# Mobile Services Exam Projects

F. Ricci

2008/2009

## Generalities

- The project should require more or less 40 hours
- The project must be presented at the last class if you want to pass at the summer session
- The project “deliverable” includes:
  - The code (one zip file): must be deployable in the wireless toolkit and must follow the standard structure of a WTK project
  - One text file (readme.txt) describing all the required steps to install and run the program on the WTK
  - One document (pdf file, less than 3000 words) describing:
    - The system functions: what are supposed to do
    - The human/computer interaction: how a typical interaction proceeds (use snapshots from the GUI)
    - The structure of the code (classes and files): describe the role of each class and the main methods
    - The major technical problems that have been tackled and how these have been solved.

## About the grade

- I will evaluate the following elements:
  - System functions coverage
  - Usability of the system (no errors, no strange commands, easy to learn, enjoy using the system)
  - Quality of the presentation
  - Quality of the report
  - Deployment (how fast I will deploy and run the system on my PC)
- The presentation must be contained in 15 mins and must illustrate the same points described in the report
- **If you reuse existing code you must clearly tell me what code you used and what have you done on top of that! If you fail to tell me that you will not pass.**

## Next steps

- Build a group of 2 people or work alone
- Select a project among those described here or write down a project description in a similar way
- Send an email to me (as soon as possible) with the following data
  - Group members: name and id number
  - Selected project: if it is not one of those suggested include the project description, or if you propose some changes (new functions) please describe in the email.

## P1 – Search Together a Restaurants

- **Functions**
  - Two or more user should be able to search a restaurant
  - Search restaurant by keywords or features
  - Browse detailed information (features, pictures, map, etc.)
  - Select a restaurant and make a proposal to your peers
  - They can reply with 'yes' or make a counter offer
  - Design a process that will "converge"
  - Store the selected restaurant (and reservation) in a local database (all the peers)
- **Implementation details**
  - Find a list of restaurants from the web or build your own sample db
  - Communicate with your peers with SMS (or sockets, if you have wifi)

## P2 - News

- **Functions**
  - Download a list of News from a RSS feed (e.g. BBC http://newsrss.bbc.co.uk/rss/newsonline_uk_edition/technology/rss.xml (better: a source of local news for Bolzano)
  - Search news by keywords in the description
  - Browse the result list
  - Browse detailed information about the news
  - Select a news and store it in a local store (organize the news in categories)
  - Implement a function to browse the saved news
- **Implementation details**
  - Use an existing RSS feed
  - Parse the XML using an existing library like kXML
  - Exploit the RecordStore library

## P3 – Museum/Building guide

- **Functions**
  - Show the building map with icons for landmarks (rooms, exhibits, etc.), toilet, café, exit
  - Your position is shown on the map and changes as you move
  - When you come close to a landmark it shows the detailed info (picture and text) about the landmark
  - You can decide to store some data (about the landmark) in a local notes’ file, or recordstore
  - At the end of the visit you should be able to browse a summary of your visit (log)
- **Implementation Details**
  - Base the implementation starting from the WTK example CityGuide (Location API)
  - Position is simulated using the WTK external event generator.

## P4 - Transportation

- **Functions**
  - Insert your current position, the destination and eventually the time (if the current time must not be used)
  - Get the timetable of the bus going to destination starting from your position
- **Implementation**
  - Position can be
    - inserted by the user with a code
    - or using the location API
    - or from a list of bus stops (the last 5 inserted)
  - The timetables are stored locally
  - Connect to a server to download updated timetables.

## P5- Expense Management

- **Functions**
  - Keep record of your expenses
    - Date: the date of the expense
    - Description: short description
    - Amount: the amount you paid
    - Payment method: the payment method used to pay
    - Category: the category of the expense
  - Generate expense report in CSV format
  - View, sort and search
  - Synchronize the expenses with a server “database” (keep it simple this!)
- **Implementation**
  - Use the MIDP record store.

## P6 - Task List

- **Functions**
  - Add, delete, modify a task (imagine a Field Force Automation situation)
  - Add, delete, modify a client
  - Add, delete, modify a category
  - View tasks filtering by date, client, category
  - Export task list in CSV or XML
  - Synchronize tasks and clients with the server
- **Implementation**
  - Use the MIDP record Store
  - Connect to a server for uploading the generated tasks and clients.

## P7 - Care Giver

- **Functions**
  - Service Request (an old person in need of assistance)
    - User sends help request by pressing a Key
    - She gets a response from the care giver (read and confirm receipt)
  - Receive directions from care giver
    - User receives an alert message (e.g. “take the pills”)
    - The alert is repeated if no confirmation is given
    - User acknowledges the receipt and the acceptance to the caller
    - After a timeout the caller is warned that the receiver has not acknowledge the request
- **Implementation**
  - Use SMS for exchanging information between the care giver and the user
  - The MIDlet must listening for incoming SMS and make the interaction simple and easy to manage.

## P8 - Tracker

- **Functions**
  - Record the position and notes (pictures) during a walk (time interval or space interval)
  - Save the collection of the positions and notes with a name
  - Load a collection from past saved collections
  - Export a collection in Garmin or Google Earth XML format (http://earth.google.com/kml/)
- **Implementation**
  - Position can be obtained using the location API
  - The collections are stored locally
  - Implement a server for uploading the files

## P9 – Domotics (Home Automation)

- **Functions**
  - Control the state of your home (raise the temperature, close the windows, start up the oven, ...)
  - Receive an alarm from your home (intrusion detection, fire detection, etc.)
  - Log the actions and the alarms in a record store and browse it
- **Implementation**
  - Build a nice GUI for this application (simple!)
  - Use SMS or sockets connections with push
  - Simulates the events in the house and the actuation of commands
  - In the house there is another "phone" that communicate with your mobile.