Object Orientation in Action: A System for Designing Speech Articulation Deficiency Treatment

Tonya Lynn, James Baughn, Anne Hays, Erica Langlitz, Holly Mirly, and Ken Surendran
Southeast Missouri State University
Cape Girardeau, MO 63701

Abstract

The authors share their experience in the application of object-oriented tools and techniques for the design and development of an unconventional complex decision support system. Following a systematic unified development process, they generated all the necessary analysis and design artifacts using UML diagrams and implemented the system in Java for maximum portability. The system is primarily intended for clinicians in testing and treating children with speech articulation deficiency. It also serves as a tribute to Dr. Robert Milisen, who proposed this clinical solution nearly forty years ago.

1. INTRODUCTION

In response to a call for projects for the Software Engineering (SE) course, a faculty member in the Communications Disorders department in the authors' university wanted a tool for designing a speech articulation treatment plan. The faculty member gave, as a client of this project, all the necessary details to the authors – a team of students who worked on this project and the SE course instructor - concerning a consistent method for diagnosing and treating speech articulation defects. In this methodology, developed through research and testing in the 1950s and 1960s by Dr. Robert Milisen [1] of Indiana University, a clinician administers a series of verbal tests, records and evaluates the results, and develops a treatment plan, which is adaptively altered throughout the treatment process. The faculty member, as a former student of Dr. Milisen, recognized the importance of this systematic approach and wanted to mechanize the whole process for convenient adoption by her own students.

A capstone system development project using the tools in object paradigm serves as the main assessment item in the SE course [2]. After establishing the preliminary feasibility, this treatment design project was assigned for development to a team of senior Computer Science majors – also the main authors of this paper - as a part of their SE course requirement in the spring of 2002. The SE course used a Unified process consisting of three stages: requirements specification, design and implementation of baseline system, and implemented system with all documentation. In this paper the authors describe the analysis and design artifacts in terms of UML diagrams, drawn using Rational Rose, and share the unique learning experience this course offered the team.

2. REQUIREMENTS ANALYSIS

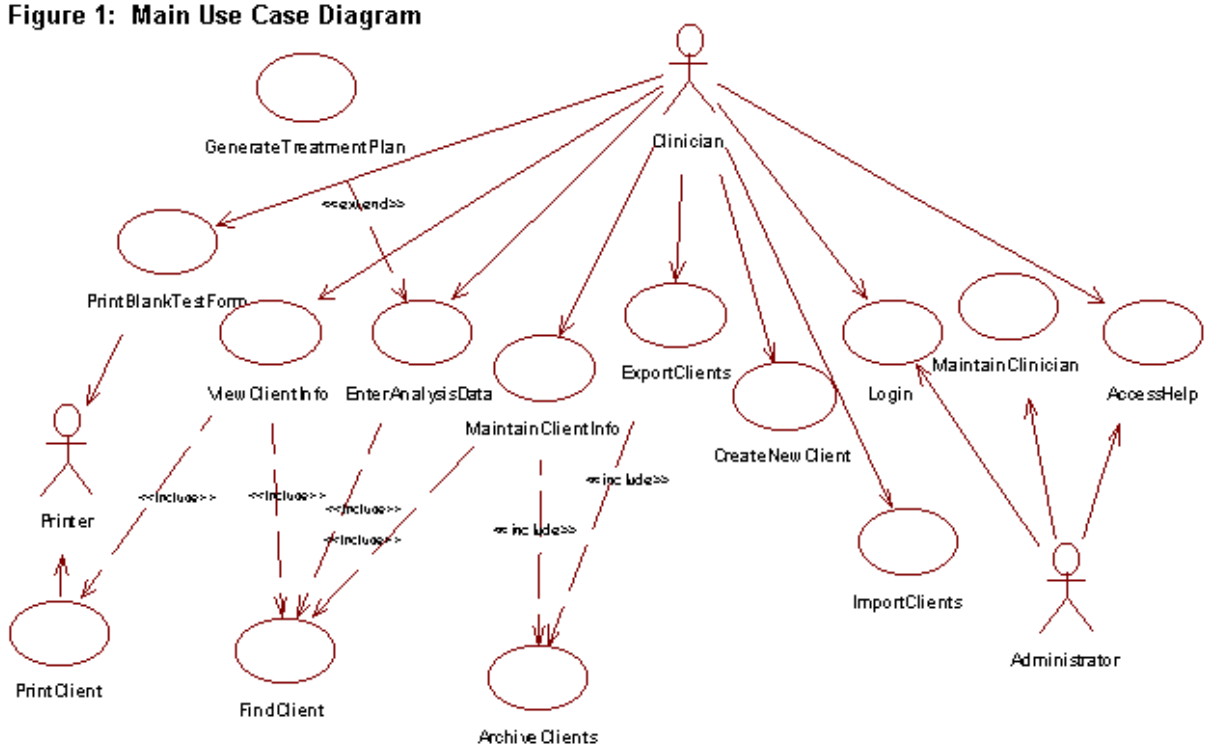
The objective of the software is summarized in the use-case diagram (Figure 1) and it is intended to include the following facilities:

