Severenity

Design Doc

# Energy recovering

## General overview

User’s actions such as: Attack, Capture, Defense, Remove owner, etc. decrease the energy. In order to recovery energy user should pass e.g. 10 meters, or wait one day to have 1 point of energy to be recovered.

## Technical overview

The recovery request is initiated by the app after user the passes 100 miters or after one day passed. On the app the information about last recovery request is stored into DB. The special table has been created for that purposes. The table has only one row which has next info: time when last update happened; distance in meters that user has passed when last update happened.

Application has timer logic that invokes every 5 minutes and verifies next things:

1. Current distance user passed minus distance that is stored in database. If the distance between this two values is bigger than 100 meters application will send RECOVERY request to the server.
2. Current time minus time that is stored in the database. If the result value will be bigger than one day then application will send RECOVERY request to the server.

The application doesn’t increase the energy by its self. This is server’s responsibility to increase the amount of the energy and notify user about that. **TBD: Right now the server will notify application with “update user info” notification, so application will get latest user’s info from the server.**

## Technical solution

### Application-server communication

Communication between the application and the server is happening by sockets. To notify server about the RECOVERY request was used:

***SOCKET\_EVENT\_ACTION\_ON\_USER*** = **"user action"**;

With next JSON structure:

***{***

***"userID":"",***

***"action": ""***

***}***

The ***userID*** should be ignore for this action call. The action should be: RECOVERY. Should be taken from:

**public enum** UsersActions {  
 ***CAPTURE***(**"capture"**), *// user captures the building* ***ATTACK***(**"attack"**), *// user attacks other user* ***REMOVE***(**"remove"**), *// user removes other user from owning place* ***ATTACKED***(**"attacked"**), *// user was attached by other user* ***RECOVERY***(**"recovery"**); *// recovery user energy due to time/distance passed* ...  
 }  
}

### Application overview

Application has EnergyRecoveryManager object that is responsible for tracking and sending RECOVERY requests to the server. It uses java.util.Timer for scheduling verifications and java.util.TimerTask for having background process where this verification will be performed. After certain amount of time the TimerTask is being executed. It reads data from local database about the last recovery event and verifies if conditions are met to request recovery. (Please see 1.3. for more info).

### Database overview

For storing information about recovery request new DB table was introduced:

|  |  |
| --- | --- |
| recovery | |
| **timestamp** | String |
| **distance** | Integer |

**Timestamp –** identifies the time in "***yyyy-MM-dd'T'HH:mm:ss.SSS'Z'***" format when the last RECOVERY request was send.

**Distance –** identifies the distance that users has passed when last RECOVERY request was send.