1 Reference Architecture and frameworks

In this section we will discuss the reference architecture and frameworks that will be used in the P.A.P.E.R system.

1.1 An object-relational mapper

Object-relational mapper is a technique of converting data between incompatible systems by means of object orientated programming. For example the conversion of database attributes into a database object. This allow the interacting object-orientated language to manipulate this database object with ease, in essence the data can be presented in a way that any object is presented in that programming language.

Django uses Object-relational mapping(ORM) with regards to its Model-View-Controller (MVC) model. The data models and a relational database (Model) are manipulated with the ORM strategy to interact with the databases.

1.1.1 Advantages

- The advantage of using object-relational mappers with databases in particular is that joins aren't used that often as object types can be followed by means of referencing pointers.
- Relationships are also established by means of pointers which can increase efficiency for complex data.
- This approach works well for large amounts of data, as the object can be manipulated easily for each field.

1.1.2 Disadvantages

- Inefficient when used with small databases as objects will still be created which might be less efficient than a quick lookup of those particular fields.

1.2 Application Server

Software framework for web applications and a server to run the environment, is the principle that Application Server approach follows. An example of Application Server architecture framework is the Java EE framework. This architecture is based on the Layer model and Client-Server model and contains a service layer which is accessed by the developer. Django supports Java EE based Application Server.

1.2.1 Advantages

- Scalability
  - Resources are allocated efficiently
  - Objects reuse is ensured
- Integrability
  - Integrates well with REST
  - Database integration is provided
- Security
  - Authentication and confidentiality is supported

2 Reference Frameworks

2.1 Django

Django is a web framework, written in Python which uses the Model-View-Controller architectural pattern, or MVC for short. Django's main aim is to provide a framework on which to build websites that are primarily based on complex databases. The use of Object-relational mapper in Django's MVC is described in 1.1. Django processes HTTP requests with a web templating system and regular-expression URL control (View and Controller).

Django further includes:

- Its own web server for developmental purposes.
- Form validation and storing of form data in database
- Caching framework with several cache methods
- Serialization system to produce and interpret XML and JSON representation of model instances.
- Python unit test framework

Django database support:

- PostgreSQL
- MySQL
- SQLite
- Oracle
- Microsoft SQL Server (through django-mssql)

Django can be used with:

- Python (Supported by default)
- JavaScript (through Swig)
- Ruby (through Liquid)
- Perl (through Template::Swig)
- PHP (Twig)

We prefer to use Django as it does have a slight learner curve, but will ease integrability with regards to our Android application.

2.2 Honarable Mentions

2.2.1 AngularJS

Web-framework making use of client-side ModelViewController model. AngularJS makes use of the MEAN stack for its front-end, consisting of MongoDB database, Express.js web application server framework, Angular.js itself, and Node.js runtime environment. We prefer to use a server side approach as this eases integrability and simplifies implementation.

2.2.2 Ruby on Rails

Ruby based web-framework based on Ruby programming language. "Rails" uses a Model-View-Controller based model and emphasizes the use of JSON and XML for data transfer. Since Ruby has a high learning curve compared to Python for example, we prefer to use Django instead.