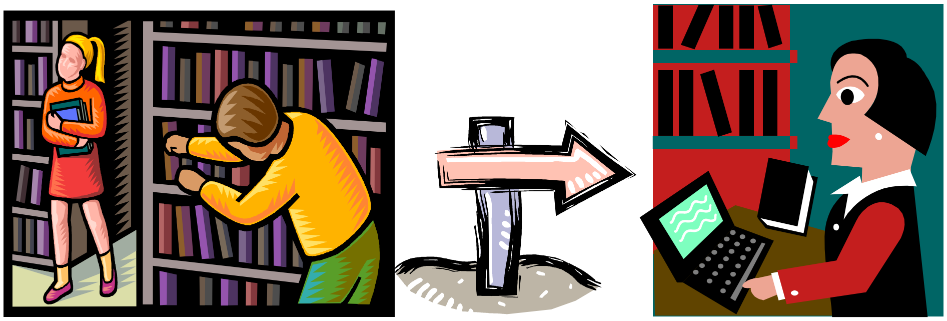
PROJECT REPORT

ON
ATLAS (Smart Library)

a pervasive context aware application



Submitted By :
Nilamkumar Patel
Chintan Shah

Guided By :
Nehal Mehta
Phd Candidate – Dr. Sun's SCS research group - www.iit.edu/~scs

Professor Xian-He Sun

Computer Science Department
Illinois Institute of Technology

Computer Science Department
Illinois Institute of Technology

INDEX

Encl:
APPENDIX-I Project Proposal
APPENDIX-II Software Requirement Specification
APPENDIX-III Technical Design document

Abstract

In library the searching book is very hectic process, if one doe not have proper direction to reach the rack which held the book. Generally library maintains call number to map relation to get the rack where book is hold. But there are many racks in library and it is very hard to find the particular rack through map. Some good library provides the online map with highlighted rack which contains book. But this is also hard to remember the map as the system is not web based and can run on library's static PC. In Pervasive computing, context aware development is going on. So by using that technology, we can improve the book searching process. Here, we developed the system smart library, which uses the location aware technology and get the current location of user and give the path to the rack which holds the book. This solution can help tremendous to the library user in searching book.

1. Introduction

1.1 Project Name

ATLAS (Smart Library – a pervasive context aware application)

1.2 Goal

To develop smart Library Environment as the Library is one of the integral parts of any educational environment and used by students as well as faculty members.

1.3 Description

The main purpose of the project is to provide the users a smart library environment, where they can search book online and can be directed with route information to the rack where the book is located. Location awareness has great potential in this application, compared to any other campus tour or city guide application. This is because, in general tour, once user is familiar with the location, in future they might not need this facility often. But in environment such as library, location awareness can be of great help to find books on daily basis. On the other hand the same application can be extended and integrated with many other library resources and make them context aware.

The name ATLAS was given on the basis of the functionalities provided by the system. It is all about concerned with routing the path from the user's current location to the location of the book in the rack.

2. Problem Statement

All students have accessed library information system to access books. Currently most of the library supports computer based book search engines to find the correct book within fractions of seconds. This project will try to make easy library flow using the pervasive context aware technology.

2.1 Current Scenario

1. Analyzing the current procedure for book search and book checkout -
2. User first uses one of the computers located in the library – to search the required book based on given criteria.
3. The search engine shows the book result in tabular form with Call number of book [usually very long numbers and so very difficult to remember].
4. The user writes down the book call number or may be multiple book number on a paper. The library staff will provide him floor map- with rack.

2.2 Problem in Existing System

This is very time consuming process and sometime if the library is very big, then it will frustrate user very easily.

NOTE: For Detail project proposal see APPENDIX-I

3. Existing Solution

3.1 Survey

We have done survey in below three libraries.

Galvin Library (Main Campus)

Kent Law Library (Downtown Campus)

Chicago Public Library

We used below questionnaire to get requirement and operational feasibility of the project.

Questionnaires

1) To search book, you have to browse book from catalog. Then you have to note down the call no. Then you have to take map of library from library staff and you have to search rack which has that book.
Do you have any difficulty in following above procedure of book searching?

□ Yes □ No

If Yes then what kind of change you want?

____
____
____
____

2) If you are provided solution like..

"You have one small computer or device will show you the path of the book in the rack of the library from your current location in the library or campus."

According to you, is it good and complete solution?

□ Yes □ No

If No then what kind of solution you think is better?

____
____
____
____

3) Please tick the thing about which you like to be reminded

□ Book Pickup

□ Due Date of Issued book

□ Some people of your class are in library who may help you in your assignment

□ Some people of your favourite subject are in Library who wants to share knowledge with you

□ Other (Specify in below space)

____

____

____

4) What are the other functionalities which you are looking or willing to have in the computerized library system?

____

____

____

____

____

Result of Survey

Question	**Yes**	**No**	**Problem**
1	95%	5%	Find to get location of book in library
2	100%	0%	This is enough.

Note: Question 3 and 4 are for the “Ringer” Project.

This all library have system we mentioned in the problem statement..

3.2 Case Study

Project: Library Management system(LMS)

Institute: Nirma Institute of Technology
Ahmedabad, Gujarat, India.

Comparison of feature with ATLAS:

Feature	**LMS**	**ATLAS**
Book Search	Efficient	Basic(Prototype)
Map	Good	Good
Scalability	**Low**	**High**
Rack Search on Map	By changing color on the map image	Giving destination Rack Path

Path	**Not Available**	**Available**
Application	**Desktop based**	**Web Based**
User Location	**Static**	**Dynamic**
Location Service	**Not Available**	**Available**
Reusability	**None**	**Location aware and Path finding can be reused for other project also**

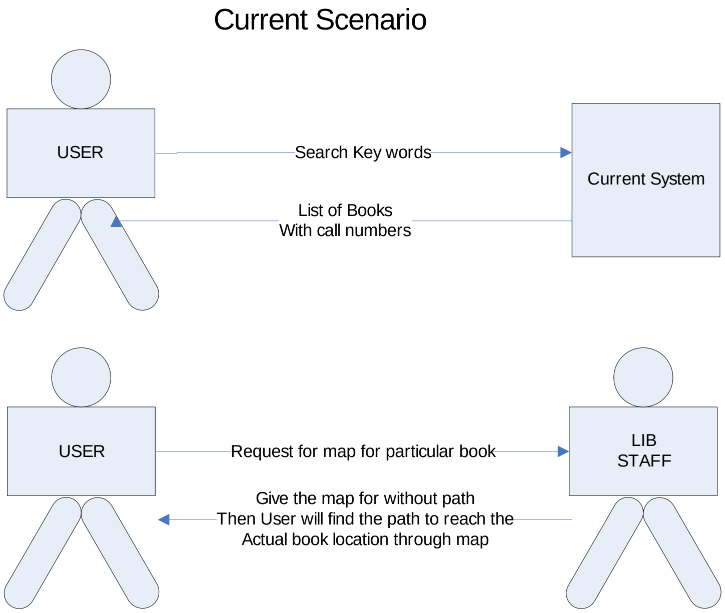
We did not found any advance library system better than above system so we may be one of the first who are using pervasive computing for library system.