- Record test results in real time
- Establish basis for treatment design through analysis of test results
- Design a customized treatment plan

The use-case descriptions, which deal with the various scenarios, pre- and post-conditions, are omitted here for want of space. The team took the opportunity to streamline Dr. Milisen's process so that the clinicians can focus on the treatment of their client, rather than the paperwork.

Non-functional requirements quickly presented themselves during the analysis and design phases of the project. The team members realized through interviews with the client that the intended users of the system, clinicians and speech pathology students, would not be guaranteed to have an extensive background in computer operation. Therefore, a user-friendly design and built-in online help system were deemed important requirements. Copies of the various forms

Figure 1: Main Use Case Diagram



and worksheets used by clinicians in Dr. Milisen's testing and treatment process, in addition to research articles written by Dr. Milisen and his associates relating to the Prescriptive Articulation Test and its development were studied by the team to establish the requirements specification.

The system allows a single administrator to manage clinician access, multiple clinicians to each maintain their own unique set of clients, and also provides archival database storage of client information for future statistical research as well as exporting of client information for transferal from one location to another. The primary use case of the system, however, is the generation of the treatment design. The clinician enters into the system the numerical scores obtained during testing and/or treatment of the client, and instructs the system to generate a treatment plan. The system then saves this analysis data, and an algorithm then computes a suggested method of treatment based upon Dr. Milisen's methodology. For the treatment to be generated, there are several preconditions that must be met. Once the clinician has successfully logged onto the system and located a pre-existing client within the database, (s)he can enter the results for the Descriptive Articulation Test, the Prognostic Test, and the Feedback Test. The data from these tests may be modified as treatment progresses, and the treatment plan is re-computed by the system; thus, an adaptive treatment plan is available to which the clinician may refer. In the event that the preconditions are not met, the system either alerts the user that the client is not found within the database (if the client is not pre-existing), or prompts the user to enter the missing test data via display of the data entry screen associated with the test variables.

The treatment plan produced by the system provides the clinician with a concise suggestion for determining the treatment. Speech articulation deficiencies are prioritized according to Dr. Milisen's methodology, ranked in the suggested order of treatment, and the initial treatment plan is established. For example, the first suggested sound to be treated may be the letter 'r' pronounced in isolation for an easy, receptive audience. Another example might be the sound 'th' pronounced at the end of a word for a moderate to difficult, unreceptive audience. These suggestions, however, while taking into account Dr. Milisen's guidelines, may be recognized as unsuitable by the clinician. For instance, a cleft palate may impair speech to the point where surgical means are necessary in order to repair the speech deficiency before clinical methods may reasonably be employed. In such instances, the system allows the clinician to modify the treatment design to suit the special needs of the client.

