# BEYOND THE PIE:

# COMMUNICATING WITH SMART OBJECTS USING

# MENU-BASED NATURAL LANGUAGE INTERFACES

Tanmaya Kumar <tkumar@uark.edu>, Craig Thompson <cwt@uark.edu>

Computer Science and Computer Engineering Dept., University of Arkansas, Fayetteville

## Abstract

In both real and 3D virtual worlds, people and avatars (representations of people) need to be able to communicate with things around them. But, without guidance, people do not know the language that things can understand. The goal of our research is to extend the 3D virtual world Second Life® to better model pervasive computing and overcome the boundaries of communication. This paper shows how to build a dynamic menu-based user interface that enables humans to communicate with model entities. We explored the applicability of object-specific grammars associated with things (objects in the real and virtual world) and a GUI consisting of cascaded menus to provide the guidance that will enable people to “talk to” things. This paper discusses the prototype model of a new virtual controller we developed, that takes us closer to our ultimate goal – a system that extends the Second Life user interface so people can task robots using a menu interface.



**Fig. 1: The bots on University of Arkansas Island in Second Life, a 3D Virtual World**

## 1. Context

This report is associated with the “Everything is Alive” project at the University of Arkansas that is exploring pervasive computing both in real-world RFID applications [1] and using virtual worlds, esp. Second Life [2] and Open Simulator.

## 2. Problem

People use natural languages to talk to other people. Researchers have been trying to develop natural language interfaces (NLIs) to talk to e.g., databases for the past 40 years but with limited success. It is currently difficult-to-impossible for people to communicate and converse using NLI with (most) non-human things around them (chairs thermostats, pets, blood pressure machines, forklifts …). A recognized reason is the habitability problem [1]: humans overshoot and also undershoot a system’s ability to understand their language. Overshooting means people use language that the system fails to comprehend, so the system is unable to respond to the command appropriately. Undershooting means people make the system execute very trivial tasks and fail to realize the capabilities of the system, thus refrain from using many powerful features of the system.

Another major issue with objects around us is that they do not explicitly know their own identity or type; the concept of ontologies is absent – *“I am a unique chair”*, nor do they have a way to associate additional information with themselves – *“I am owned by Tanmaya” … “I am a light switch that has been turned on 313 times this year”*.

It’s not just real world objects we want a way to talk to. In virtual worlds, in-world objects may have associated information and scripts but the ability to extract the information or how an object can be manipulated may rest solely in the head of the object developer. No avatar passing by can learn the command language of the object and interact with it. To aid the user Second Life does offer the PIE user interface that can be accessed by a user by selecting an object and which allows the user to access generic commands such as sit, take, copy, buy etc. However none of these commands are object specific and do not allow the user to manipulate the special capabilities the object may have, for example thermostats do not have their own object type specific commands.

## 3. Objective

Our research was aimed at enabling human-object interaction by providing a GUI interface that can be parameterized by the physical properties of everyday objects which would enable the interaction, in both real and virtual worlds.

In our initial implementation, we limited interactions to one type of entity - in our Second Life 3D virtual world, we had robots with command language modules available to us, so we decided to test our form-based natural language interface by commanding and querying the robots.

As observed above, Second Life currently employs a pie shaped interface that allows the user to select from a “fixed and limited” array of generic verbs that allow the avatar to perform very basic actions such as sitting, touching objects, etc. Even though, the pie can cascade to give a small number of additional commands; the pie commands are not object-specific. We needed a way to extend this user interface to be able to communicate with things in a thing specific language. For our current work, we targeted the Second Life robots on our University Island.

## 4. Approach

We used a series of Form Based Graphical User Interfaces (GUI’s) which provide a common way humans communicate with computer-based objects. A complementary alternative is Menu Based Natural Language Interfaces (MBNLI’S), which provide sequential command completion menus [3], similar to drop down menus. Both provide a way to solve the habitability problem since both provide a way to display all and only the legal commands a system can handle. Instead of “creating” an unsupported query as done in conventional NLI, using GUI or MBNLI, human users can “recognize” the command they meant to formulate while creating an appropriate string of commands using a command builder. This method also enables humans to see commands that they might not have known about – that is, humans are guided to rendezvous with the capabilities of the system, thus eliminating the chance of a user undershooting or overshooting a system’s capabilities.

