TECHNISCHE UNIVERSITÄT DRESDEN

Department of Computer Science Institute for System Architecture, Chair for Computer Networks

Application Development for Mobile and Ubiquitous Computing

Seminar Task

Adaptation Concept Presentation

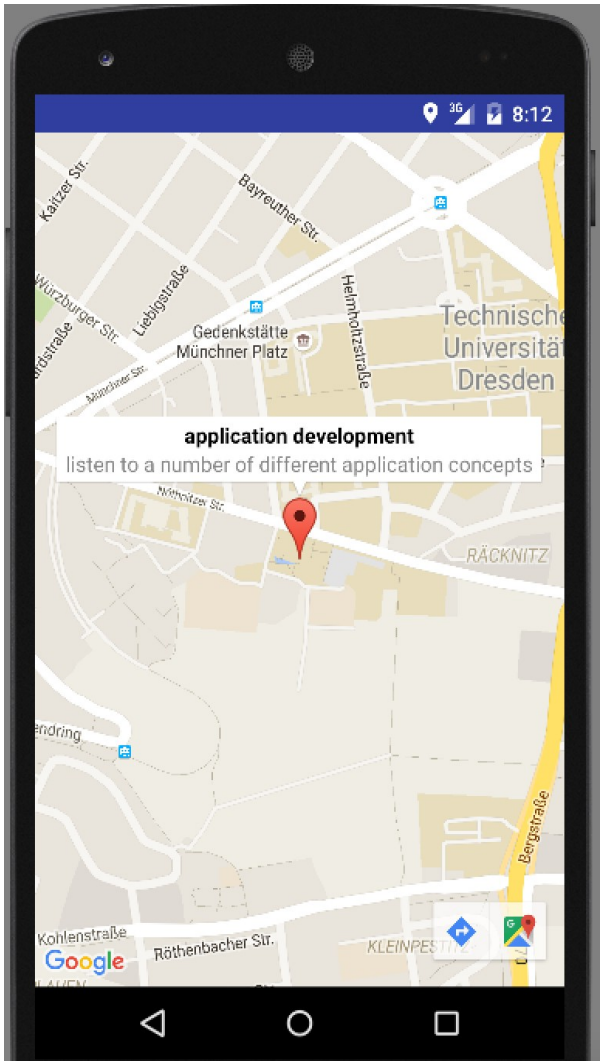
GroupNo. 6

Team: Nicolas Fricke, Johannes Maresch

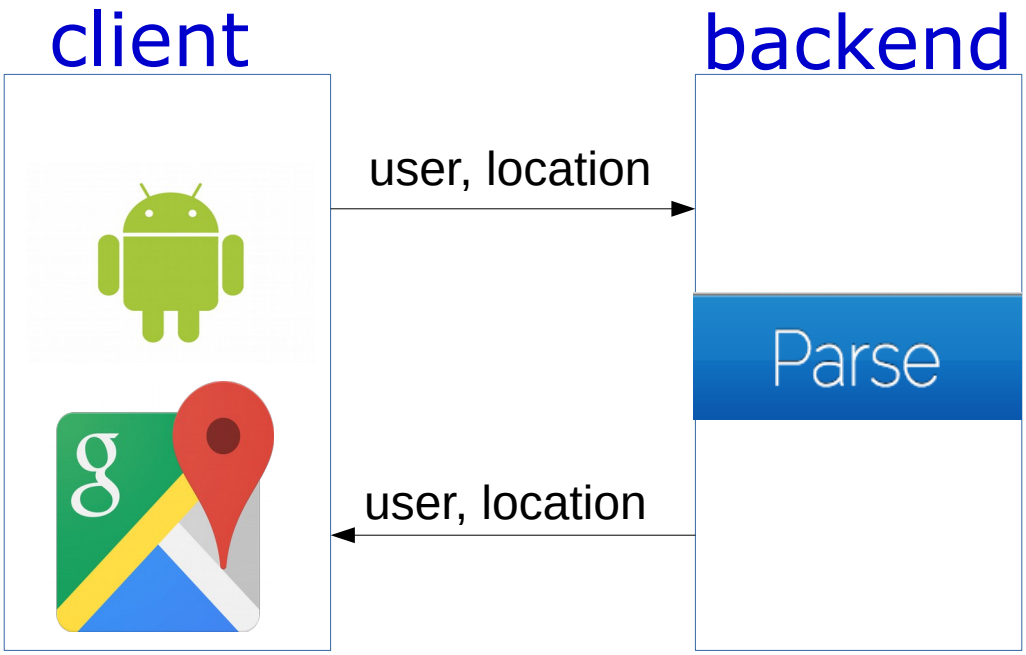
Application Scenario

Flashmeet

- an application for meeting new people, finding new interests and exploring the city
- get to know the culture around you apart from books and guides
- get spontanious inspirations for your day



Architecture and Technologies



Challenges

- **Energy Challenge**
  - usage of gps for location determination
  - high battery drain
  - higher intervals of location requesting
  - energy aware communication
- **Connectivity Challenge**
  - location based retrieval of information
  - different network types/ limited mobile data usage
  - reduction of data size
- **Offline Challenge**
  - constant updates of events
  - unforeseeable network problems
  - local storage and usage of last synchronized information stack

Adaptation and Context

- battery status and network connection
- BatteryManager, TelephonyManager

- **Energy Challenge**
  - gps as positioning service
  - get battery status with BatteryManager class
  - check battery status after every location update/gps activation
  - change update interval based on return value

```
public static int getBatteryLevel(Context context)
{
    Intent batteryIntent = context.registerReceiver(null, new IntentFilter(Intent.ACTION_BATTERY_CHANGED));
    int level = batteryIntent.getIntExtra(BatteryManager.EXTRA_LEVEL, -1);
    int scale = batteryIntent.getIntExtra(BatteryManager.EXTRA_SCALE, -1);
    if(level == -1 || scale == -1) {
        return (int)(50.0f);
    }
    return (int)(((float)level / (float)scale) * 100.0f);
}
```

Energy Challenge

- we use the method getBatteryLevel() to get the current battery state in percent
- our gps update mechanism redetermines the current position after a short interval of 5 seconds
- the current position in latitude and longitude is needed to filter the events (explained in connectivity challenge)