3. DESIGN SPECIFICATION

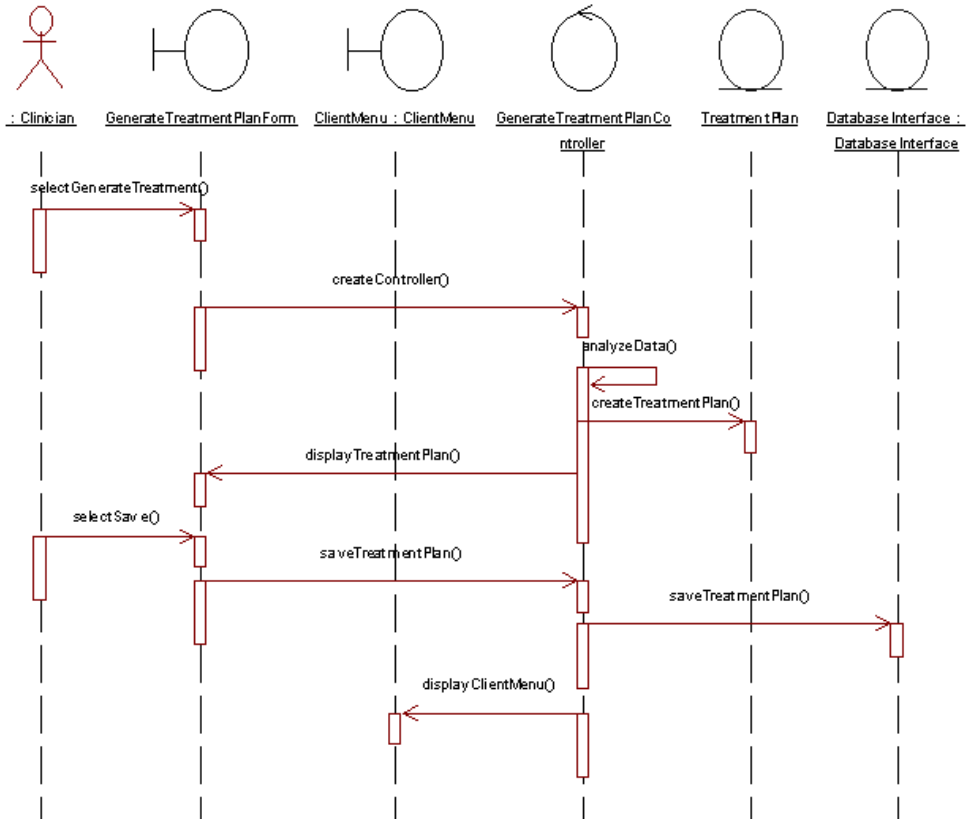
The conceptual design of the Milisen Prescriptive Articulation Test system is based upon a three-tiered architecture. The top layer, or access layer, is composed of the database and the help system. The central, or business layer, contains the interfaces to the help system and the database, as well as classes that handle the processing of the data within the system. The bottom, or view, consists of those classes that compose the user interface. Theoretically, these layers could be separated to run on different systems and/or in different locations; however, the user's requirements and the scope of the project dictated that the layers were to be combined within one system.

Analysis and design models for the system were produced for all use cases, for all main flows as well as for several alternate flows. Figure 2 depicts the sequence diagram for the Generate Treatment Plan use case described earlier in this paper.

Many classes were used in the implementation of the system, as demonstrated in Figure 3. In this class diagram, the main classes used in the Generate Treatment Plan use case and the interacting classes from other packages are shown. The use of graphical user interfaces, programmed in Java, combined with additional interfaces for the database and help system, necessitated the use of the classes implemented.

The database schema for the system is given in Figure 4. It is important to note that entities like Speech and Score are the critical elements that capture the essence of Dr. Milisen's methodology

2: Sequence Diagram, Generate Treatment Plan



4. IMPLEMENTATION

The classes used in the system were consolidated into packages, which were then distributed among the team members for implementation. Each team member was responsible for the construction and testing of one or more packages.

The program is divided into seven different subsystems, represented by the packages and dependencies shown in Figure 5. These packages and the classes contained within them were coded in Java, as the team's goal was to make a system functional across a variety of operating systems with different windowing environments. The language chosen allowed for a great deal of ease in interfacing with both ODBC and JDBC database interfaces as well as for initiating SQL statements.

