RAGS
Recreational Activity Guidance System

Lars Olsén     Jonas Johansson
c99lon@cs.umu.se  c99jjn@cs.umu.se

- 9 juni 2005-

Abstract
Inline skating, running and other outdoor recreational activities are popular nowadays. Umeå as a university city has a young and active population and a wide range of bicycle lanes and running tracks. But where are they and which ones are the most popular at a given time? Newcomers to Umeå naturally have a hard time figuring out the best routes for their particular activity. RAGS is a web based interface that allows users to input their favorite routes as well as viewing other users routes to get inspiration for their next recreational activity.

1 Introduction

Inline skating, running and other outdoor recreational activities are popular nowadays. Umeå as a university city has a young and active population and a wide range of bicycle lanes and running tracks. But where are they and which ones are the most popular at a given time? Newcomers to Umeå naturally have a hard time figuring out the best routes for their particular activity.

RAGS (Recreational Activity Guidance System) basic functionality is to allow users to input their favourite routes when inlineing, running and biking so that other users can get tips and be inspired in their recreational activity. Apart from adding and searching for routes, users can also grade existing routes and report changes in road condition along different segments.

Features not supported yet is the ability to see history of road conditions along routes and the ability to input routes on more maps than the one provided so far.

The chosen database system for the application is MySQL and the application itself is implemented in PHP. PHP offers a easy to use session handling and is simple to program in.
2 Approach

The work started by establishing requirements and features of the application. The established requirements were:

- The interaction method with the system should be through a web based graphical user interface.
- A clickable map should be the main tool for input of spatial information.
- Users of the system should be separated into regular users and administrators.
- The regular user should be able to:
  - Add own routes.
  - View and grade existing routes, potentially from other users as well.
  - Report changes in road conditions.
  - Choose the different layers of information presented on the map.
  - View the road conditions on a route at a specific point in time.
- The administrator(s) should, in addition to the regular user commands, be able to:
  - Input base coordinates.
  - Input segments.
  - Input features.

The basic components of a map are base coordinates and segments. A map can be seen as a graph, where base coordinates are vertices and segments are edges. A route consists of one or more connected segments. The reason that only the administrator should be able to input base coordinates and segments is to prevent populating the database with invalid routes, for example crossing through buildings. A map can contain features that makes a route more interesting or challenging. For example, features can be an ice cream shop, stairs or benches to grind on.

An Extended Entity-Relationship (EER) diagram was created, showing the basic structure of the database at a conceptual level. The notion of ISA relationships was used to describe generalization of common entities. When translating from the conceptual level (i.e. the EER diagram) to the representational level (i.e. a SQL database schema) the conventional algorithm was not followed entirely because of practical reasons.

MySQL was chosen as the database system to use. The reason for selecting MySQL before any other system (e.g. PostgreSQL) was that the server hosting the system already used MySQL and the spatial components needed could be represented with the MySQL spatial extension. The system was implemented with PHP, using the graphical library GD2 for easy creation and handling of images on the fly.
Sessions in PHP was used to handle user management. The user creates a
account by registering on the site. When logging in, a session is created that
keeps track of necessary states and information. The ease of session handling
was, along with GD2, a strong argument for choosing PHP.

3 Results

The resulting application, RAGS, makes it possible for users to share their
favorite routes and getting suggestions of possible routes from other users. This
system fulfills almost all of the requirements that was established in the first
phase of the project. A snapshot of the graphical user interface is shown in figure
1.

![RAGS GUI](image.png)

**Figure 1: RAGS GUI**

Segments along a chosen route is displayed in various colors depending on the
surface type so a user easily can get a quick and easy idea of the characteristics
of the surface of a route. The end of a route is indicated with a checked flag and
features included are highlighted. On the side of the map a legend of color codes
are shown, along with information about the selected route; included features,
grade and distance. There are a couple of possible search criteria for routes. It
is possible to search for routes containing specific features. It is also possible to specify that only routes from a certain user should be fetched.

There are however limitations to the usability of the system. The response time when inputting data by clicking on the map is slow because forms are used, having the effect of reloading the page, and thereby the image, every time. At the moment, only a small image showing the Umeå University campus area is available. The ability to see the road condition of a route at a given time is also absent at the moment. However, all the information necessary to provide this feature is stored in the database.

4 Discussion

At first, some comments on the created database. The database is mainly used as storage of data. There is a high lack of use of other functionality provided by database systems in general, such as foreign key constraints. Since there was an interest in using the spatial functionality in MySQL, a specific database engine called MyISAM was used. However, in the MYISAM engine, foreign key constraints are not yet implemented. Foreign key constraints are therefore handled outside the database by inserting data in proper manner.

Several of the spatial functions that were thought to be available were not fully implemented yet and where thus not applicable, for example distance calculation between two points. Own functions for these calculations were used instead.

The performance of the application is limited mainly because of frequent re-loads of the web page when interacting with the system. The main source of extended response time is loading the map image and this is not scalable if larger images are to be used. When a user clicks on the map and the closest point is to be found, all points of the map are examined. The number of examined points could be reduced with the use of an indexing structure such as an R-tree. Java Script is another possibility to increase performance since it could assist in decreasing the frequency of page re-loads.

Concerning the GUI, the possibility to add more maps would make it possible to create more interesting routes since the area on the map of the Umeå University campus is rather small. Since segments are temporized, it would be a simple task to add the ability to see the road condition of a route at a given time. Also, the ability to express more advanced search criteria could be useful; such as the possibility to only display routes having a distance within a certain length, not containing any obstacles (i.e. road constructions).

5 Conclusion

The intention was good, but the outcome was rather jaded! We felt that we had good intentions with this project when we wanted to make use of topics relevant to the course, such as spatial and temporal concepts. However, due to limita-
tions of spatial functions in MySQL, in contrast to what we first thought, we couldn’t implement useful functions for the user. If we would have skipped the spatial extension to MySQL it would have been possible to make use of foreign key constraints, which would have been better from a point of view relevant to this course.

We also felt a shortage of time. This could possibly have been avoided if we had been a group of three or four members. There’s also always the question of how to divide the workload between group members, splitting up work in disjoint parts prevents members from learning from all parts of the project. There was too much of two persons, one keyboardstyle programming.

As a final conclusion: The project idea should have been further developed and more thoroughly examined prior to implementation.