- once the battery status is below 15% we increase the interval to 60 seconds to save the user energy for longer usage

Adaptation and Context

- **Connectivity Challenge**
  - fixed range query based on current location
  - check connection type with TelephonyManager class
  - network change listener receives network status
  - method changes data update size based on context information (wifi/any mobile data/ [different network types])
  - filtering → mobile data = less data quantity(reduce area size)

- **Offline Challenge**
  - synchronizing after every location update
  - caching(parse local data storage)
  - last image gets displayed

```
NetworkInfo info = myCheck.getNetworkInfo(context);
if(info != null && info.isConnected())
{
    if(info.getType() == ConnectivityManager.TYPE_WIFI) return 1;
    if(info.getType() == ConnectivityManager.TYPE_MOBILE) return 2;
}
```

Connectivity Challenge

- method to return a value based on network type
- based on the value we alter the range query size
- value 1 for wifi so we can get data in a bigger range query with a range of 5km since we have no limited traffic
- when the method detects a mobile network type we return the value 2 and alter the range query to a small 500m to save mobile traffic
- this way we effectively filter data by reducing the size we download from the backend

Offline Challenge

- the user can create events that get uploaded to the server and download events made by other from the server
- if the user wants to create an event he has to be online otherwise he gets an error message
- we use the isConnected() method to check whether he is online
- if a user wants to synchronize to receive events near him we use isConnected() to determine whether he is able to download or he gets the data from cache
- the user can only add data to the server and download it but can't update it therefore no conflicts should occur

Work plan

Second Phase(mostly finished)

- connect isolated functions with the main code
- improve the basic ui of the map layout

Third Phase

- Debugging & Testing
- continue implementation cycle with non-core features