## 5. Progress

We developed a prototype next generation PIE interface for Second Life that uses a combination MBNLI-GUI to enable humans to communicate with specific things.

In our initial implementation, we limited interactions to one type of entity – robots [8]. A student at University of Arkansas, Nick Farrer, had previously developed a Robot Assembly Language that provided chat-based commands in Second Life for controlling a fleet of robots. Robots can go from location to location following way points. They can pick up, carry, and put down objects.

In order to get a PIE that operated in both Second Life and OpenSim, we developed our PIE code outside both environments so that it overlays as an external application on top of those browser-clients.

We developed object-specific grammars like the one shown below for robots. If the user clicks on a robot, the grammar commands for the robot are interpreted and displayed in a cascade on the menu. At the end of a PIE command sequence like *“Robot – Pickup – the ball”*, the command is translated into a command in the Robot Command Language and then transmitted over to the robot in second life via an HTTP callback and then executed.

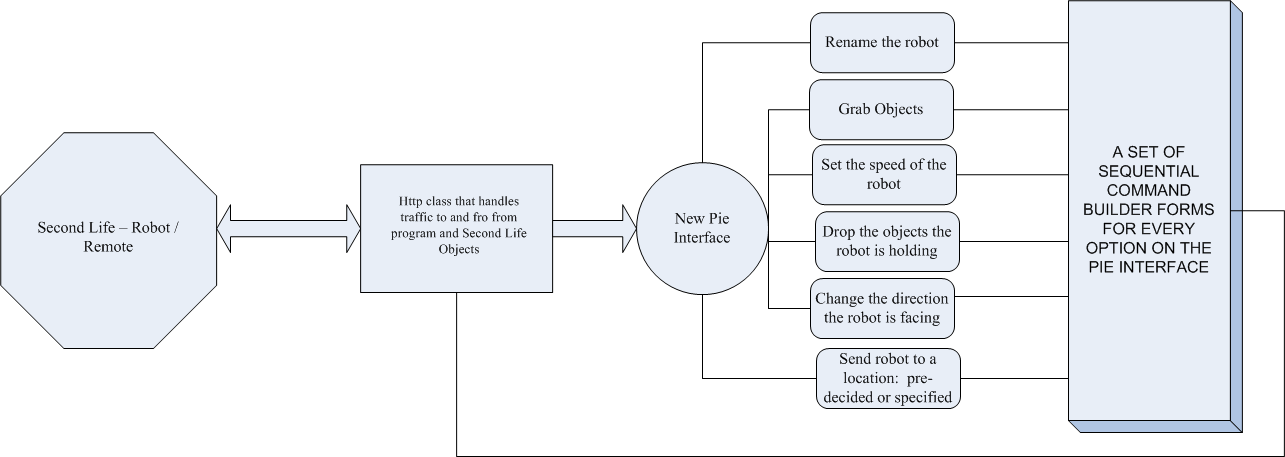


Fig. 2: The program design and interaction paradigm

The change form one pie to another is seamless, as the new pie has been integrated into a small script that corresponds to the man program running on a standalone system via an HTTP callback and is activated by a single left mouse click on the object the avatar is interested in. The new Dynamic PIE 2.0 is shown below against the pie menu provided by Second Life®.



**Fig. 3: The new pie interface (left) [2] vs. the one provided by Second Life (right)**

### 5.1 Implementation and Hurdles

The primary aim of the project was to develop an interface to facilitate the interaction. The interface was built using Visual C# and it was built entirely outside Second Life®. It interfaces with objects inside Second Life® using an HTTP callback class which ferries commands and requests to and from the target object – in our case the Robots and the soft controller/transmitter on the Avatar’s hand. The stand alone nature of the PIE 2.0 is not accidental and it was developed as a plug-in to Second Life®, OpenSim and all supported Virtual worlds to support extreme portability and to avoid the legal software licensing issues that Second Life and Open Simulator suffer from. [9]

### 5.2 Sample Command Building Process

The new pie interface works by using a pre-programmed set of forms as interfaces for the command development. Normally the bots function by listening on the common chat channel: “channel 0” for commands being issued to them by the user (Avatar). The commands are all type set and are hard to memorize, formulate and reproduce at the desired instant. Following is an example of what a user has to type into the chat-bar to command the robot to go to the pharmacy and get a pill bottle:

1. SampleBot rename to MyBot --Renames robot to MyBot
2. MyBot wp shelf2 --sends robot to waypoint named shelf2
3. MyBot grab nearest Clock --Grab clock form the shelf
4. MyBot wp loading --sends robot to loading dock
5. MyBot drop --Robot drops clock at the loading dock

The following figures show how some of these commands can be built in the new interface without having to have these command steps memorized.