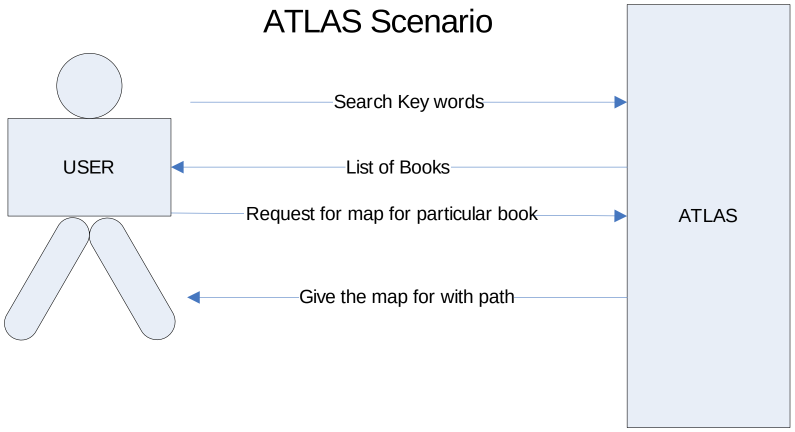
NOTE: For Detail Requirement collection and feasibility study see APPENDIX-II

4. Solution

To help user to do more with doing less, we are proposing smart library environment, where the users can search book online and can be directed with route information to the rack where the book is located. Location awareness has great potential in this application, compared to any other campus tour or city guide application. This is because, in general tour, once user is familiar with the location, in future they might not need this facility often. But in environment such as library, location awareness can be of great help to find books on daily basis. On the other hand the same application can be extended and integrated with many other library resources and make them context aware. The below use case scenario realize the improvement in the process.

4.1.User Scenario and Use-Cases


Current Scenario
USER
Search Key words
Current System
List of Books
With call numbers
USER
Request for map for particular book
LIB
STAFF
Give the map for without path
Then User will find the path to reach the
Actual book location through map


ATLAS Scenario
USER
ATLAS
Search Key words
List of Books
Request for map for particular book
Give the map for with path

4.2. User Functional Requirements

ID	Description	Rank
UF1	The system should provide support for searching book through browser-based interface.	1
UF2	If the book is in the same library where user is, then system should show the path from his current location to the rack where the book is placed.	1
UF3	If the book is on the different floor, where the user is standing, then the system would show the path of the books from the current location to the location where the book is placed.	2
UF4	The system should be web based application.	1
UF5	The system should work on the user's own devices.	3
UF6	The system should work at outdoor also (just searching, no path finding).	2
UF7	It should fetch the content information from Amazon web service.	3
UF8	User's account should be maintained properly by the system.	2

4.3.User Non-Functional Requirements

Requirement	Description	Rank
UNF1	Security: The user accounts and the critical information of library should be secure.	3
UNF2	Availability: The system should be available at any time and any place where internet is available.	1
UNF3	Efficiency: The path should be shortest	2
UNF4	Reliability: The location of book should be correct.	1

NOTE: For Detail Requirement Elicitation see APPENDIX-II

4.4.Tools & Platform:

Initially, we started to work with .Net environment but as we started we had resource and incensing problem so we choose to go with open source tools. The reason to choose .Net is to get complete development environment for Database, web service and web based application. But as we had some problems we started with below open source tools which are also same efficient as .Net (individually).

4.4.1. PHP (GD Library)

PHP is good web based application tool which is object oriented tools. Moreover, it is easily wrapped as web services and it has good compatibility with other web services. GD library we chose to support image editing in web based application.

4.4.2. JAVA

As we have JAVA API for EKAHAU Positioning Engine, we also required to make web service in JAVA.

4.4.3. MySQL

MySQL is good & open source Database application, which is also reliable.

4.4.4. Apache Tomcat

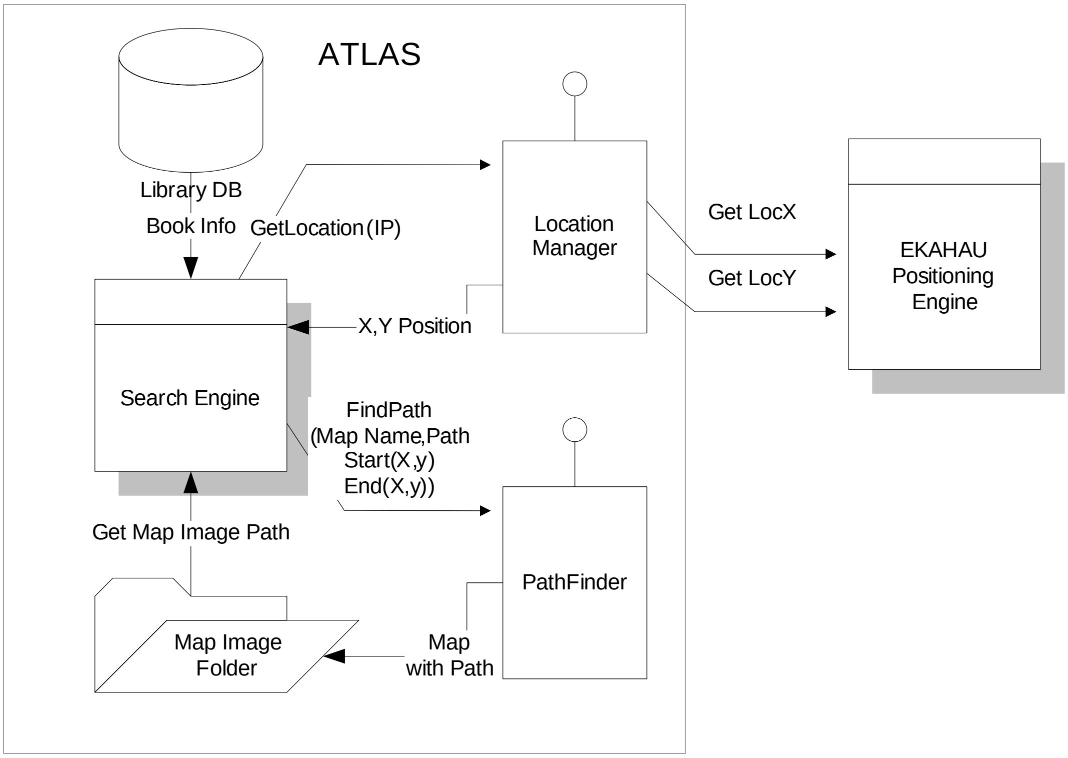
To wrap JAVA application as web service we need to have Apache Tomcat web sever.