Figure 3: Generate Treatment Plan Class Diagram

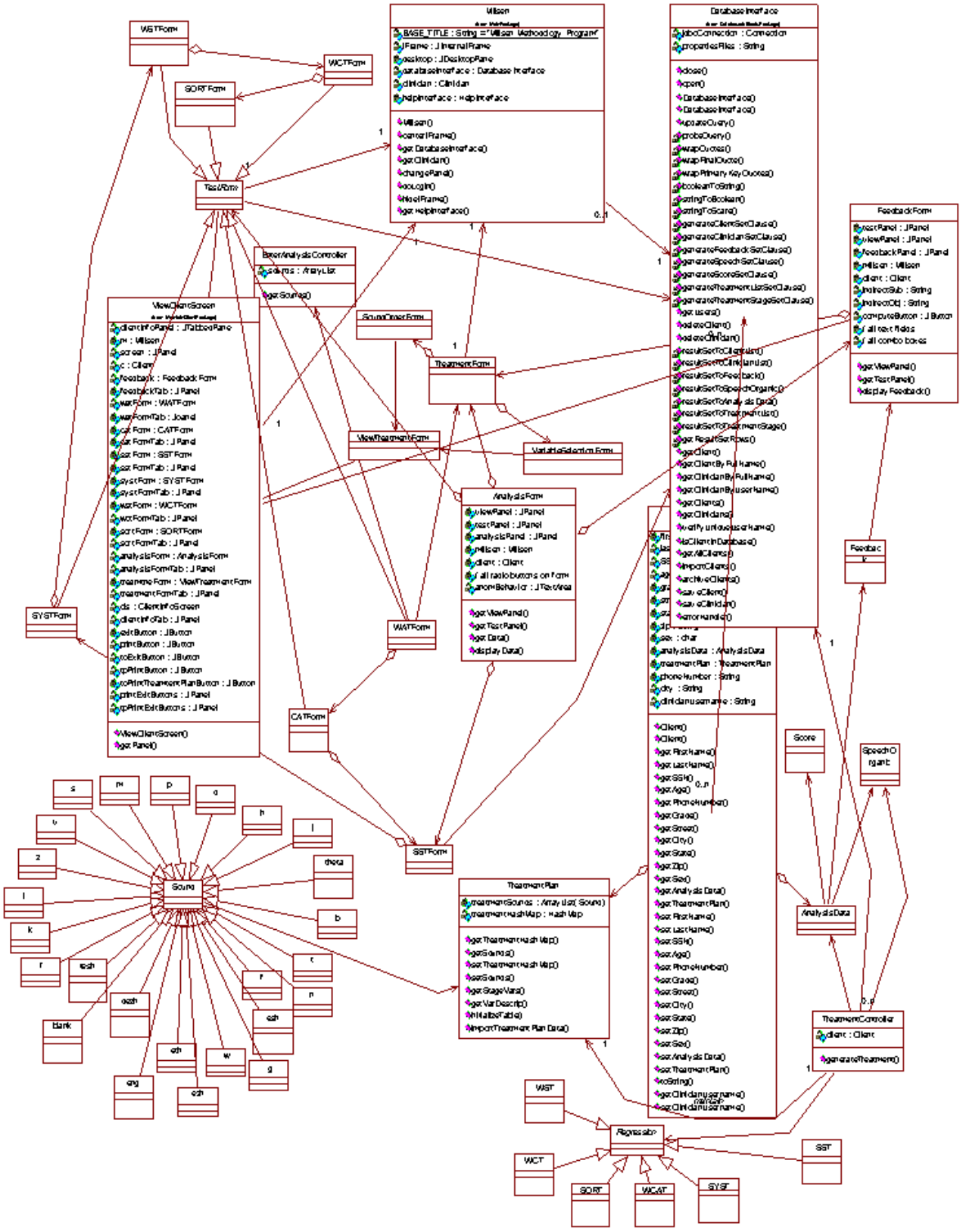
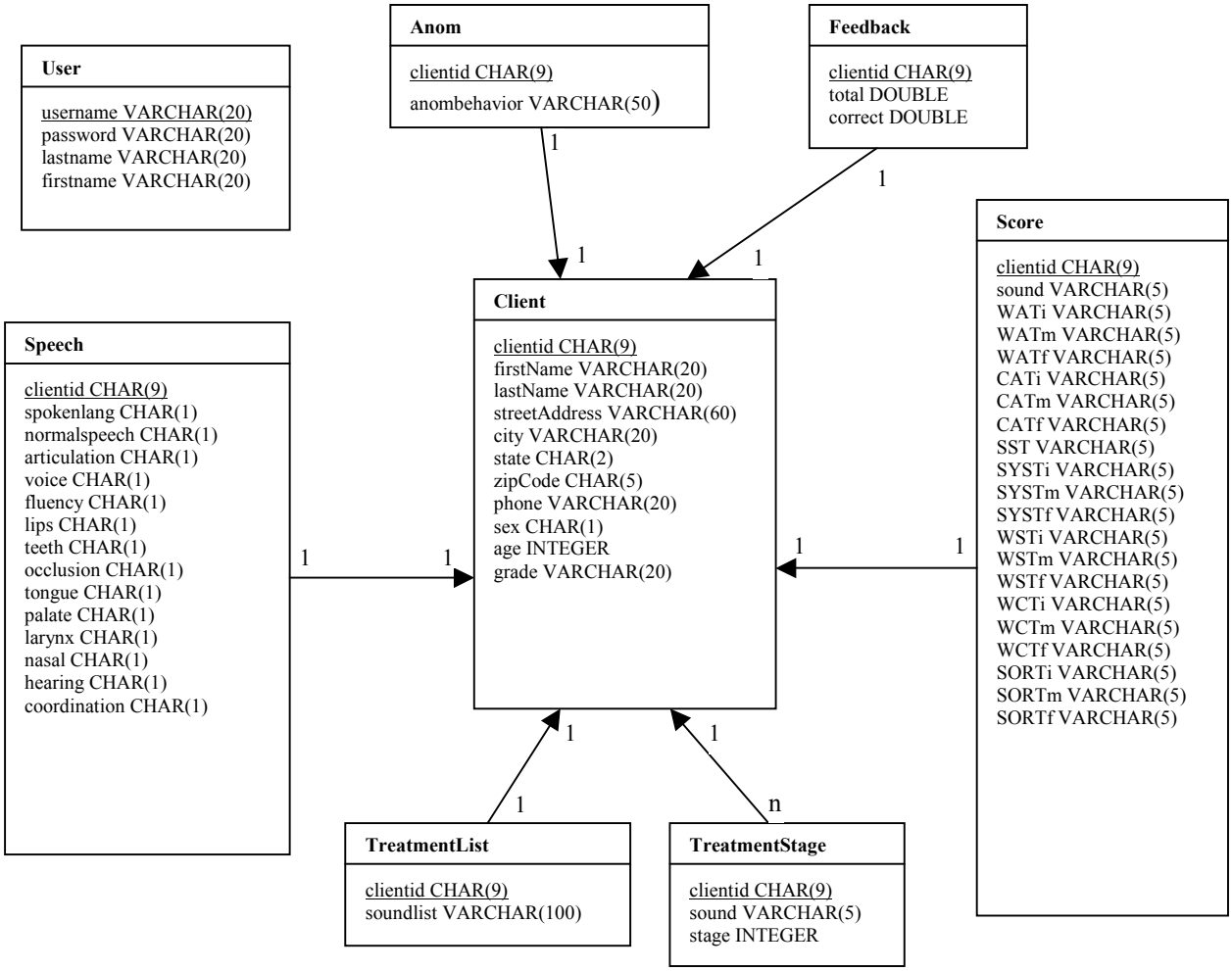


