**GOOGLE APP ENGINE**

**INTRODUCTION**

Google App Engine is a platform as a service(Paas)offering that lets you build and run applications on google’s infrastructure. App engine applications are easy to build,easy to maintain and easy to scale as our traffic and data storage needs change. With App Engine,there are no servers for us to maintain.We simply upload our application and its ready to go.

**FEATURES OF GOOGLE APP ENGINE**

The Google app Engine can be divided into three parts:

* Runtime Environment: Itself presents as the place where the actual application is executed.
* Persistence and the datastore: The stateless execution of applications creates the need for a datastore that provides a proper way for persistence.
* Services: Abstraction are implemented as a services that can be directly called from the actual application.

GOOGLE APP ENGINE

ARCHITECTURE:

The google app engine is a google’s answer to the ongoing trend of cloud computing offerings within the industry.In the traditional sense,GAE is a web application hosting service,allowing for development and deployment of web-based applications within a pre-defined runtime environment.Unlike other cloud-based hosting offerings such as Amazon Web services that operate on an IaaS level,the GAE already provides an application infrastructure on the Paas level. This means that the GAE abstracts from the underlying hardware and operating system layers by providing the hosted application with a set of application-oriented services. While this approach is very convenient for developers of such applications, the rationale behind the GAE is its focus on scalability and usage-based infrastructure as well as payment.

Cost:

Developing and deploying applications for the GAE is generally free of charge but restricted to a certain amount of traffic generated by the deployed application. Once this limit is reached within a certain time period, the application stops working. However, this limit can be waived when switching to a billable quota where the developer can enter a maximum budget that can be spent on an application per day. Depending on the traffic, once the free quota is reached the application will continue to work until the maximum budget for this day is reached.

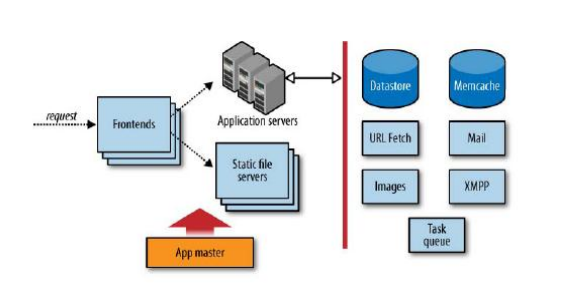


Fig:Structure of Google App Engine

APPLICATION DEVELOPMENT USING GOOGLE APP ENGINE

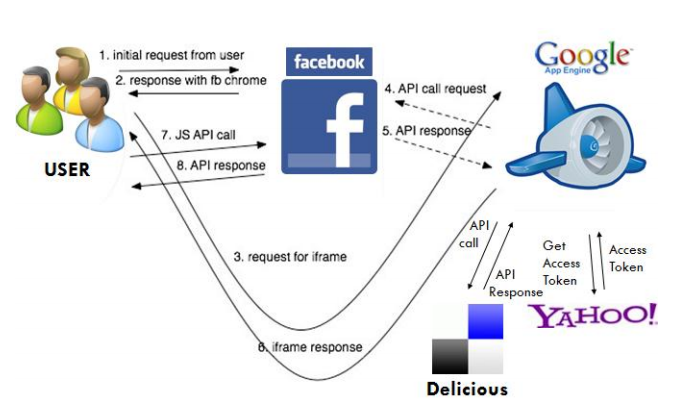
* General idea:

Inorder to evaluate the flexibility and scalability of the GAE we tried to come up with an application that relies heavily on scalability.

* Requirements and functionality
* Implementation:
  + Development Environment: Because of our expertise in Java and familiarity with the Eclipse IDE, we decided to use both for developing our application. Furthermore, by using Eclipse we were able to use the GAE plugin that improves the development and debugging of a local GAE application significantly. Also, we agreed on using the Google Web Toolkit (GWT), a Java framework that helps developing rich web applications with AJAX-based user interfaces.
  + Application Environment: As stated in the requirements, the application should run embedded within the Facebook website.As in fig. user can find and select the application via Facebook and send a request to start the application within Facebook. This will trigger a request for an iFrame that is forwarded to the GAE where the actual application is started. The GAE will then call the Facebook API and wait for a response.As soon as the response is received by the GAE, it returns the iFrame with the application to the user. The application is then visible within an iFrame in the Facebook UI and is ready to be used. If the user is not yet logged in to delicious he can now do so. The login to delicious is done via Yahoo authentication. So the GAE sends a request for authentication to Yahoo’s authentication servers and receives an access token. With this access token our application running within the GAE can then access the user`s bookmarks by requesting them from the delicious servers.

Application Architecture:

LAYER & COMPONENT OVERVIEW

Our application is based on a Three-Tier client/server architecture incorporating a presentation, an application/business logic and a data tier. Figure 7 shows an overview of the components and layers involved. The presentation layer is represented by the component web browser.Based on the GWT, Java code is automatically compiled into AJAX code running withi the client`s web browser.