This is Platform independent as it is web based application.

4.5.Architecture

To implement above requirements, we designed below architecture

Architecture



This is the over all architecture of the ATLAS.

Subsystems

- Search Engine
- Location Manager
- PathFinder

Database

- Library Database
- Map Image Folder (This folder has map image file and the map db file which consist of nodes of map).

External Component

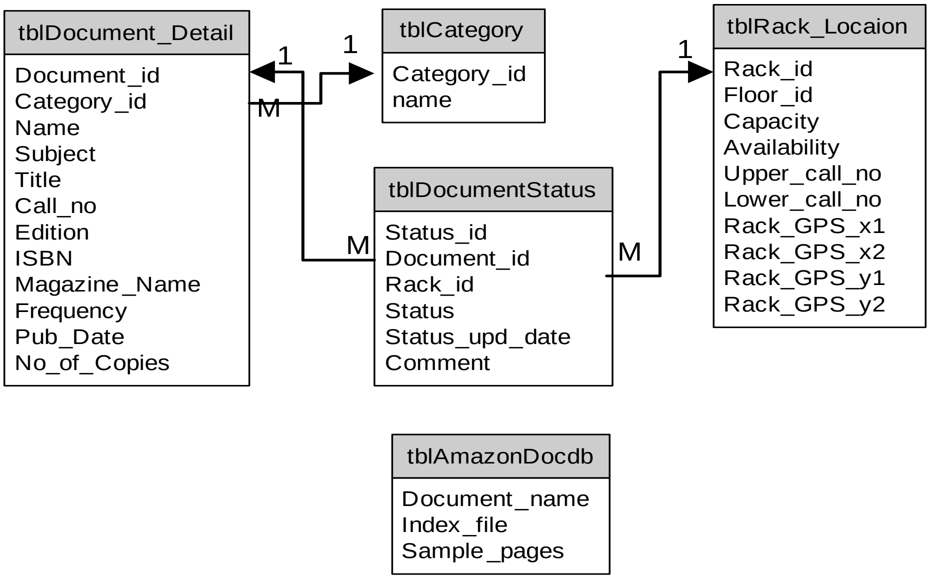
- EKAHAU Positioning Engine (This is responsible for tracking and giving the current location of the user).

Now, lets discuss design of each component

4.5.1. Library Database Design

This is the design of database from which the search engine will get the data related books.

E-R Diagrams



4.5.2. Library Map DB file Design

Library Map DB file is a file which stores the node graph of the map. It stores the nodes of the map with its x and y position and with its connected neighbors' list.

Extension: .db

Name: Name should be same as the map image file name. for example if image name id LIBMAP.jpg then the DB file should have name LIBMAP.db

Format:

Node	X	Y	n[0]	n[1]	n[2]	n[3]	n[4]...........		N[9]
1	100	100	2	3	0	0	0	....	0
2	120	100	3	4	0	0	0	....	0
...	.....	....	...	...	..	..	..	....	..

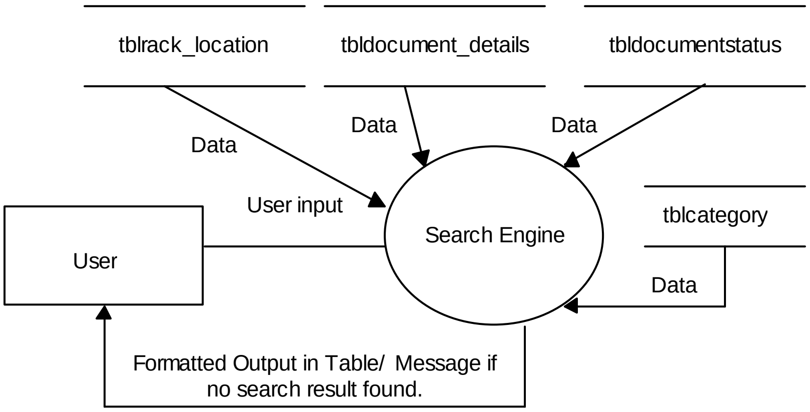
4.5.3. Search Engine Design

This module is responsible for searching the books from user's keyword. This module searches the books and also searches the rack id and position of rack from the database. Moreover as shown in architecture diagram, this module also call the location manager and pathfinder module to generate the path enabled library map on the browser.

As this module is fully database module. So let's see Data Flow Diagrams (DFD) of this system.

NOTE: For Detail Design of Search Engine and feasibility study see APPENDIX-III

Level 1 DFD



4.5.4. Location Manager

Location Manager is component which is responsible for giving location of particular Device. This location manager uses the EKAHAU Positioning Engine Server, which tracks the device on map. So let's first understand how EKAHAU gets the Location.

EKAHAU has three components.

1. Positioning Engine: This engine is always running and tracking the devices which has EKAHAU client installed. It continuously communicating with client and gets it position after configured time.
2. Manager: This component is allow us to load the map and calibrate it with actual location. This also shows the map and tracked device.
3. Client: client is installed on user's machine. This client continuously sending it position to Location Engine.