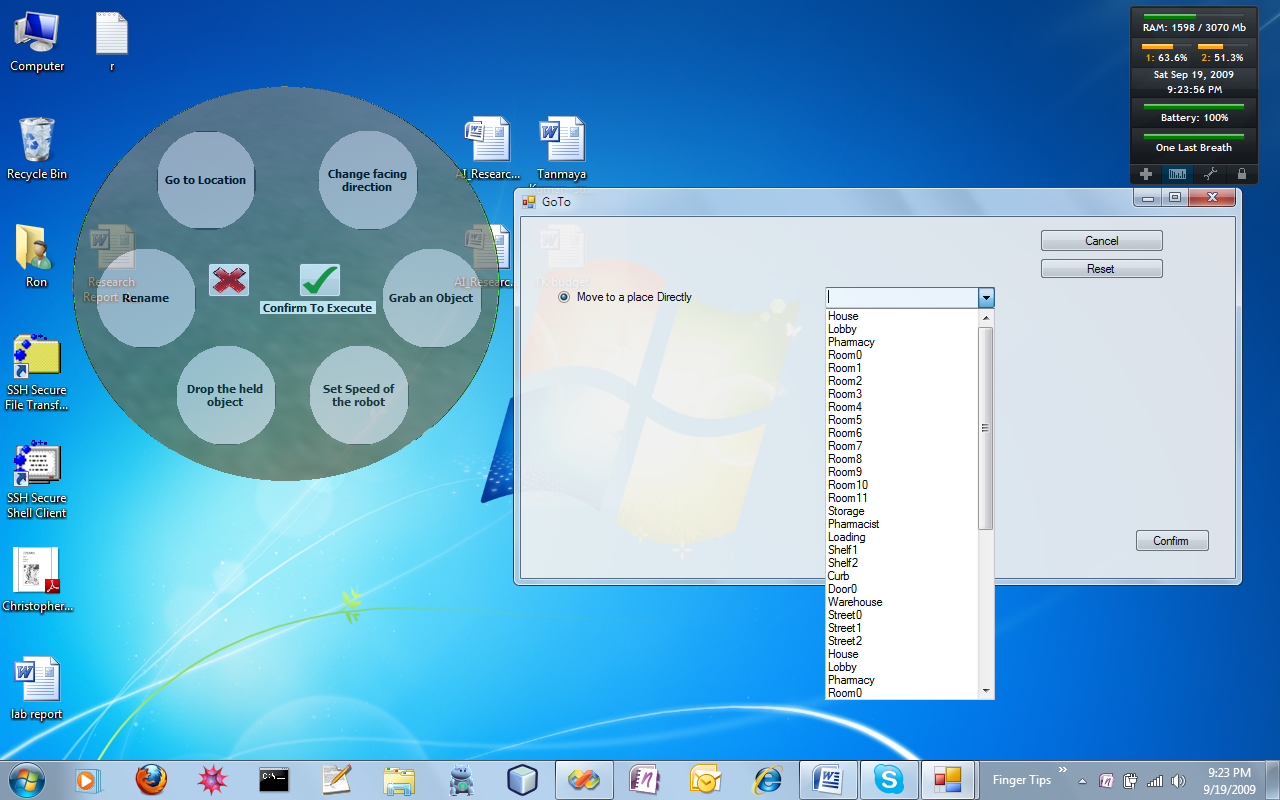
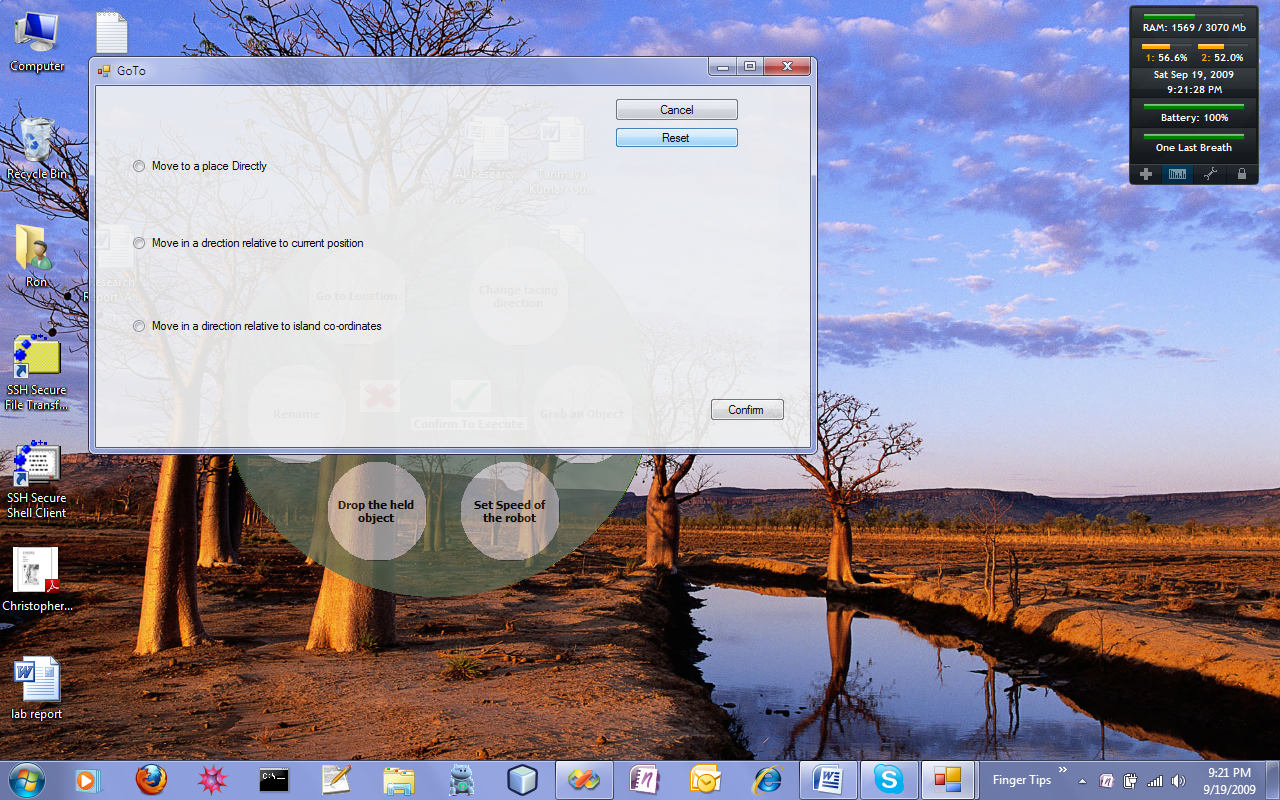