Presentation Layer:

The presentation layer is distributed over several sub-packages, in order to fulfill the division between the parts of the presentation layer that are fully server- and those that are solely client-based.

Data Layer:

In order to persist data within the GAE, we are using JDO classes. Although Google’s BigTable is by definition a schema-less database, the JDO classes serve as a means to define the schema for kinds of entities.

PROPOSED SYSTEM

* Easy to build,maintain and easy to scale
* No need of server to maintain
* It can be used to build mobile and social applications,traditional web browser-based applications,standalone web services or offline batch processing.
* GAE services enables users to build and host web apps on the same computing infrastructure that powers Google applications.

ADVANTAGES FOR THE SOFTWARE DEVELOPER:

* Service-oriented distributed computing will emerge as a factor in shaping the industry.
* Changing the way business is conducted and how services are delivered and managed.
* Cloud computing opens up exciting new possibilities based on a mix of old and new technologies for the next generation of software developers.

SOFTWARE ENGINEERING ASPECTS

Functional aspects:

* PaaS environments set certain restrictions to the developer in terms of programming techniques,languages and other elsewhere available functionality.

Usability:

* GAE offers easy access to the domain of Cloud Computing by providing abstractions for important services such as persistence as well as an easy to use development environment.

Scalability:

* Instead of running an application on more powerful hardware,the application is executed on more instances of less powerful hardware.

Integration:

* It will bring a new form of cloud service,that is cloud integration service.

Availability:

* A high availability architecture is adopted and both the platform and applications are fully tested.

Support:

* Cloud services should be designed for easier usability.

Privacy:

* Current privacy concepts such as the fair information principles may face damage to their reputation.

Tips of SE:

1. Minimize personal information sent to and stored in the cloud

2. Protect personal information in the cloud

3. Maximize user control

4. Allow user choice

Privacy should be built into every stage of the product development process: it is not adequate to try to bolt on privacy at a late stage in the design process.

User Authentication:

For user authentication the GAE comes along with the Google authentication service. This enables developers to easily integrate logins for Google accounts into their applications. Given the acceptance of Google accounts, this feature is really useful and a great advantage compared to IaaS solutions where authentication has to be handled by the developer on its own.

Legal Issues and Compliance:

Enterprise users must maintain business legal documents and assure their integrity in order to comply with various laws. Cloud computing vendors have to adopt technologies to ensure that their enterprise users’ data satisfy their compliance requirements. Again, this does not seem to have received much attention yet .At this stage the GAE does not make any specifications about legal issues and compliance. For applications that are heavily dependent on such restrictions, the GAE might not be the right choice at this point in time. But once again, Google has to look into these issues when becoming a serious PaaS provider on the enterprise level as its start of the GAE for Business suggests.

Cost:

The 3rd party provider owns and manages all the computing resources (servers, software, storage and networking) and electricity needed for the services. The users only need to “plug into” the cloud. The users do not need to make a large upfront investment on computing resources; the space needed to house them; electricity needed to run the computing resources; and the cost of maintaining staff for administering the system, network, and database. In terms of costs, the GAE offers a usage-dependent pricing scheme that starts with a basic version which is free of charge but subject to certain limitations. The paid version of the GAE removes some of these limitations, however other limitations still exist caused by the GAE’s design. As no upfront investments are necessary, the GAE is a good way to test new applications (even for free) and pay as the acceptance and spread of the application grows. Interestingly, the recently launched GAE for Business goes into a different direction, as Google now offers a flat-rate pricing scheme for enterprise users.

**CONCLUSION**

Cloud Computing remains the number one hype topic within the IT industry at present. Our evaluation of the Google App Engine has shown both functionality and limitations of the platform. Developing and deploying an application within the GAE is in fact quite easy and in a way shows the progress that software development and deployment has made. Within our application we were able to use the abstractions provided by the GAE without problems, although the concept of Bigtable requires a big change in mindset when developing. Our scalability testing showed the limitations of the GAE at this point in time. Although being an extremely helpful feature and a great USP for the GAE, the built-in scalability of the GAE suffers from both purposely-set as well as technical restrictions at the moment. Coming back to our motivation of evaluating the GAE in terms of its sufficiency for serious large-scale applications in a professional environment, we have to conclude that the GAE not (yet) fulfills business needs for enterprise applications at present. As the discussion showed, some of these needs are yet to be satisfied by Cloud Computing platforms in general, others are GAE-specific issues. However, seeing the benefits and potential of PaaS-based approaches such as the GAE, the question remains whether quite inflexible and non-standardized PaaS platforms can establish themselves in the market for serious large-scale applications or will remain platforms for small and simple applications as seen today.