EKAHAU is providing some JAVA API to get location from engine. So we can use that API and get the location manager.

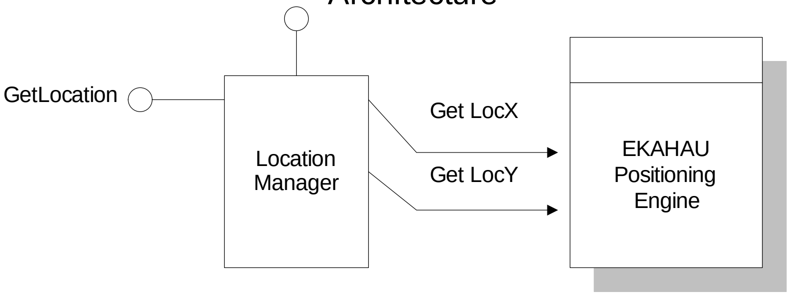
Location manager is basically component which is responsible for talking to the EKAHAU Engine through API and gets the location. As these APIs are in

JAVA, we should implement this location manager in JAVA. Then we wrap it as Web-Service and can be used by any browser client.

We are implementing it as a web service, because the location manager component can be used in other application like Campus Tour and RINGER. So, it will increase the reusability of the developed work.

The below diagram shows how it communicates with EKAHAU engine.

Location Manager Architecture



4.5.5. PathFinder Design

We have studied below algorithms.

- A* Algorithm
- Depth First Search
- Krushkal algorithm
- Best First Search

The above all algorithms are basically designed for path finding in Gaming Problem. So they are very much complicated and time consuming. In our problem domain, there is no such complex algorithm required. Because, In our Problem domain, all the path are open. No single node is end point. All are node are connected to each other. So there is no wrong path will be selected. So we modified Best first Search algorithm and made it simple, so it will take less time. Below is our algorithm.

NOTE: For Detail Design of pathfinder objects see APPENDIX-III

Algorithm

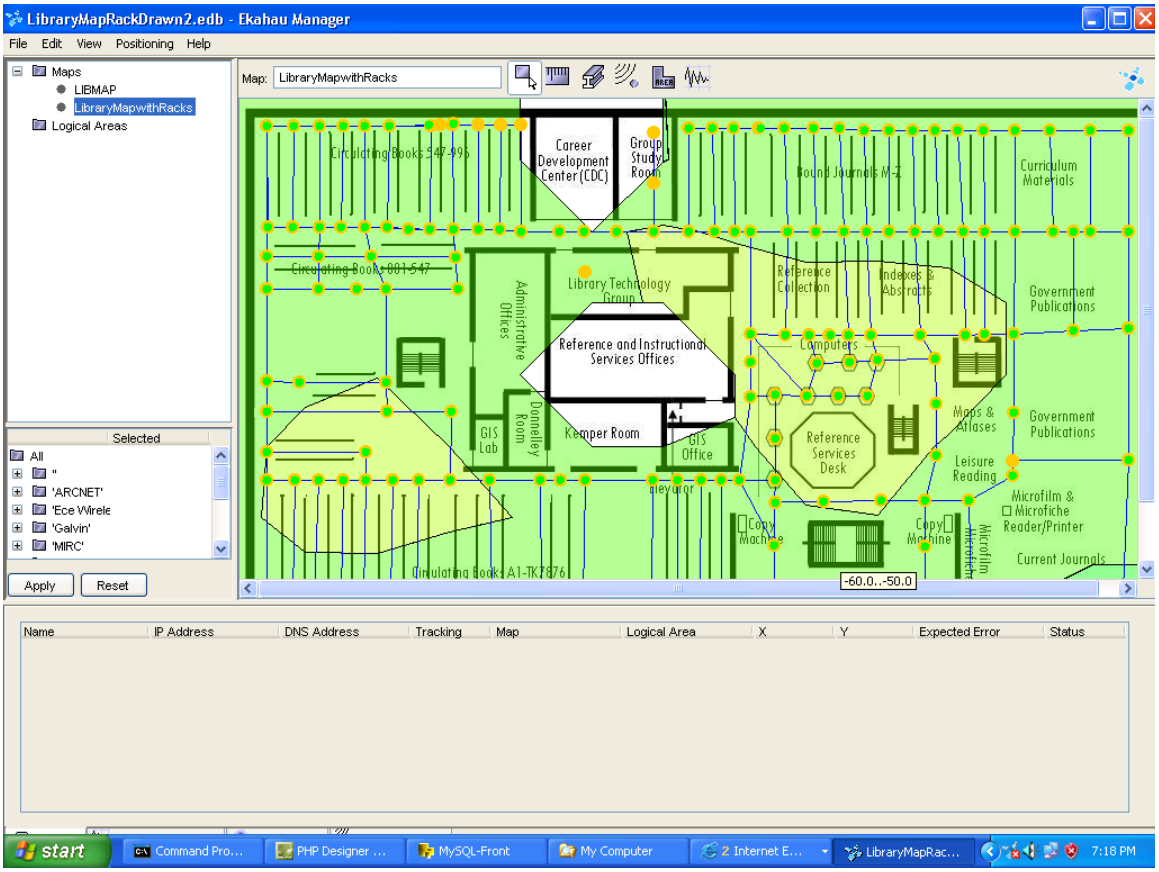
```
Procedure findPath(start,end)
Input: start: Starting point
       end: Destination Point
OutPut: Path: Array of Nodes, ImageFile with path

Steps:
    init = start //Store first node
    call findP(start,end) and store output in path
    callDrawPath(path) //which draws the path in file and store it
    return path

Procedure findP(start,end)
Input: start: Starting point
        end: Destination Point
Output: Update path array
Steps:
    Add start in array path
    if start = end
        return
    Get the neighbors of start in neighbours array
    i = 0
    flag=0
    for i= 0  to number of neighbors
       if ith neighbor in InClseList
                continue
        flag=1
        g=Distance(ith neighbor, end)
        f = g
        if f is < minimum
                 minimum = f
                next = ith neighbor
        if flag=0
                backtrack and remove start from path
                next = parent of start
     AddInCloseList(start)
    call findP(next,end)

Procedure: DrawPath(Path)
Input:   path array
Output: Path drawn image file

Steps:
    x1 = path[0].x //assign fisrt node’s X cooridnate
    y1 = path[0].y // assign fisrt node’s X coridnate
    loop(steps 4-8) from i = 2nd node of path to end of path
        x2 = path[i].x
        y2 = path[i].y
        Drawline(x1,y1,x2,y2)
        x1=x2
        y1=y2
```

