GOOAL: A Graphic Object Oriented Analysis Laboratory

Hector G Perez-Gonzalez
Department of Computer Science
Univ. of Colorado at Colo. Springs
Colorado Springs, CO, USA
(719) 262-3325
hectorgerardo@acm.org

Jugal K. Kalita
Department of Computer Science
Univ. of Colorado at Colo. Springs
Colorado Springs, CO, USA
(719) 262-3432
kalita@pikespeak.uccs.edu

ABSTRACT
Our goal is to enable rapid production of static and dynamic object models from natural language description of problems. Rapid modeling is achieved through automation of analysis tasks. This automation captures the cognitive schemes analysts use to build their models of the world through the use of a precise methodology. The methodology is based on the use of proposed technique called role posets, and a semi-natural language (called 4W). Original problem statements are automatically translated to 4W language. The produced sentences then, are analyzed with role posets to produce static model views. Finally the 4W sentences are used to generate dynamic views of the problem. This set of methods maximizes analysis process agility, promotes reusability and constitutes a valuable tool in the learning process of object thinking. The prototype tool: “GOOAL” (Graphic Object Oriented Analysis Laboratory) receives a natural language (NL) description of a problem and produces the object models taking decisions sentence by sentence. The user realizes the consequences of the analysis of every sentence in real time. Unique features of this tool are the underlying methodology and the production of dynamic object models.

Keywords
Object oriented modeling, computational linguistics, discourse analysis, set theory, object oriented education.

1. PROBLEM DESCRIPTION
The successful development of any software system depends on the communication between customers and software developers. Customers communicate in common, widely comprehensible, but at the same time vague and potentially contradictory natural language (NL). Computer specialists communicate using precise, but at the same time, not widely understandable formal languages. Past similar efforts [2][4] produce static object views and the associated automatic tools are based in methodologies considering noun frequencies. Our techniques look for simplicity and effectiveness considering also semantic issues and discourse analysis and simplification.

2. METHODOLOGY
The purpose of this methodology is to promote rapid software development, reusability, and support memorable experiences in “object oriented thinking” and it is mainly supported by the use of automated tools. The general steps are as follows:

Step 1. Analysts obtain a set of describing requirements coming from stakeholders of different kinds. (several documents from several final users, several from different clients, etc.)

Step 2. The name of the problem, problem domain’s (PD) name and possible sub-domains are declared.

Here is an example of one such simple requirements document:

Problem Domain: “Operating systems”
Sub-domain: “Concurrent programming”

Problem name: “Dinning philosophers simulation”

Problem Description: There are 5 philosophers and 5 forks around a circular table. Each philosopher can take 2 forks on either side of him. Each fork may be either on the table or used by one philosopher. A philosopher must take 2 forks to eat.

Step 3. Each one of the documents is processed using an automatic tool considering associations between documents with the same kind of authors. The tool produces design view diagrams that are validated by the user.

Step 4. Any participant, stakeholder or mainly any student of object oriented (OO) design may modify, delete or add new sentences to any requirement text to identify the consequences in real time and produce a memorable learning experience.

Step 5. Produced information is stored associated with the declared problem domain in order to promote reusability.

The proposed techniques produce OO static and dynamic model views of the problem in Unified Modeling language (UML) [1].

2. THE 4W LANGUAGE
In order to have enough simple sentences to be analyzed with the role posets technique, we propose a subset of English called the 4W language to which original sentences are translated. In the 4W language certain expressions show a syntactic subject who performing an action, an optional syntactic object receiving the action (if the verb is transitive) and an optional prepositional phrase giving information about adverbial or adjectival relation to some other word in the sentence. In addition, we can have a link between sentences, a relationship that gives us information about time sequences or conditional rules between them. In summary, a 4W sentence tries to answer the following four questions related to a particular object: What does the object do?, Who receives the action?, Which other participates? and When does it happen? 4W sentences are displayed in a 4W table with each slot filled with a word.
Automatic 4W translation of our requirements example is:

1. five philosophers are around a circular table
2. five forks are around a circular table
3. Each philosopher has side.
4. Each philosopher can take two forks on side.
5. Each fork may be on the table.
6. Each fork may be used by one philosopher.
7. A philosopher take two forks.
8. A philosopher eats (When 7).

This semi-formal language (4WL) is expressive enough to translate most declarative sentences, clear enough to be understood and validated for problem domain users and unambiguous enough to produce didactic design diagrams.

4. ROLE POSETS

Partially ordered sets of roles (Role Posets) are a conceptual framework to take analysis and design decisions to produce OO static model views (Class, Objects and use cases diagrams). It is based on the linguistic concept of theta roles [3] and the mathematical concept of partially ordered sets [5].

Attributes, behavior, and relationships are the components that an entity (represented by a noun) must have in order to be promoted to a class. These three components can be discovered through the analysis of the role every noun plays with the complete requirements texts, and the verbs that define these roles. This is done with a proposed semantic structure called “role machine”, a proposed semantic abstraction that associates groups of numerous verbs in general formal schemes.

We propose a universal partially ordered set of roles composed by: Agent (Ag), User (Ur), Modified (Mr), Used (Ud), Whole (Wh), Part (Pt), General (Gl), Special (Sp), Theme, LSP (Location, situation or position) and attribute (At). Our prototype tool internally labels every noun in the text with the particular role it plays according to its associated verb, and its relation with it. At the end of this automated analysis, there is a list of nouns (and adjectively qualified nouns), every one associated with a list of roles it plays.

After following the complete technique, we obtain the probabilities every noun (N) has to become a Class. In this particular example it would be:

<table>
<thead>
<tr>
<th>Noun</th>
<th>Noun's name</th>
<th>Prob. To be Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>N1</td>
<td>Philosopher</td>
<td>100%</td>
</tr>
<tr>
<td>N2</td>
<td>Fork</td>
<td>83%</td>
</tr>
<tr>
<td>N3</td>
<td>Table</td>
<td>20%</td>
</tr>
<tr>
<td>N4</td>
<td>Side</td>
<td>9%</td>
</tr>
</tbody>
</table>

Table 1. Result of analysis of nouns

Trustable choice of roles are based in this role poset scheme and works to model simple and complex relationships as aggregation or inheritance.

Figure 1 shows a simple class diagram obtained from our example considering a validation threshold of 50%.

5. THE GOAL OF GOOAL

The objective of GOOAL is to produce rapid software development, reusability and support memorable experiences in the learning process of object oriented design.

6. DEMONSTRATION OF GOOAL

The audience of this demonstration will see how a group of sentences describing a problem are received by GOOAL system, it takes illustrative decisions and produces static model views of the problem, then the system shows its interpretation in 4W language which is validated by the user, finally dynamic object views of the problem are shown.

Unique features of this tool are the underlying methodology and the production of dynamic object models. GOOAL has produced good results with simple problems. It is being developed to work with complex ones. Observed advantages are standard notation, formalization, validation, traceability, efficiency and early identification of misunderstood requirements.

7. REFERENCES