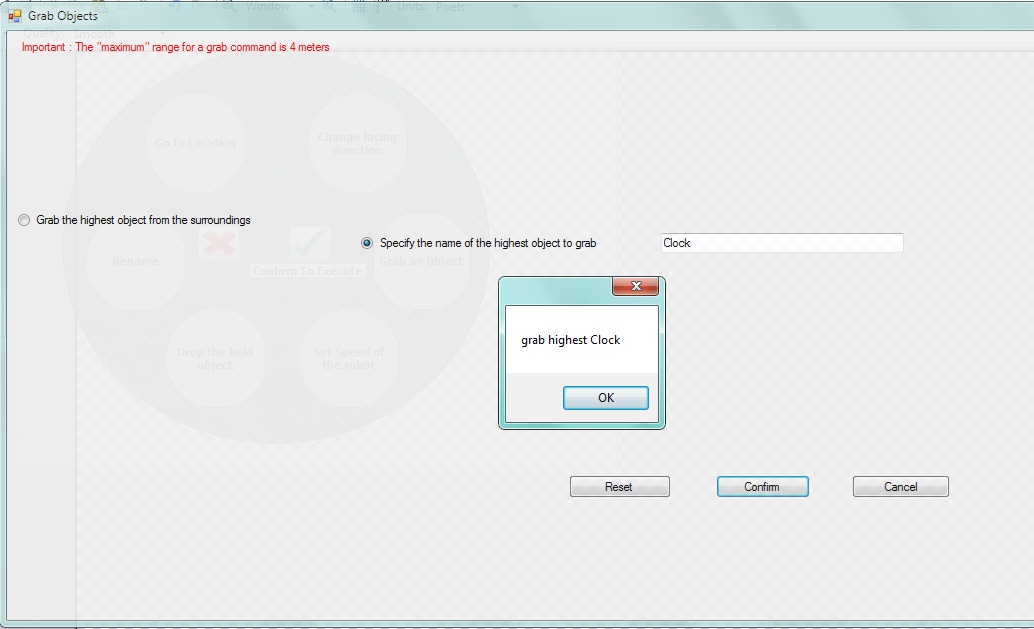