5. Implementation & Result

In Order to implement above design, we have followed the below steps:

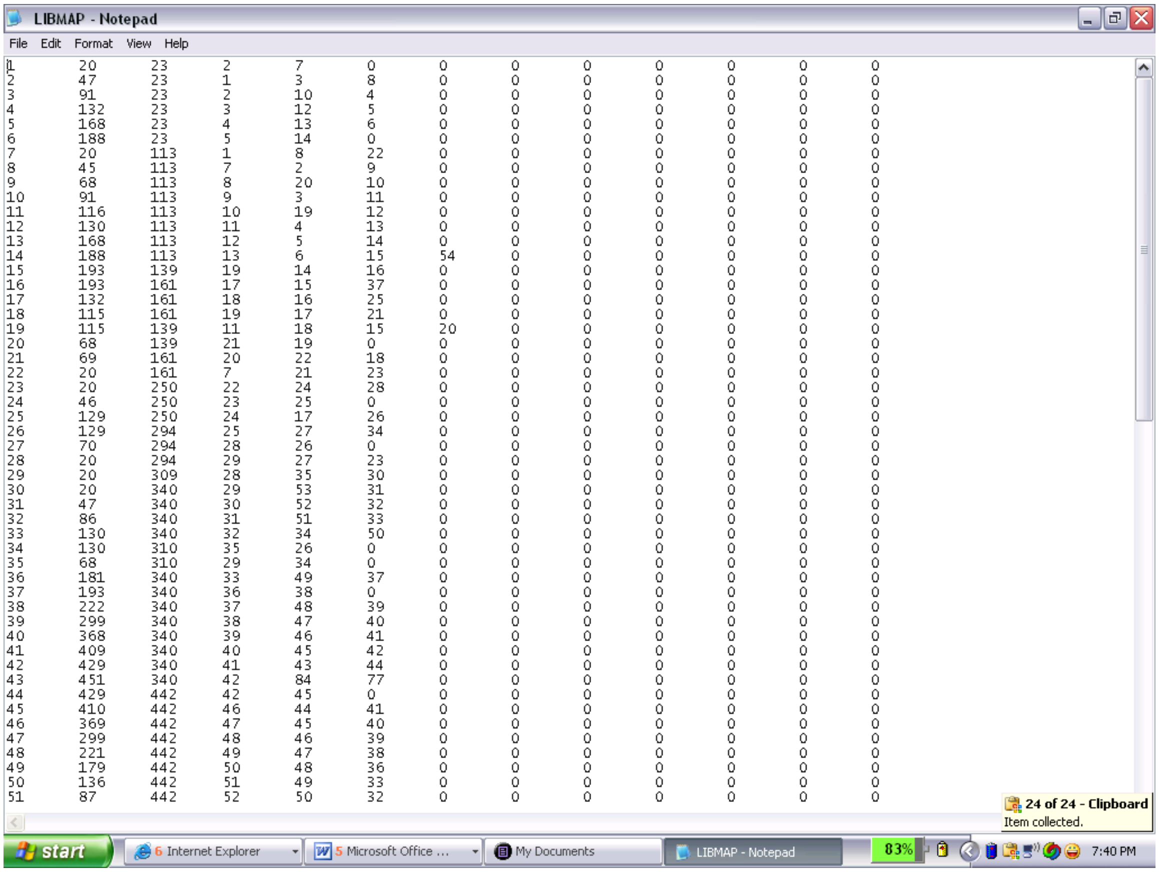
5.1. Loading and calibrating the library map in EKAHAU positioning Engine

We load the library map in EKAHAU Positioning engine. Then we calibrated the map by doing site surveying of library. Below screen capture is the site surveyed and calibrated map of library.



5.2. Generating the Library Map DB file.

From above calibrated map, we developed CSV file which has structure mentioned in the Solution section. This file has all the node of above calibrated map.

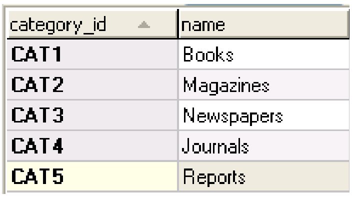


```
1       20      23      2       7       0       0       0       0       0       0       0       0
2       47      23      1       3       8       0       0       0       0       0       0       0
3       91      23      2       10      4       0       0       0       0       0       0       0
4       132     23      3       12      5       0       0       0       0       0       0       0
5       168     23      4       13      6       0       0       0       0       0       0       0
6       188     23      5       14      0       0       0       0       0       0       0       0
7       20      113     1       8       22      0       0       0       0       0       0       0
8       45      113     7       2       9       0       0       0       0       0       0       0
9       68      113     8       20      10      0       0       0       0       0       0       0
10      91      113     9       3       11      0       0       0       0       0       0       0
11      116     113     10      19      12      0       0       0       0       0       0       0
12      130     113     11      4       13      0       0       0       0       0       0       0
13      168     113     12      5       14      0       0       0       0       0       0       0
14      188     113     13      6       15      54      0       0       0       0       0       0
15      193     139     19      14      16      0       0       0       0       0       0       0
16      193     161     17      15      37      0       0       0       0       0       0       0
17      132     161     18      16      25      0       0       0       0       0       0       0
18      115     161     19      17      21      0       0       0       0       0       0       0
19      115     139     11      18      15      20      0       0       0       0       0       0
20      68      139     21      19      0       0       0       0       0       0       0       0
21      69      161     20      22      18      0       0       0       0       0       0       0
22      20      161     7       21      23      0       0       0       0       0       0       0
23      20      250     22      24      28      0       0       0       0       0       0       0
24      46      250     23      25      0       0       0       0       0       0       0       0
25      129     250     24      17      26      0       0       0       0       0       0       0
26      129     294     25      27      34      0       0       0       0       0       0       0
27      70      294     28      26      0       0       0       0       0       0       0       0
28      20      294     29      27      23      0       0       0       0       0       0       0
29      20      309     28      35      30      0       0       0       0       0       0       0
30      20      340     29      53      31      0       0       0       0       0       0       0
31      47      340     30      52      32      0       0       0       0       0       0       0
32      86      340     31      51      33      0       0       0       0       0       0       0
33      130     340     32      34      50      0       0       0       0       0       0       0
34      130     310     35      26      0       0       0       0       0       0       0       0
35      68      310     29      34      0       0       0       0       0       0       0       0
36      181     340     33      49      37      0       0       0       0       0       0       0
37      193     340     36      38      0       0       0       0       0       0       0       0
38      222     340     37      48      39      0       0       0       0       0       0       0
39      299     340     38      47      40      0       0       0       0       0       0       0
40      368     340     39      46      41      0       0       0       0       0       0       0
41      409     340     40      45      42      0       0       0       0       0       0       0
42      429     340     41      43      44      0       0       0       0       0       0       0
43      451     340     42      84      77      0       0       0       0       0       0       0
44      429     442     42      45      0       0       0       0       0       0       0       0
45      410     442     46      44      41      0       0       0       0       0       0       0
46      369     442     47      45      40      0       0       0       0       0       0       0
47      299     442     48      46      39      0       0       0       0       0       0       0
48      221     442     49      47      38      0       0       0       0       0       0       0
49      179     442     50      48      36      0       0       0       0       0       0       0
50      136     442     51      49      33      0       0       0       0       0       0       0
51      87      442     52      50      32      0       0       0       0       0       0       0
```

