International Journal of Computer Science
and Engineering (IJCSE)
ISSN(P): 2278-9960; ISSN(E): 2278-9979
Vol. 4, Issue 6, Oct - Nov 2015, 1-4

International Academy of Science,
Engineering and Technology
Connecting Researchers; Nurturing Innovations

DASHBOARD ARCHITECTURE FOR WEB-BASED APPLICATIONS

SAGAR GUPTA

Senior Software Developer, Amadeus Labs, Bangalore, Karnataka, India

ABSTRACT

In the fast improving technological world, an organization depend son updates from various business bodies. Some of these updates are so crucial and vital for the organization that it serves as a tool for business owners to predict the future growth or keep the business operational round the clock.

This paper discusses about collaborating these updates from various operational and functional business entities, on a single platform which can be used for business analysis.

KEYWORDS: Dashboard Architecture, Web-Applications, Widgets, Widget Design

INTRODUCTION

Business processes where data collaboration is required to keep the business operational, data analysis playsa vital part in designing business operation strategy. This information sometimes comes from similar business entities or from entities which deals with different segments of the business model. To organize this data in collaborative manner for management or operational purpose in the organization is one of the biggest tasks in current IT world as the information flown into the collaborative system might be structured or unstructured. The information may also vary based on the domain from which updates are fed to the system like updates from the finance department could be different in terms of frequency and structure when compared to the information coming from sales department.

To resolve such issues where structuring and representation of data onto a common platform is key to keep the business functional can be resolved by adopting ***Dashboard Architecture.***

DASHBOARD ARCHITECTURE

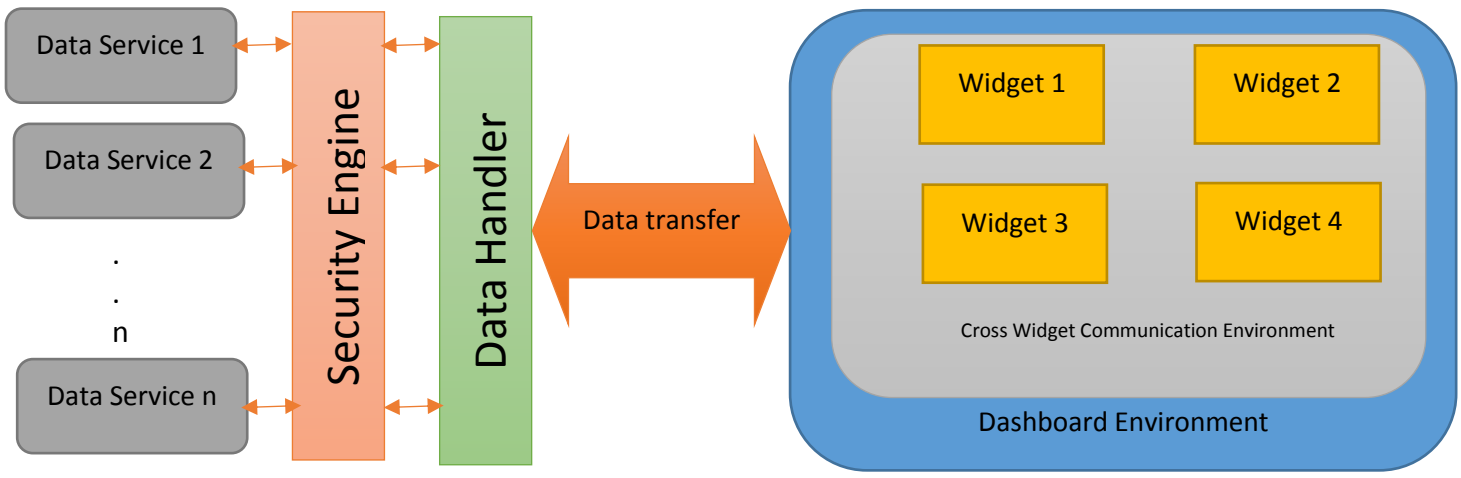
Dashboard architecture facilitates in creating a common platform for heterogeneous systems to be hosted on a single collaborative environment where all different business entities send the data in standardized structure and this data is used to update corresponding ***Widget.***

Widget: It's a small application which has a scope limited to specific business task or operation. A widget plays a role of transient or auxiliary application, where the first type serves a role of information gatherer for business operations and later serves as port let or web-part to collect information from the end-user for the business operations.