Fig. 4: An instance of the Go to Location Command builder and list of waypoints



**Fig. 5: A snapshot of the Grab a clock command**

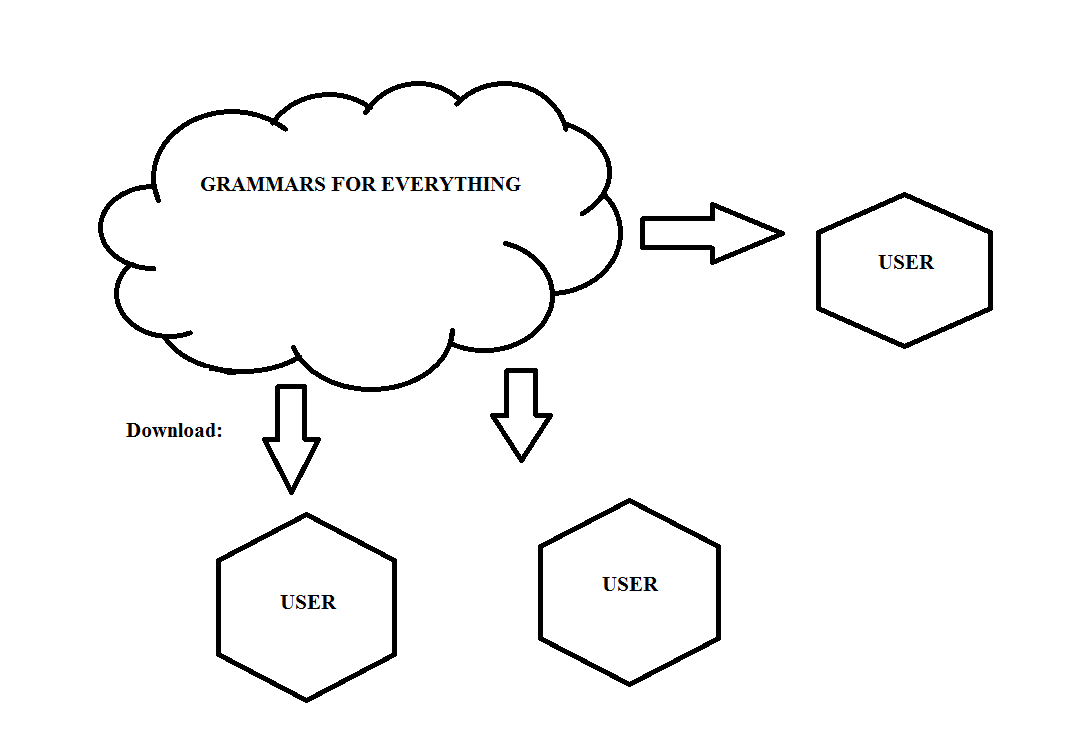
## 6. Future Work

The ability for humans to communicate with things is a significant step towards a smart world (smart homes, smart buildings, smart hospitals…) where many or all objects are network objects with an identity, the ability to communicate (wired or wirelessly) with other objects or with humans.

The research we conducted allowed us to understand the working of the Second Life PIE in detail; it was a prototype hardcoded to control the Robots on “University of Arkansas” island in Second Life. Even then, the interface did not cover all of that command language. Most other Second Life objects still lack the ability to understand their own type and super and sub classes; which might be another place to begin. The grammars are not yet dynamically loaded into the PIE. So there is considerable work ahead – but we have isolated a next set of problems to solve.

