

PROJECT GROUP ASSIGNMENT SYSTEM

Kim Doyle, Susan Kroha, Arunima Palchowdhury, and Wei Xu
Under the guidance of Professor Charles Tappert
Pace University

Abstract

This paper describes a Web-based interactive system for creating teams (groups) for project work in industry or academia. The system allows project team members and managers to use a Web interface to enter projects and preference information into a database, and then allows the project manager to execute an algorithm that creates the teams. This system addresses problems of the usual manual team creation process which can be tedious, time consuming, perceived as unfair, and may not get a good mix of expertise. Potential team members securely log into the system and complete a survey form to record their project preferences, availability times and preferred locations for team meetings, and their academic and/or professional experience. The project manager controls information about the project: suggested team size, client information, and parameter weights for team candidate project preference, experience, geography, and availability. After the candidates enter their preferences, the project manager can execute a Java program that groups the candidates into teams, and can then reassign team members where necessary and eliminate cancelled projects or team member drop outs. Once the teams are finalized both team members and program managers can view the resulting teams through the Web interface. The system was successfully tested in the capstone Software Engineering course of Pace University's CSIS Master's Degree program by comparing the results of this automated system to the original manual grouping of 45 students into 11 teams.

Introduction

Academic, industrial, and corporate institutions use the team structure for project work and the success of teams is largely due to gathering the right people in forming the teams. Therefore, there is a need for an objective and fair process to efficiently form the teams so that they include the right level and diversity of expertise and experience.

Group formation usually occurs, especially in academic institutions, through people choosing teammates who are their friends or neighbors, and this results in inconsistent team strengths. For example, our observation in team formation for course projects is that while some strong teams are created, there are also unbalanced teams – that is, while members of some teams are all experienced with the technologies required for their projects, members of other teams are new to some of the required technologies. The

popularity of a few projects can also cause others not to be chosen or selected with low preference, and in some cases the course instructor must re-distribute initially over-sized teams. This process is often time consuming and does not always successfully consider factors such as where and when team members can meet for team meetings. Finally, although the team-generation process sometimes considers the diversity of talent and experience, there can be difficulty in identifying everyone's talents.

Although there is little past work on this problem, an automated group assignment system was developed and used at Rutgers-Camden with a diverse population of student team members having diverse schedules [1]. A computer program written in PASCAL assigned candidate team members to groups and was used in six classes over a four-year period. For this system the student's time schedule was the most important parameter, although project experiences and student preferences were also considered. Team member preferences are gathered during class-time and then typed into an input file for the stand-alone program.

In contrast, our system offers added flexibility to handle a greater number of parameters, and it allows the program manager to assign weights to them. Our grouping algorithm is coded in Java for better portability and we use a Web interface to allow candidate team members to independently enter their preferences directly into a database that is later accessed by the algorithm.

Methodology

Our methodology was to develop an interactive system that allows candidate team members to enter information about themselves, to assist the project manager in assigning candidate team members to project groups based on their interest, experience, geography and availability time schedule. The new system address the problems of the current manual process that can be time consuming, perceived as unfair, may not get a good mix of expertise, and can be inefficient.

The system includes:

- A database with candidate team member information,
- A web-based survey for team members to enter information,
- A java-coded algorithm to process the data,
- A mid-tier that interfaces between the java program and the database, and
- An output that will be written to a table storing team members and their group assignment.
- A web interface for the algorithm

A team member can only belong to one group for a particular a course. Membership in different groups in different courses is permitted.

Figure 1 shows the system diagram that includes the web interface, relational database, and algorithm.