5.3. Created Sample Database

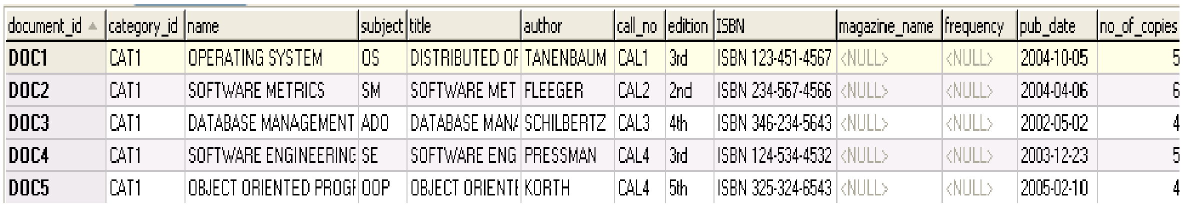
We created sample database to test the search engine component. This database is same as we mentioned in our solution section. This database is created in MYSQL below is the screen capture of sample database.

tblcategory:



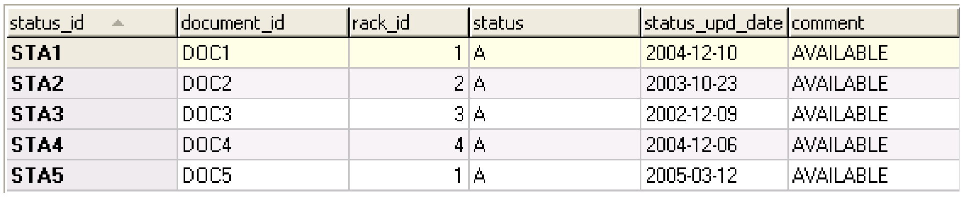
category_id	name
CAT1	Books
CAT2	Magazines
CAT3	Newspapers
CAT4	Journals
CAT5	Reports

tbldocument_detail



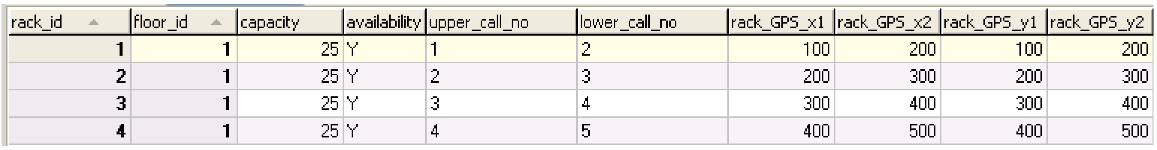
document_id	category_id	name	subject	title	author	call_no	edition	ISBN	magazine_name	frequency	pub_date	no_of_copies
DOC1	CAT1	OPERATING SYSTEM	OS	DISTRIBUTED OF	TANENBAUM	CAL1	3rd	ISBN 123-451-4567	<NULL>	<NULL>	2004-10-05	5
DOC2	CAT1	SOFTWARE METRICS	SM	SOFTWARE MET	FLEEGER	CAL2	2nd	ISBN 234-567-4566	<NULL>	<NULL>	2004-04-06	6
DOC3	CAT1	DATABASE MANAGEMENT	ADO	DATABASE MANA	SCHILBERTZ	CAL3	4th	ISBN 346-234-5643	<NULL>	<NULL>	2002-05-02	4
DOC4	CAT1	SOFTWARE ENGINEERING	SE	SOFTWARE ENG	PRESSMAN	CAL4	3rd	ISBN 124-534-4532	<NULL>	<NULL>	2003-12-23	5
DOC5	CAT1	OBJECT ORIENTED PROGI	OOP	OBJECT ORIENTE	KORTH	CAL4	5th	ISBN 325-324-6543	<NULL>	<NULL>	2005-02-10	4

tbldocumentstatus



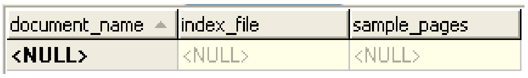
status_id	document_id	rack_id	status	status_upd_date	comment
STA1	DOC1	1	A	2004-12-10	AVAILABLE
STA2	DOC2	2	A	2003-10-23	AVAILABLE
STA3	DOC3	3	A	2002-12-09	AVAILABLE
STA4	DOC4	4	A	2004-12-06	AVAILABLE
STA5	DOC5	1	A	2005-03-12	AVAILABLE

tblrack_location



rack_id	floor_id	capacity	availability	upper_call_no	lower_call_no	rack_GPS_x1	rack_GPS_x2	rack_GPS_y1	rack_GPS_y2
1	1	25	Y	1	2	100	200	100	200
2	1	25	Y	2	3	200	300	200	300
3	1	25	Y	3	4	300	400	300	400
4	1	25	Y	4	5	400	500	400	500

tbl_amazondocdb



document_name	index_file	sample_pages
<NULL>	<NULL>	<NULL>

5.4. Setting up the environment for server

We installed Apache server on windows machine and PHP with GD library. This machine will use the web server. And also second machine with EKAHAU server, which will track the devices and give the location.

