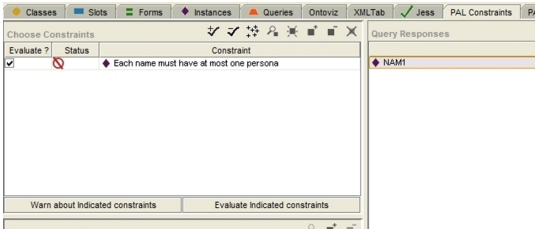


Fig. 15b. Constraint Violation – PAL Constraints Tab

Persona Ontology - PAL Constraint 2: *Each name can only have one persona (i.e. no two personas have the same name).*

```
(defrange ?name1 :FRAME Name)
(defrange ?name2 :FRAME Name)
(forall ?name1
    (forall ?name2
        (=> (and (own-slot-not-null isPersonaNameOf ?name1)
                 (own-slot-not-null isPersonaNameOf ?name2))
            (=> (and (= (first_name ?name1)(first_name ?name2))
                     (= (last_name ?name1)(last_name ?name2))
                     (= (middle_name ?name1)(middle_name ?name2)))
            (= (isPersonaNameOf ?name1)(isPersonaNameOf ?name2))))))
```

Explanation: This constraint concerns all instances of the Name class. For all names, if two name instances have the same names (first name, last name, middle name), then they must have the same persona.

Persona Ontology - PAL Constraint 3: *An interest can only belong to one interest category.*

```
(defrange ?interest1 :FRAME Interest)
(defrange ?interest2 :FRAME Interest)
(forall ?interest1
   (forall ?interest2
      (=> (and (own-slot-not-null isInterestPartOf ?interest1)
               (own-slot-not-null isInterestPartOf ?interest2))
          (=> (= (name ?interest1)(name ?interest2))
              (= (isInterestPartOf ?interest1)(isInterestPartOf ?interest2))))))
```

Explanation: This constraint concerns all instances of the Interest class. For all interests, if two interests instances have the same interests names, then they must have the same isInterestPartOf relationship property (with the InterestCategory class).

Behavioral-GST Ontology - PAL Constraint 4: *No two viewpoints can have the same persona name, environment, and role (i.e. a viewpoint is uniquely identified by persona, environment, and role).*

```
(defrange ?vpname1 :FRAME Viewpoint)
(defrange ?vpname2 :FRAME Viewpoint)
(forall ?vpname1
    (forall ?vpname2
        (=> (and (own-slot-not-null hasPersona ?vpname1)
                 (own-slot-not-null hasEnvironment ?vpname1)
                 (own-slot-not-null hasRole ?vpname1)
                 (own-slot-not-null hasPersona ?vpname2)
                 (own-slot-not-null hasEnvironment ?vpname2)
                 (own-slot-not-null hasRole ?vpname2))
            (=> (/= (id ?vpname1)(id ?vpname2))
                (or (/= (hasPersona ?vpname1)(hasPersona ?vpname2))
                    (/= (hasEnvironment ?vpname1)(hasEnvironment ?vpname2))
                    (/= (hasRole ?vpname1)(hasRole ?vpname2)))))))
```

Explanation: this constraint concerns all instances of the Viewpoint class. For all viewpoints, if two viewpoint ids are not the same, then at least one of the relationship properties of these two viewpoints, i.e. hasPersona, hasEnvironment, hasRole, must be different.

4. Conclusion and Future Works

This paper contributes towards demonstrating the application of persona concept and ontologies developed in the OntoPersonaURM model to support and enhance the requirements engineering activities, via a course registration web application case study. A five-step iterative ontology development process has been developed which aimed to help guide requirement engineers, ontology engineers, and developers in the development of ontologies of the OntoPersonaURM model on the case study. The case study demonstrated (1) how the concept of persona, in the context of the concepts of viewpoint, goal, scenario, task, requirement, environment can be integrated in a unified

environment to help engineers and developers gain a better understanding of target users' needs and behaviors and identify missing requirements early (section 2.1) in the requirements engineering process, and (2) how the relationships of the concepts can be explicitly represented ontologically to provide a shared common understanding of target users' needs and behaviors. Our future work includes one or more of the following: (1) To continue developing and improving the OntoPersonaURM model with respect to the course registration system case study as well as applying to other application domains. (2) To further conducting constraints checking on the ontologies via the PAL[24] plugin toolset. 3) To check for requirements correctness, completeness, and consistency by utilizing inference mechanism capability of ontology via JESS plugin.