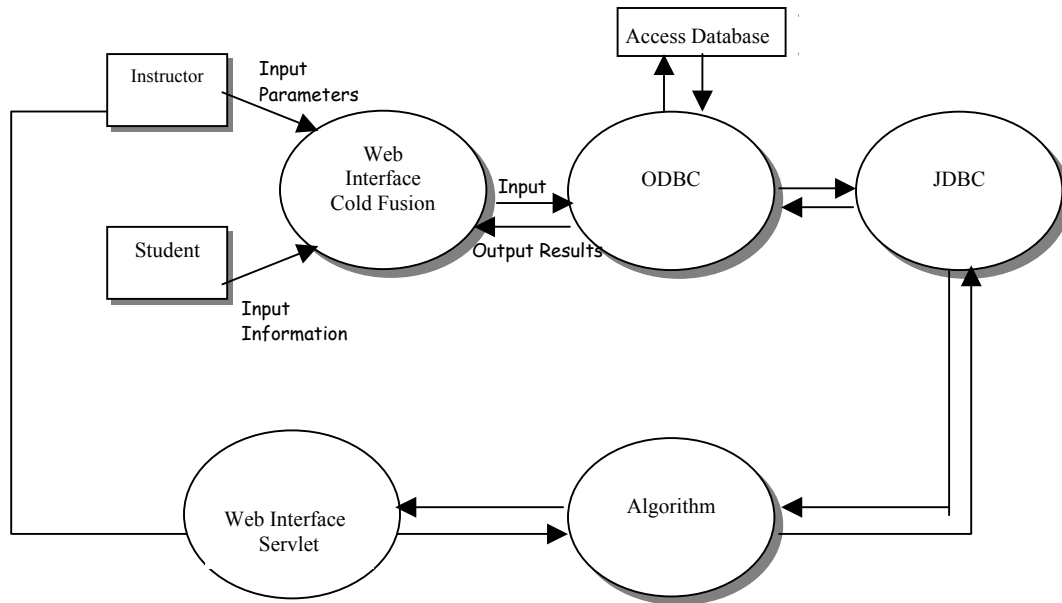


Figure 1. System Diagram

Web Interface

The purpose of website is to collect candidate team member input regarding the four parameters (project preference, professional and academic experience, location preferences, & schedule availability for team meetings) and to display retrieved output (project teams).

The theme of the website is student and instructor pages with links to forms and reports. Access to the website is through user ID and password validation; once validated the user can use hyperlinks to get to the survey, project team output, run algorithm or alter groups (if it is the instructor's ID). The website was developed in Internet Explorer version 5.5, Cold Fusion version 4.0, and Dreamweaver version 3.0 while connected remotely to a MS Access 97 database on a server at Pace University.

The site map is as follows:

Index page has links to:

- Register - The UserID is recommended to be the user's email address. Their password is encrypted using ColdFusion's hashing algorithm, and stored in the database.
- Login – A temporary cookie is used to secure the session. This is a secure login so team members' preferences remain private and tamper-proof. After logging in, the user is directed to the Home page.

Home page has links to:

- Instructor page
- Student page.

The instructor page has 4 sections for managing information:

Manage students

- View student preferences – Students fill out a survey to record their first five choices of project that they want to work on, time they can meet, place they can meet and their top 5 levels of experience.
- Delete students – to manage test data, delete students that are no longer enrolled in the course, and to clear out semester information in preparation for new data.
- View students – This will display students that have registered in the system, regardless of whether they've filled out any preferences.

Manage clients

- Enter clients – Instructor can record information about clients including name, email, phone number.
- Delete clients – To delete test data, remove clients no longer sponsoring projects, or clear out semester information in preparation for new data.
- View clients – This will display all clients entered in the system.

Manage projects

- Enter projects – Instructor can record the project, the course semester in which the project will be worked on, client, and suggested project team size.
- Delete projects – This will delete the project from all tables in the database.
- View projects – This will display the project, client, and semester in which it is offered.

Manage teams

- Create teams – This is a link to a java servlet login page. The instructor logs in, selects the course and semester data for which he wants to run the java algorithm to create teams, runs the algorithm, and then views the teams created.
- Modify teams – After running the algorithm, the instructor may choose to redistribute team members among other teams.
- View teams - This page display groups formed by the algorithm and modified by the instructor.
- Modify parameter weights – This page allows the instructor to modify the weights of project, time, location and experience parameters from 1 to 10.

The student page has links to:

- Student survey – This page will forward students to a page where they select the semester and course for which they want to specify their preferences, and then to the survey page where they select their 1st through 5th preferences for project, location, time, and experience. If students previously filled out the survey for that course and

semester, they are advised and prompted to select a different course and survey or update their preferences. Duplicate entries are not permitted on the survey page, and students are asked to correct any such entries.