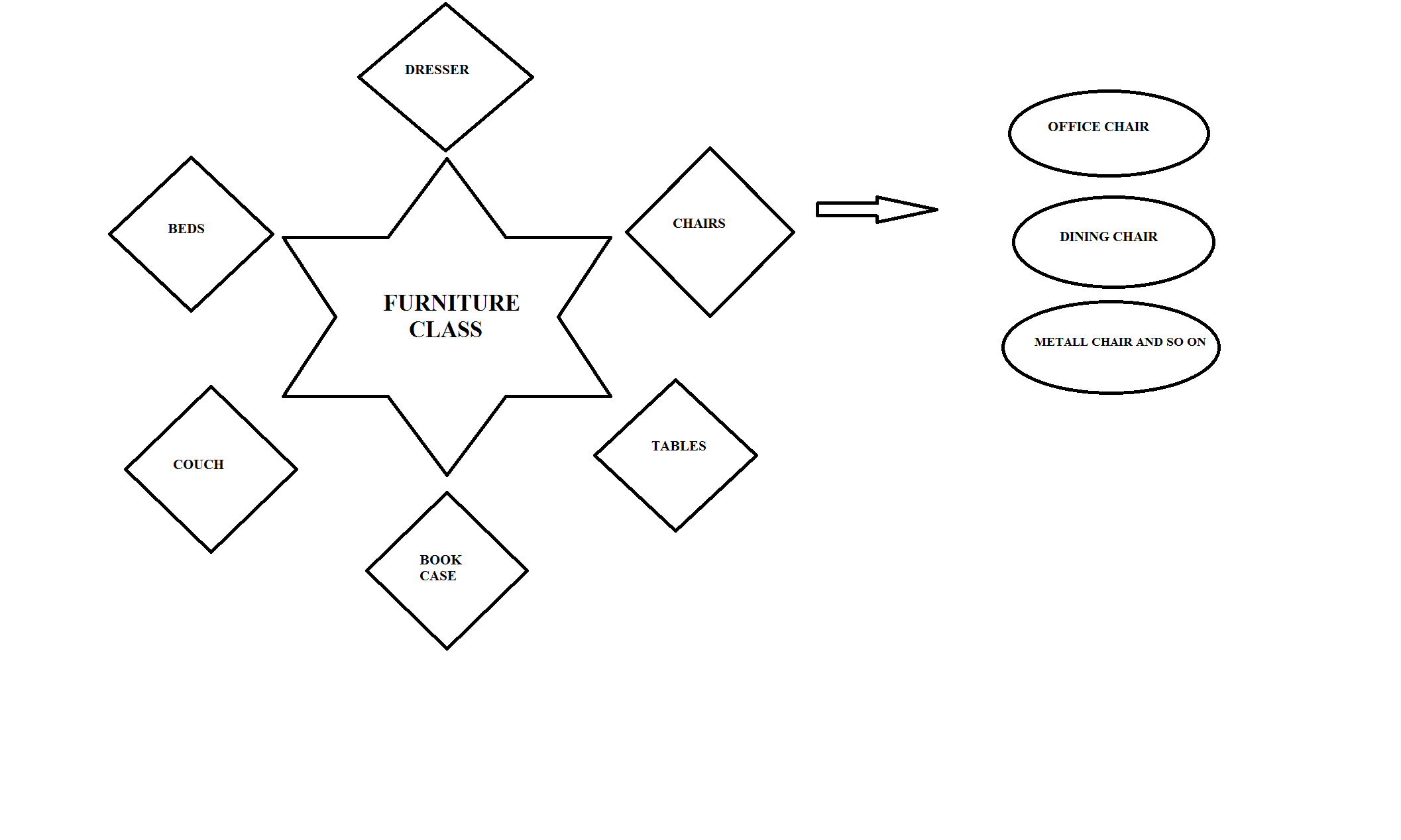
Some requirements for future work will include the development of objects that can identify themselves as belonging to a parent class. Such as a chair knowing it is a type of furniture. Also, an API that recognizes the “type” of the object and recalls the particular set of grammar rules that apply to the object; possibly even differentiating the availability of grammar rules on basis of the user. Last but not the least; the inclusion of the concept of object specific grammars and the need for developing grammars for every kind of real-world thing. More specifically:

1. Consider other Second Life objects such as objects of a smart home. If we could develop grammars for everything and put them in a remote cloud to be accessible to the users and which, the users can download on the fly on their smart phones, the boundary of communication between a person and his surroundings would be breached. The person can then in simple sense talk to anything around him wherever they are.



