|  |  |
| --- | --- |
| Category | Details |
| Program Organization | * Use the app to call a person and have that person’s phone volume be maximized * Grant permission and keep track of people allowed to use this feature for user |
| Major Classes | * PermittedMembers * PermittingMembers * AddNewCaller * RequestPermission |
| Data Design | * Lis of people allowed to call user * List of people that user can call * Ringtone object for calls received from app |
| Business Rules | * Use as little communication resources as possible |
| User Interface Design | * Home: List of Users that can be called * New Request: Name and Number * New Permission: Name and Number |
| Resource Management | * Resources shouldn’t be too scarce or stressed to be an issue |
| Security | * Users must approve people trying to call them * Permission done on the receiving end so the most recent removal of approved callers is accounted for |
| Performance | * Little computation needed |
| Scalability | * If database is used, a version of parse or amazon can be considered * Initial design does not call for central database (phone to phone communication only) |
| Interoperability | * Send information to other phones * Use data to send info phone to phone * Prototyping required to determine whether central database needed or whether direct communication possible via phones |
| Internationalization /  Localization | * No current effort to expand |
| Input/Output | Input:   * Touch screen * Back key   Output:   * Phone call * Screen display |
| Error Processing | * Crash Program on error |
| Fault Tolerance | * No current plans to deal with fault tolerance |
| Architectural Feasibility | * May require reliance on data * Request to be able to make calls without data but that may be a requirement * Phone to Phone communication needs to be researched |
| Overengineering |  |
| Buy vs. Build | Buy:   * Buy database service if necessary   Build:   * Android development * Eclipse |
| Change Strategy |  |

Architectural Feature Not Being Considered:

1. Overengineering
2. Change Strategy