Widgets are created for business entities to facilitate information gathering for functional or operational needs. Moreover widgets serves as tool to showcase statistics based on data coming from various sources on a collaborative platform to make the operations more efficient. A typical widget dashboard environment comprises of following components:

- **Dashboard Environment:** Platform to host applications developed for various business units in a collaborative form. The environment provides UI interface to the end-user for creating or updating the dashboard for business needs.
- **Widgets:** Port lets or applications which are running in dashboard environment to cater the needs of one or many business functionalities. These widgets are ideally designed by adopting common UI model which enables hosting and maintenance of this widget generalized across business operations.
- **Data Handlers**: Services which act as data processing units for the environment where data is processed based on the inputs from the end user or act as data collaborators to project reports into the widget used for business analysis.
- **Communication Handlers:** A widget may depend on data fed directly to it or on the data coming from other widget or from other service. In that case cross widget communication plays important role in transferring data from one widget to other. As these data sources might be different, data security plays crucial factor for these type of cross widget communications. Cross domain widget communication may prone to data integrity issues hence data transfer protocols plays vital role while designing communication handlers.
- **Security Engine:** The whole idea of collaborative platform for widget hosting is based on security platform underneath. Security engine acts as backbone for such platforms as security engine handles access privileges for whole system users without which data handling and data transfer may result into security loopholes. Dashboard environment should be powered with single sign on policy, which means all the widgets which are available on the dashboard should use a common policy for user authentication. *Access Token* mechanism or *Passport Authentication* mechanism can be used for such security engine design.

ARCHITECTURE DIAGRAM



DISCUSSIONS

To create dashboard environment for business model, any lightweight and performance incentive technology can be used such as JAVA, HTML5, and ASP.NET etc. For creating such setup modularization and structural segregation of various business entities is required. The dashboard environment should provide following for widget development

framework:

- A common deployment strategy across domains.
- APIs for cross widget data communication.
- APIs for UI customization and a standard UI template for cross browser support.

The widgets which are deployed on the dashboard environment should have common data transfer and security authorization protocol such that data security and data handling couldn't open loopholes in the environment. Widgets can be designed by using any server side framework which is supported by the hosting platform i.e. dashboard should provide support for such framework for widget development.

Data handler engine plays an important role in this environment as the data produced or consumed by the widgets should adhere to common data model to solve communication related issues, and can easily be used by cross widget communication modules.

Security Engine which provides implementation for the authorization related tasks can be implemented using following mechanisms:

- **Access Token Mechanism:** In this mechanism when the end user logs into the dashboard system using access credentials a security token gets generated from the security engine and this access token is passed to all the data services to either fetch data from other resources or to provide data to the various business entities. To maintain security in the environment these tokens should use 2-way authentication model to maintain user's access related privileges for the data services
- **Passport Authentication Mechanism:** Passport authentication is based on a authentication done by particular website which issues passport to the user based on the user credentials such as hotmail, devhood etc. This passport once obtained by the security engine will be used for authorizing the user into the environment and for various data services invocation for widgets.

BENEFITS

- This architecture gives organizations a collaborative platform to handle information coming from various entities which are responsible for different business operations.
- Each widget can provide functionality to the end user to maintain business entity in efficient way as data flowing in-out from the system can be used and shared among widgets.
- Analytical/Statistical reports can be generated for various business entities using dashboard platform.
- Cross widget communication provides vital mechanism to conceptualize future growth of the business through data analytics of business entities hosted on this system.

CONCLUSIONS

Dashboard architecture implementation provides capability to business owners to come up with a common platform for all business entities and stake holders for data exchange and data collaboration. It also provides implementation of single

sign on policy which helps the businesses to maintain the authorization related implementations and gives data service provider common mechanism for handling access for all users.

REFERENCES

1. "Web Application Working Group's Widgets: Family of Specifications". W3C.
2. Rajesh Lal: Developing Web Widget with HTML, CSS, JSON and AJAX (ISBN 9781450502283).
3. MSDN: Passport Authentication provider.