References

[1] Defense Acquisitions: Significant Challenges Ahead in Developing and Demonstrating Future Combat System's Network and Software. (2008, March 7). http://www.thefreelibrary.com/Defense Acquisitions: Significant Challenges Ahead in Developing and...-a0178781051
[2] Verner, J, Cox, K, Bleistein, S, Cerpa, N. (2005). Requirements engineering and software project success: An industrial survey in Australia and the US. Australian Journal of Information Systems, 13(1), 225-238.
[3] Cooper, A. (1999). The Inmates are Running the Asylum. SAMS.
[4] Gruber, T. (1993). A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition, 5(2), 199-220.
[5] Alspaugh, T., Antón, A. (2008). Scenario Support for Effective Requirements. Information and Software Technology, 50(3), 198-220.
[6] Liu, L., Yu, E. (2004). Designing Information Systems in Social Context: A Goal and Scenario Modeling Approach. Information Systems, 29(2), 187-203.
[7] Sutcliffe, A.G., Maiden, N.A.M., Minocha, S., Manuel, D. (1998). Supporting Scenario-Based Requirements Engineering. Software Engineering, IEEE Transactions on, 24(12), 1072-1088.
[8] Antón, A. (1996). Goal-Based Requirements Analysis. International Conference on Requirements Engineering (ICRE'96), Colorado Springs, Colorado, USA, April 1996, pp. 136-144.
[9] Dardenne, A., Lamsweerde, A. Fickas, S. (1993). Goal-Directed Requirements Acquisition. Science of Computer Programming, 20, 3-50.
[10] Rolland, C., Souveyet, C., Achour, B. (1998). Guiding Goal Modeling Using Scenarios. IEEE Transactions on Software Engineering, Special Issue on Scenario Management, 24(12), 1055-1071.
[11] Shibaoka, M., Kaiya, H., Saeki, M. (2007). GOORE: Goal-Oriented and Ontology Driven Requirements Elicitation Method. ER Workshops, 225-237.
[12] Kaiya, H., Horai, H., Saeki, M. (2002). AGORA: Attributed Goal-Oriented Requirements Analysis Method. Paper presented at the meeting of the International Conference on Requirements Engineering, Essen, Germany.
[13] Pruitt, J., Grudin, J. (2003). Personas: Practice and Theory. Conference on Designing for User Experiences, San Francisco, CA. 1-15.
[14] Aoyama, M. (2007). Persona-Scenario-Goal Methodology for User-Centered Requirements Engineering. Proc. IEEE RE'07, pp. 185-194.
[15] Castro, J., Acua, S.T., Juristo, N. (2008). Integrating the Personas Technique into the Requirements Analysis Activity. Computer Science, Mexican International Conference on, pp. 104-112.
[16] Katifori, A., Vassilakis, C., Daradimos, I., Lepouras, G., Ioannidis, Y., Dix, A., Poggi, P., Catarci, T. (2008). Personal Ontology Creation and Visualization for a Personal Interaction Management System. Workshop on The Disappearing Desktop: Personal Information Management 2008. CHI2008, Florence, 5th & 6th April 2008.
[17] Golemati, M., Katifori, A., Vassilakis, C., Lepouras, G., Halatsis, C. (2007). Creating an Ontology for the User Profile: Method and Applications. In Proceedings of the First International Conference on Research Challenges in Information Science (RCIS), April 23-26, 2007, Ouarzazate, Morocco.
[18] Sim, W. W., Brouse, S. P. (2014). Empowering Requirements Engineering Activities with Personas. In: 12th Annual Conference on Systems Engineering Research (CSER 2014), Procedia Computer Science, 28, 237–246. http://dx.doi.org/10.1016/j.procs.2014.03.030
[19] Sim, W. W., Brouse, S. P. (2014). Towards an Ontology-Based Persona-Driven Requirements and Knowledge Engineering. In: Conference on Complex Adaptive Systems (2014), Procedia Computer Science, 36, 314–321. http://dx.doi.org/10.1016/j.procs.2014.09.099
[20] Noy, N., McGuinness, D. (2000). Ontology Development 101: A Guide to Creating Your First Ontology. Stanford University, 2000.
[21] Protégé Ontology Library. http://protegewiki.stanford.edu/wiki/Protege_Ontology_Library
[22] DAML Ontology Library. http://www.daml.org/ontologies/
[23] Protégé-Frames User's Guide. http://protege.stanford.edu/overview/protege-frames.html
[24] Protégé Axiom Language (PAL). http://protege.stanford.edu/plugins/paltabs/pal-quickguide/