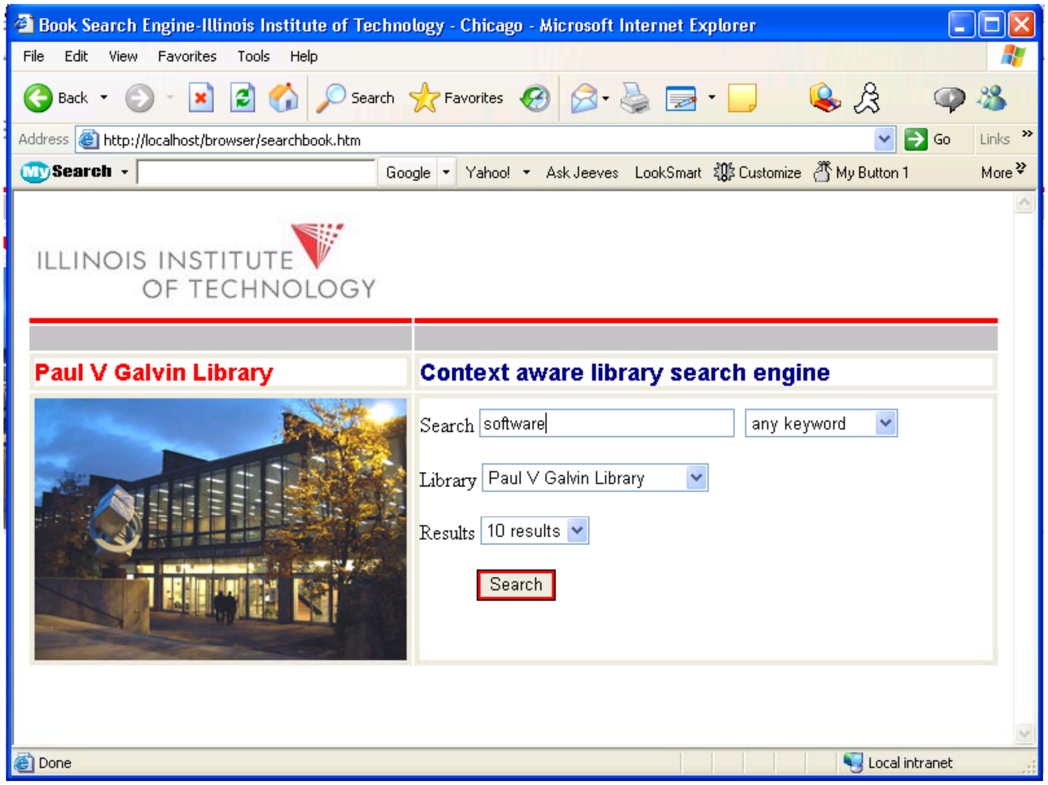
5.5. Implementing Path finder, Search Engine and Location Manager

We implemented above mentioned modules. The result of above application is as shown in below screen captures.

Screen1: In first screen user searches for the keyword "Software".

Screen 2: In Second screen Search engine has displayed the results matching with "software" keyword.

Screen3: In third screen User has shown the path of clicked book on library map



Book Search Engine-Illinois Institute of Technology - Chicago - Microsoft Internet Explorer

Address http://localhost/browser/searchbook.htm

Paul V Galvin Library

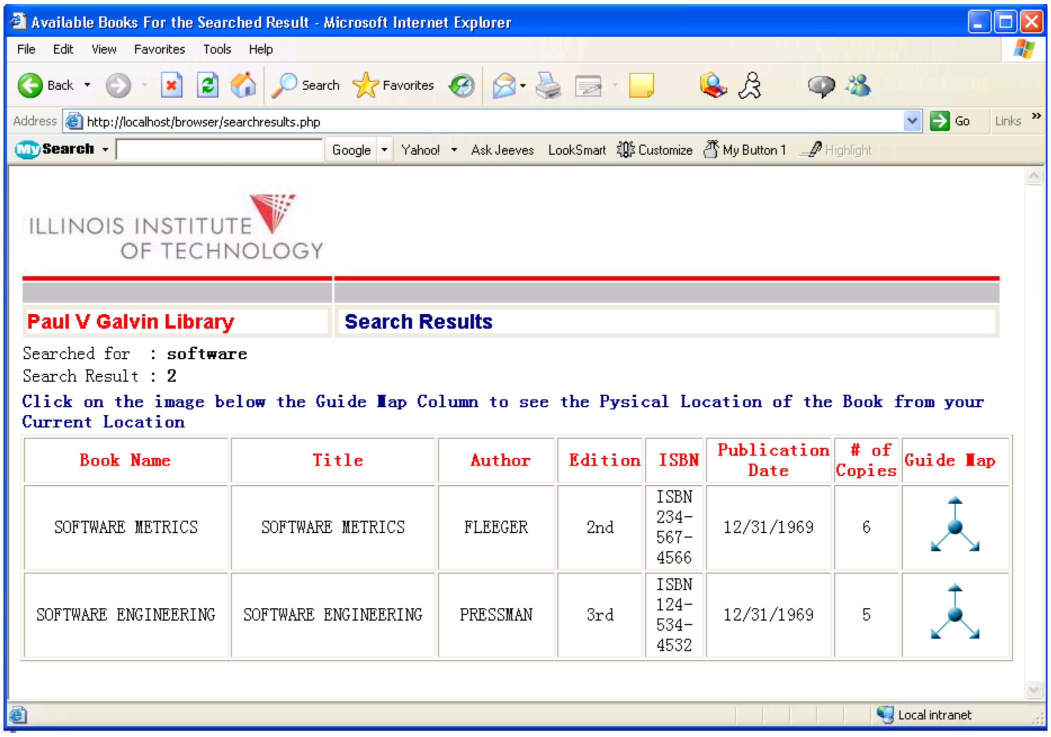
Context aware library search engine

Search software any keyword

Library Paul V Galvin Library

Results 10 results

Screen 1


Available Books For the Searched Result - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/browser/searchresults.php
Go Links
Search Google Yahoo! Ask Jeeves LookSmart Customize My Button 1 Highlight
ILLINOIS INSTITUTE OF TECHNOLOGY
Local intranet