- ‘Update survey’ page to change their parameter preferences (project, location, time, experience).
- ‘View teams’ page to display report of groups formed

Relational Database

The MS Access 97 database was developed to adapt to changes in addition, deletion and changes of preferences and weights. The database provides storage of participants’ and clients’ information, and provides potential for statistic analysis of the success rate of projects and satisfaction of participants. The table relationships of the database are depicted in Figure 2.

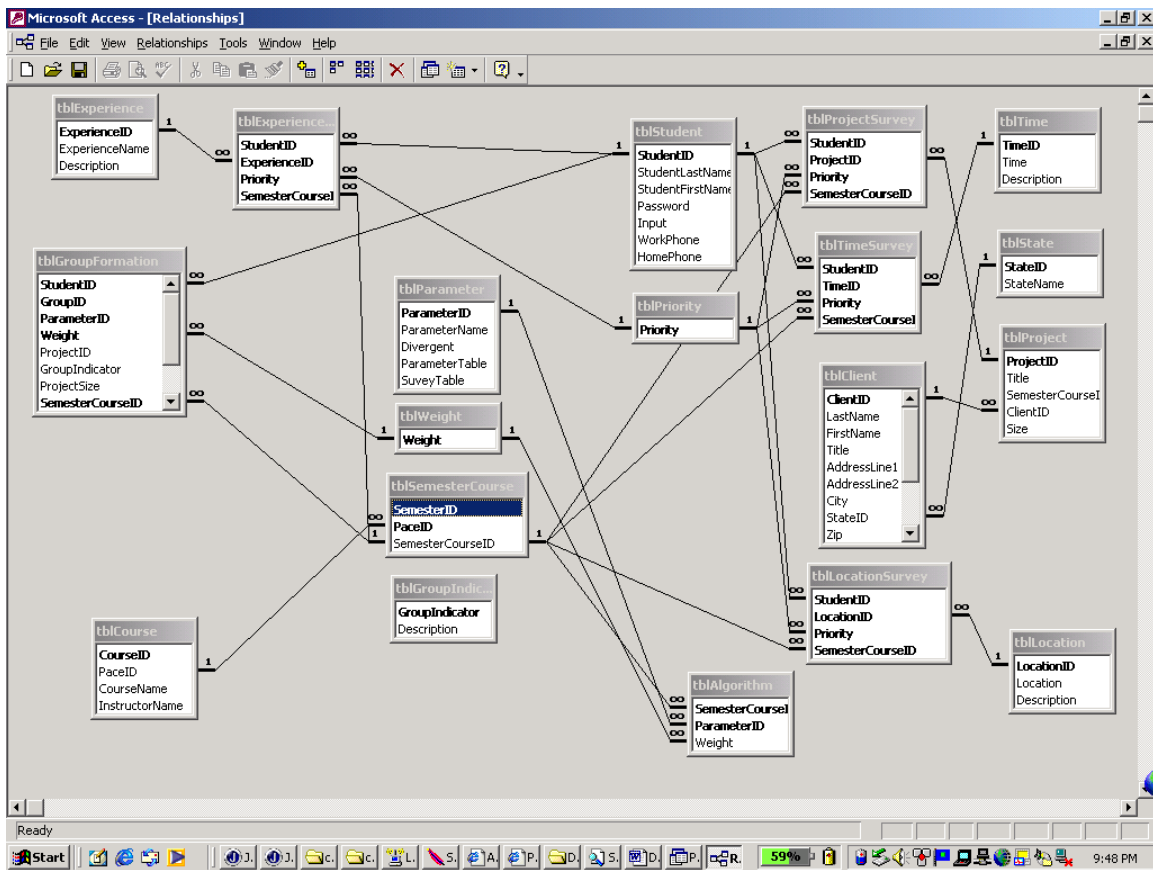


Figure 2. Database Table Relationships

Algorithm

Algorithm Overview

The Java algorithm has a web interface via Java Servlets, as shown in the subsystem diagram (Figure 3). The Servlet interface allows easy access to the database through browser [2].

The algorithm also includes convergent and divergent classes, as shown in the major class diagram (Figure 4). The algorithm uses preferences and parameter weights to create teams. Three parameters are used in the convergent processes consecutively: groups of people with similar preferences for project, location and time are created, and the output from each process is used as the input for the next process. One parameter is used in the divergent process: groups of people with diverse professional and academic experience are created.