Figure 4: Database Schema for Milisen Methodology Program



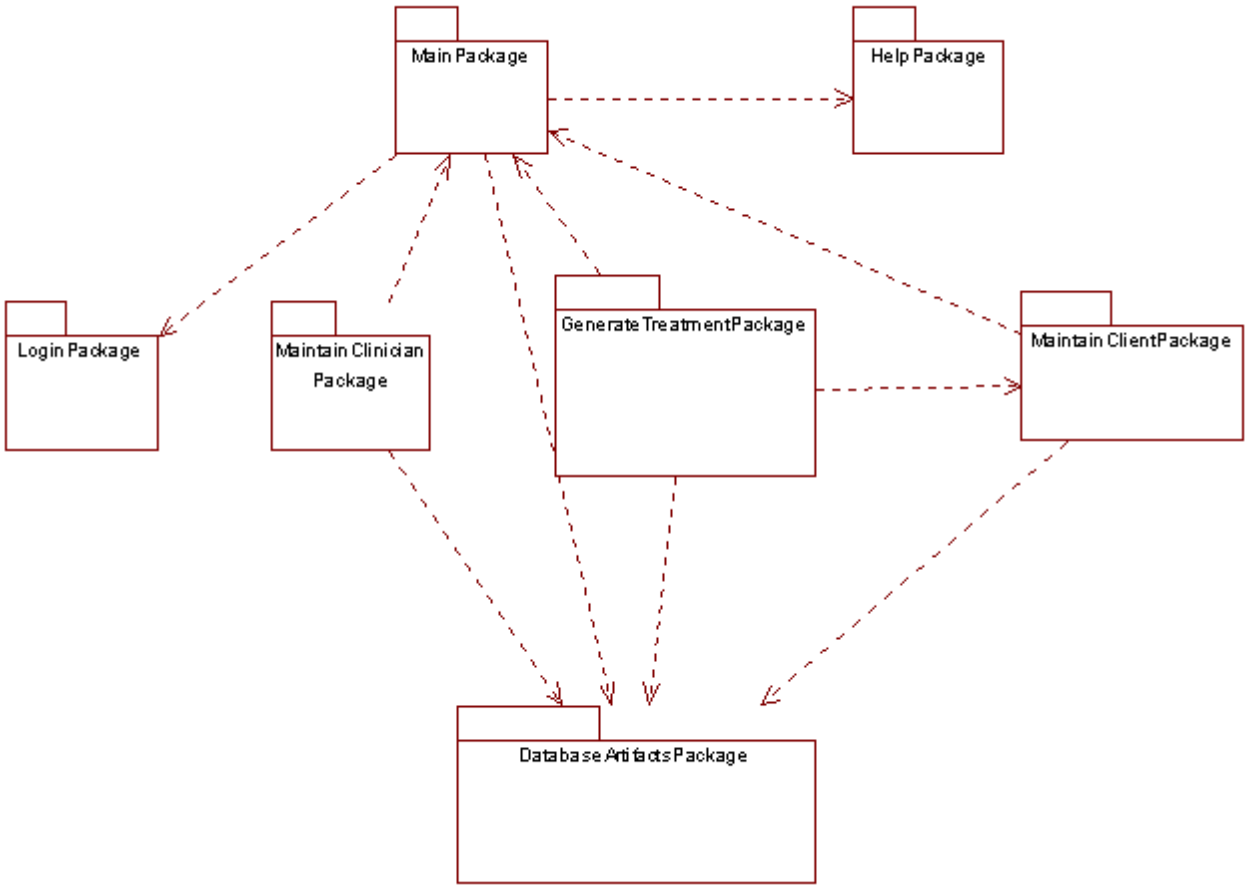
A variety of free and shareware utilities were used in the development of the system. All Java coding was performed in the shareware program TextPad, while the Java SDK and JavaHelp are free utilities provided by Sun. The help system was automated through use of a program called HelpBreeze, a free trial program available at the time of system development. InstallAnywhere Now v. 4.5 allowed easy development of an initial executable.

5. EDUCATIONAL VALUE OF PROJECT

For most of the five members of the software development team, this was their first experience in writing a piece of stand-alone software of this size. The team members all agreed that the process was educational. Each member learned specific technical skills related to their assigned package for implementation, and all achieved the experience in analysis and design using UML and Rational Rose as outlined by the SE course. Additionally, several non-technical lessons were learned. The team members gained experience in interpersonal communication, as all members had the opportunity to participate in developmental and post-production interviews with the client. Each team member maintained a full academic schedule at the time of project development, and some members also had work and extra-curricular commitments to balance as well. As each team member possessed an excellent scholastic record, many were challenged by the prospect of delegating responsibilities, a necessary task in a project of this size. Learning to work with the strengths and weaknesses of each individual contributed to the group learning experience. The team members agreed that working as a team, as well as management of time and personal resources were highly useful skills learned over the course of the project.

The value of IT lies in the application domain. Exposure to such intricate systems offers considerable challenges in learning a new application and effectively applying the analysis and design techniques learned in the course

Figure 5: Package Diagram, Milisen Prescriptive Articulation Test



6. CONCLUSIONS

The system described is a form of decision support system used for designing the treatment plan in a clinical area. The system provides an initial treatment plan and allows the clinicians to modify the plan as treatment progresses. Object-oriented techniques and tools were deployed in the design and development of the system, allowing the team to focus on critical use cases first and then add others as time permitted.

Dr Milisen is in his 90's. This system captures his essential contributions to the treatment of speech articulation deficiencies. The authors along with the client consider it a fitting tribute to Dr. Milisen.

References

[1] Milisen, R. Chapter 25, "Methods of Evaluation and Diagnosis of Speech Disorders", in *Handbook of Speech Pathology*, Ed. L.E. Travis, Appleton Century Crofts, New York: (1971).

[2] Bruegge, B. and Dutoit, A.H., Object-*Oriented Software Engineering*, Prentice Hall, 2000.