**Fig. 6: The grammar cloud concept**

1. Build a more advanced interface that can handle more complex commands using grammar-based control structure, complex translations and Ontology structures.



**Fig. 7: Ontology Architecture**

1. Develop role based interfaces such as interfaces specific to a particular task: wood cutting machine for a carpenter, auto tools for a mechanic, etc.

Develop Access Control mechanisms to control and limit access by users to differentiate by authorized and unauthorized access, or to attach an inventory of objects to a particular avatar so only that avatar can access them : similar to the act of owning and commanding objects of my own home.

Finally, if smart grammars work in Second Life, they can be made to work in the real world. If every real-world object had and RFID tag that identified it, and if every smart phone could read RFID tags (which will happen when RFID becomes a consumer level technology), then consumers can shop or walk around and use their smart phone to communicate with things around them (canned goods, the thermostat, their cat) – all using technology similar to the technology we are designing.

Some Companion papers discuss strides made in related areas:

* protocols for extending an ordinary real or virtual world object into a smart object [5]
* a universal soft controller architecture that humans can use to communicate with things [6]
* an ontology service that associates knowledge and interfaces with things [7]

## 7. Potential Impact

If we can determine the kinds of interfaces an object can possess and then develop a synthetic grammar for the commands and replies for the object and then extend the communication interface to support such interaction, it will make it possible for humans to interact with objects. We believe that a similar approach can be used for people using soft controllers (smart phones) to communicate with everyday objects in the real world. Consider if every real world object has an RFID tag that indicates the object’s individual ID. A smart phone with and RFID reader could communicate this information to a remote ontology on the web to download an interface that lets a consumer talk to the thing. If it becomes a standard (optional) protocol to define such interfaces for all things, then anyone anywhere can communicate with any tagged thing!

**References**

[1] Everything is Alive project website (aka Modeling Healthcare in a Virtual World), http://vw.ddns.uark.edu

[2] Second Life website, http://www.secondlife.com

[3] C. Thompson, P. Pazandak, H. Tennant, “Talk to your Semantic Web,” *IEEE Internet Computing*, November-December 2005. http://csce.uark.edu/~cwt/DOCS/2005-11--PAPER--IEEE-Internet-Computing--Talk-to-your-Semantic-Web.pdf

[4] Tanmaya Kumar, "Enabling Communication between Humans and Inanimate Objects in a Virtual World," Final Report, Undergraduate Research Project, Honors College, University of Arkansas, Fall 2009. http://vw.ddns.uark.edu/content/2009-10--Honors-College-Research-Grant--Final-Report--Tanmaya-Kumar.doc

[5] A. Eguchi, C. Thompson, "Smart Objects in a Virtual World," X10 Workshop on Extensible Virtual Worlds, venue: Second Life, March 29 – April 2, 2010. http://vw.ddns.uark.edu/X10/content/X10--papers.htm

[6] J. McCrary, C. Thompson, "Soft Controller – A Universal Remote for the Internet of Things and for Virtual Worlds," X10 Workshop on Extensible Virtual Worlds, venue: Second Life, March 29 – April 2, 2010. http://vw.ddns.uark.edu/X10/content/X10--papers.htm

[7] T. Censullo, C. Thompson, "Semantic World: Ontologies in the Real and Virtual World," X10 Workshop on Extensible Virtual Worlds, venue: Second Life, March 29 – April 2, 2010. http://vw.ddns.uark.edu/X10/content/X10--papers.htm

[8] Nick Farrer, "Second Life Robot Command Language," Tech Report, CSCE Department, University of Arkansas, Fayetteville, AR, 22 February 2009. http://vw.ddns.uark.edu/content/2009-02--SL-Robot-Command-Language-v0--Nicholas-Farrer.doc

[9] “Open Sim”; Licensing Issues, website: http://www.opensimulator.org/wiki/Contributions\_Policy