The first process will form teams based on common project preferences (for convergent algorithm). Two subsequent processes of the convergent algorithm and one process of divergent algorithm will form teams based on the team formed in the previous processes and preferences. During the last three processes, larger groups, formed based on first project preferences, are broken down into smaller group with 3 to 5 members. The students that are still not assigned to a project are then assigned to a small group with 2 members, if one of their project preference matches with the project of that group. The algorithm attempts to add to the two-member group first, if the student's project preference matches that of the group. The students with no matching project preferences are then assigned to existing groups with a group size of 4 or less, starting with the smallest group. The algorithm attempts to match the most preferable project preference first. If some students still do not have any preferences that match any of the project groups, they will be assigned to a relatively smaller group. The output of this last pass is the final group assignment. The subsystem diagram is shown in Figure 3.

Algorithm Detail

The algorithm forms groups of students based on to their preferences of projects, preferable time to meet the group, preferable place to meet the group and their expertise. The algorithm is very flexible, it runs based on the weight assigned by the professor for the 3 parameters (time preference, location preference, experience). The algorithm works as follows:

The algorithm selects the students enrolled in the particular semester and course that the professor selects in order to form the project team groups. The professor may have student data in the database for students enrolled in other courses and run the algorithm for that group of students simultaneously. The professor can assign different parameter weights for different courses, or the same courses but offered at a different time.

The project choice is the first criteria for the formation of the groups. It extracts the first choice of project for the students and runs convergent algorithm. The convergent algorithm selects students with the same choice of that parameter (in this case it is the project choice) and forms groups. So each group now contains students with the same

first preference choice of project. It does not take into account the size of the group at this stage.

After the first run, if the size of the group is between 3 and 5, that group is the final for that project. But if the size of any particular group is greater than 5, the next parameter (which is determined by the weigh assigned by the professor) is taken into account. Depending on the parameter, the convergent or divergent algorithm is used. Larger teams will be assigned a most desirable parameter for the subsequent process, based on the members' most popular preferences (such as a location to meet, or a time to meet, or professional experience). The first preference for a parameter will be assigned 1000 points, and subsequent preferences will be assigned a score of 1000 divided by the order of the preference (for example, the second preference will be given a score of 500, the third preference will be given a score of 333.3).

The score and convergent algorithm checks the preference of the students of a large group, and the most popular preference. The convergent algorithm forms a group of students having the same preference as the most popular parameter. The algorithm will take away the students whose choices do not match, but keep a minimum group size of 5. The algorithm will keep all students with the same popular preference in the same group. The next process will score the modified group, and eliminate students from the group, if their preferences do not match the popular preference. However, the last process will generate a group with 5 students.

The score and divergent algorithm works little differently : it checks the choices of that particular parameter (in this case experience) of the students of a particular group and checks if there are any repetitions of the choices. The algorithm attempts to remove students with the same experience away from the group. For the experience parameter divergent algorithm is used because we want to have a group with diverse experience.

The convergent algorithm is used for the other two parameters (time and location preference) because the students of a group would want to meet at the same time and at the same location. However, the divergent and convergent algorithm will no longer remove students from a group if the group size is 5, while working in conjunction with the scoring algorithm.

Finally, the students, who are removed from a large group, need to have a project and group. We consider the second through fifth choices of projects of these students and appropriately assign them to the smaller groups, if one of their project preference matches that of the group. Students with no matching first project preferences are also assigned to a group, based on their second to fifth project preference. In this way, the algorithm attempts not only to fairly assign all students to projects that they have selected in their list of preferences, but also to generate good project groups with students with like time and location preferences but diverse experience and skills, in sizes of 3 to 5 people.