Paul V Galvin Library

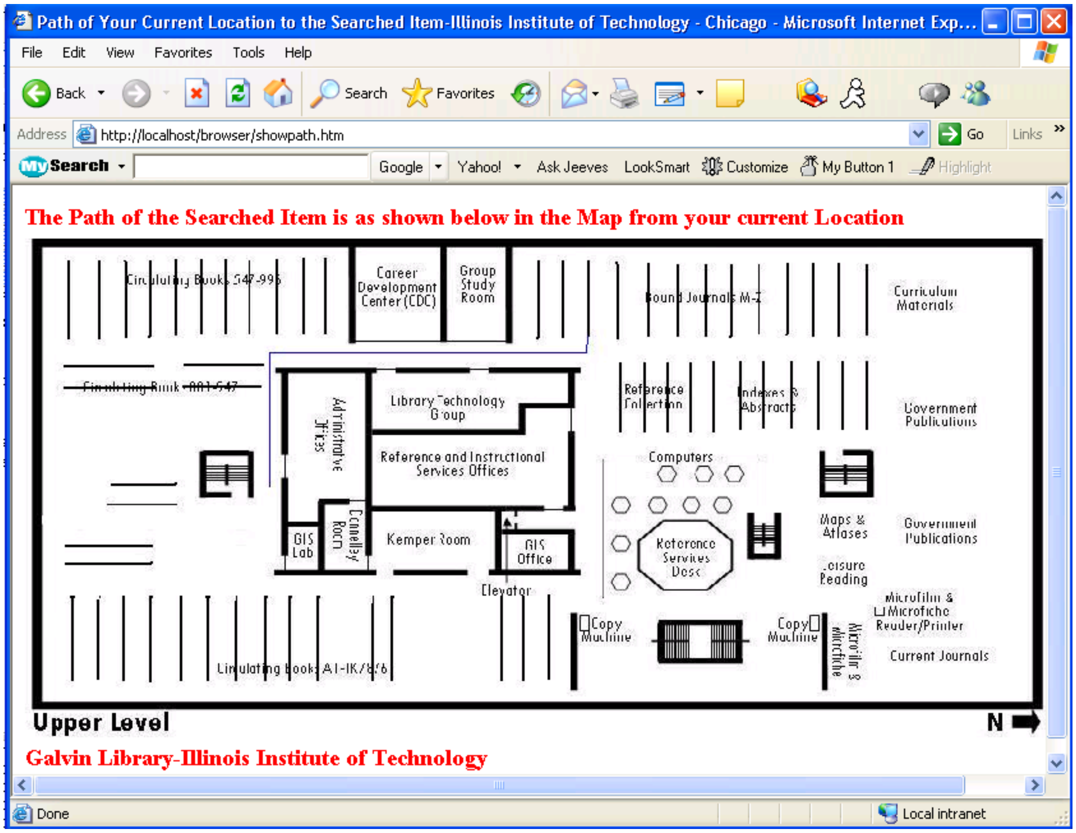
Search Results

Searched for : **software**
Search Result : **2**

Click on the image below the Guide Map Column to see the Pysical Location of the Book from your Current Location

Book Name	Title	Author	Edition	ISBN	Publication Date	# of Copies	Guide Map
SOFTWARE METRICS	SOFTWARE METRICS	FLEEGER	2nd	ISBN 234-567-4566	12/31/1969	6	
SOFTWARE ENGINEERING	SOFTWARE ENGINEERING	PRESSMAN	3rd	ISBN 124-534-4532	12/31/1969	5	

Screen 2



Path of Your Current Location to the Searched Item-Illinois Institute of Technology - Chicago - Microsoft Internet Exp...

Address http://localhost/browser/showpath.htm

The Path of the Searched Item is as shown below in the Map from your current Location

Galvin Library-Illinois Institute of Technology

Screen 3

6. Analysis and Conclusion

6.1 Analysis

Challenges:

1. Tool Challenge: We have created Search engine with PHP and Location Manager in JAVA. As we can not make JAVA and PHP bridge we have implemented Location Manager as web service.

Solution:

If we can make bridge between JAVA and PHP then there would not be need to create web-service. (By the way, creating web service has many other advantages).

2. Algorithm Challenge: we started to work on developing the algorithm on our own, but we found that it was not much reliable for the Path Finding.

Solution:

If we have implemented the well-known algorithm like Best First Search and A* algorithm then reliability would be more than current project.

3. EKAHAU Challenge: EKAHAU is over all good positioning engine, but sometime it fails to give very much accurate results. So the accuracy is some less by using it.

Solution:

We can make same positioning engine and instead of taking service from EKAHAU we would be having our own accurate Engine which has a good reliability.

Future Enhancement:

1. Instead of just showing path we can update the current position of the User while he is walking towards rack.
2. We will be having good algorithm which has more accurate result then current one.
3. We can wrap the Path finder and web service and make it usable by other campus's location aware project.
4. We can have a better UI of MAP on browser with zoom in and zoom out facility.
5. This library application is for one floor one building only. In future we can make it multiple building and multiple floors supported.

6.2 Conclusion

An ATLAS is pervasive computing application software, which is providing the Path Finding Facility to the Library Users in such a manner that they can see the location of the books, magazines, journals, etc…. This project has a good impact on starting use of pervasive computing application in library environment. This project has somewhat less accuracy in location and path finding aspect, but the approach of hitting the target was good.

7. Reference

1. M. Satyanarayanan:”Pervasive Computing: Vision and Challenges”, IEEE Personal Communications, August 2001

2. Nigel Davies, Keith Cheverst, Keith Mitchell and Adrian Friday: ”Caches in the Air: Disseminating Information in the Guide System”, Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99), New Orleans, Louisiana, U.S., 25-26 February 1999

3. Hawk tour – context aware tour guide – www.hawktour.net

4. Path finding algorithm - My campus project - http://www.cs.cmu.edu/~sadeh/mycampus.htm

5. PhP - http://www.php.net/

6. php class repository - http://www.phpclasses.org/

7. Install guide for installing apache and php- http://mpcon.org/apacheguide/
8. Tomcat - http://tomcat.apache.org/

9. Apache axis - http://ws.apache.org/axis/

10. Ekahau - http://www.ekahau.com/