Subsystem:

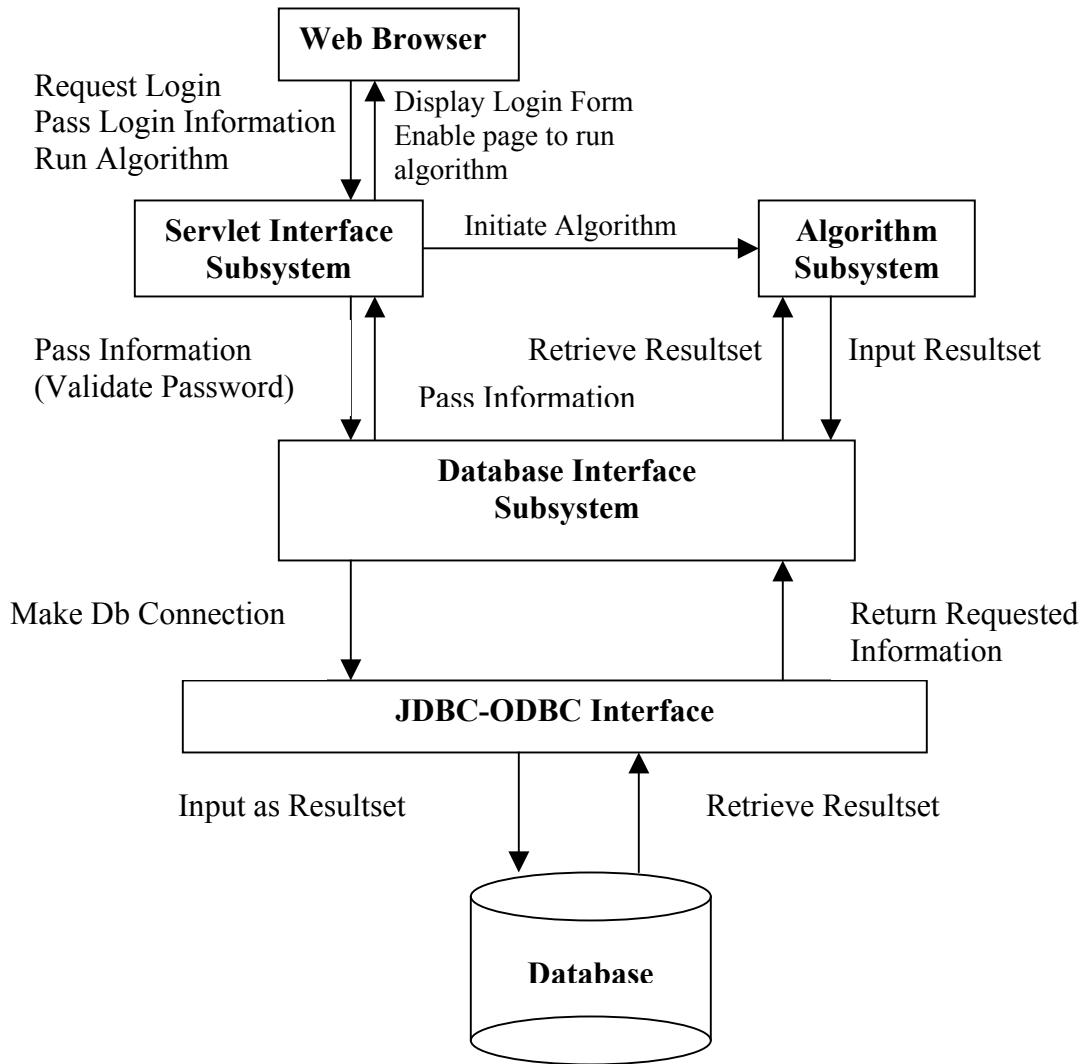


Figure 3. The Subsystem Diagram

Major Java Classes:

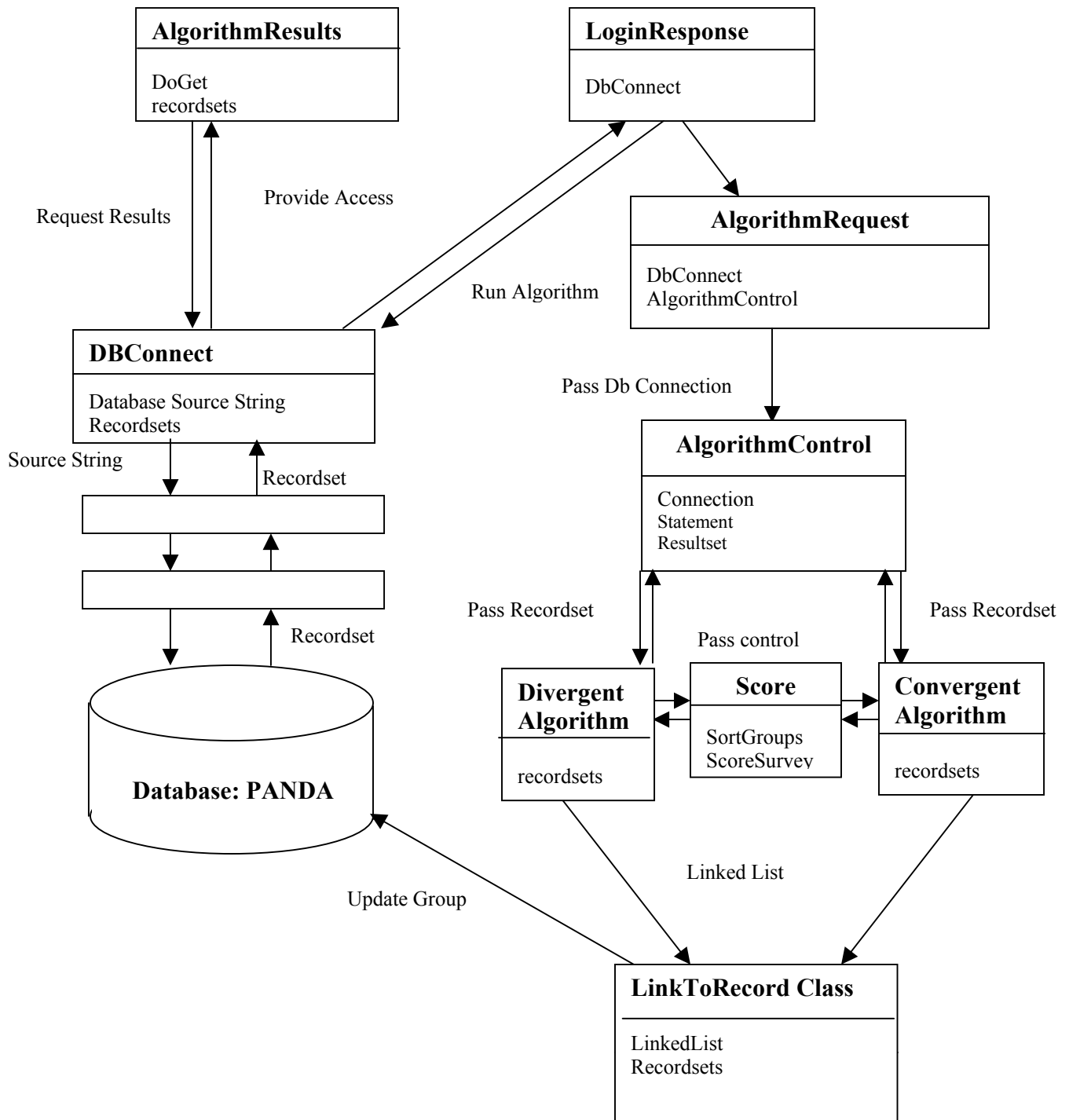


Figure 4. Major Java Classes

Results

Initial results demonstrate that the algorithm is experimentally successful in forming groups. The database is robust and very flexible - parameters can be changed without having to change the algorithm. The interface is efficient – we are able to access the website and add and update records on the database, and correctly display information. The design specifications of the system for this semester have been met, and we successfully demonstrated a prototype to our client.

Conclusions and Recommendations

After a successful test in Spring 2002, we recommend that the system be used in the Pace University courses that require project teams. Such system can also be used in other academic and industrial institutes, where project teams are required. Successful implementation of this system is intended to result in teams with balanced, diverse talents with common project interests, able to meet at times and places satisfying to team members.

Should the system prove useful in assigning teams, it can possibly be applied to industrial and corporate settings where team projects are common.

References

- [1] Redmond, M.A. “A Computer Program to Aid Assignment of Student Project Groups”, in Proceedings of the 32nd SIGCSE Technical Symposium (2001), ACM, pp134-138.
- [2] Hall, M. and Brown, L. “Core Web Programming”, in the Sun Microsystems Press Java Series, 2001, pp 874-907.
- [3] Brooks-Bilson, R. "Programming Coldfusion", O'Reilly & Associates, Incorporated, August 2001.