Business Systems Integration

Trident University

Jeff McCoy

Module 1 Session Long Project

CSC425: BSCS Integrated Project

Dr. Ying Liu

1 April 2014

The business integration project I have chosen is a training requirements and scheduling web application. The application will provide tracking of training needs for 40,000 individuals a year across the United Stated Air Force and handle all scheduling and instructor utilization tracking. This paper will define the scope of the problem and planned solution to meet the business needs of the customer.

The Problem Defined

Existing System

This application known as Field Training Support System (FTSS) will solve three distinct problems once completed: identification of student mandatory training requirements, advertising and scheduling of unused training seats and reporting of instructor utilization. These disparate processes each contain closely related, and often overlapping data that would be ideally suited for one system. The follow paragraphs discuss each of those problems in more detail.

The identification of training requirements is currently a paper-based system known as an 898 requirement that has been in place since 1987. Within this system, a unit identifies a number of training seats required per course, per location each month. The training unit then has five days to respond and determine if they can meet the training need. This process establishes a contract between the two military organizations and certain courses are mandated and therefore must be prioritized during the requirements process. Because only numbers are given to the training unit, this process often is convoluted and error-prone with no recourse for the training unit. Numbers can easily be misrepresented or miscalculated by the unit requiring the training. These procedures are outlined within Air Force Instruction (AFI) 36-2232, p. 36.

The scheduling process is currently a Microsoft SharePoint solution that is prone to errors, data corruption, significant performance degradation and lacks any single source of truth for data as defined by Burkhow, 2011. This system leverages complex SharePoint Collaborative Application Markup Language (CAML) to generate reports for users to act upon; these reports cause a performance bottleneck that at peak times leads to server instability (Microsoft, 2010). Additionally, data corruption often occurs as SharePoint workflows are utilized that fail the ACID test: Atomicity, Consistency, Isolation, Durability as defined by Gray, 1981. These workflows often require copying data to other lists or notifying users through email of an action required. When they fail, they are simply queued indefinitely leaving the data in an unknown state and no longer providing a single source of truth for the state of data (Sampson, 2012).

Finally, instructor utilization is currently a manual process of data aggregation through individual Excel spreadsheets. In this process, each instructor has a tab in a spreadsheet that is cloned each month, they are required to annotate their teaching hours, number of students and other data already in several other systems as well as any hours not spent teaching must be accounted for. This is done monthly at all 48 geographically-separated training units across the world. The data is then manually compiled to generate a monthly report. More than 90% of the data manually input by instructors is already accounted for in other systems which leads to inconsistency and wasted time.

Proposed Solution

Languages and Frameworks

The server limitation for FTSS is a critical consideration that greatly limits the options available; additionally, any desktop executable would require an extensive approval and

certification process. Therefore, a web application that leverages existing approved information systems is ideal. Within the Air Force, an enterprise Microsoft SharePoint server farm is readily available for anyone to use and create site collections. This platform is secured using DoD Common Access Cards and all permissions are handled through a standard web interface. The AF version runs on SharePoint 2010 which provides an Open Data (OData) Protocol API for interacting with SharePoint data sources directly as defined by the OData 2.0 Specification published by Microsoft on February 13, 2014. This API enables client-side applications to perform CRUD operations using HTTP.

When considering client-side frameworks for web applications, there are several choices available including AngularJS, Backbone and Ember. Of these three, only AngularJS and Ember are full-featured client-side frameworks with the tools needed to build a robust solution for FTSS. AngularJS is a more flexible, less opinionated framework which provides the flexibility to deal with the uniqueness of SharePoint's OData API. AngularJS is a Google-backed rich Single Page Application (SPA) framework that enable rapid prototyping using regular HTML and plain JavaScript objects for the model (InfoWorld, 2013).

The primary languages for FTSS are JavaScript, HTML 5 and CSS 3. Additionally, CoffeeScript and Jade are used in the source code and later compiled into JavaScript and HTML during the build process. FTSS will require a modern, standards-based web browser such as Internet Explorer 10 as it will utilize numerous HTML 5 features to optimize the user experience.

Build Process and Development Environment

Over the years I have used many Integrated Development Environments starting with Eclipse introduced to me at Trident University. After years of using Eclipse and then Net Beans

I now use JetBrain's WebStorm for web application development. WebStorm has native extensions for AngularJS, CSS 3 and HTML 5 in addition to source control and terminal support. For this application, I chose to use Brunch, a relatively new automated build system running on NodeJS. This system automates error checking, file watching, compression and testing to enable rapid prototyping. Additionally, Brunch uses the Bower package management system to enable one-line dependency injection of third-party libraries. For years I used Ant, Brunch is an incredibly powerful, fully automated build system that is years ahead of Ant both in capability and simplicity of configuration.

Conclusion

We have now defined the business problem, a possible solution and the languages and tools that will be used to create this application. The Air Force has placed significant restraints on what technologies can bus used to create FTSS; fortunately, today there are many technologies available today that will enable even these significant limitations to be overcome to produce a viable solution.

References

Air Force Instruction 36-2232 Maintenance Training. (2010, June 21). United States Air force.

Burkhow, J. (2011, August 13). Data and the "Single Source of Truth" | Data Enthusiast. Retrieved from http://www.dataenthusiast.com/2011/08/data-single-source-truth/

Gray, J. (1981). *The Transaction Concept: Virtues and Limitations* (No. Tandem TR 81.3) (p. 23). Cupertino, CA: Tandem Computers Incorporated. Retrieved from http://research.microsoft.com/en-us/um/people/gray/papers/theTransactionConcept.pdf

InfoWorld, P. K. |. (2013, October 29). What's so special about Google's AngularJS. *InfoWorld*. Retrieved April 21, 2014, from http://www.infoworld.com/t/javascript/whats-so-special-about-googles-angularjs-229673

Microsoft. (2010, May). Introduction to Collaborative Application Markup Language (CAML). *Microsoft Developer Network*. Retrieved from http://msdn.microsoft.com/en-us/library/office/ms426449(v=office.14).aspx

Open Data Protocol 2.0 Specification. (2014, February 13). Microsoft Corporation. Retrieved from http://www.odata.org/documentation/odata-version-2-0/

Sampson, D. (2012, June 10). Dave Sampson's Adventures in IT: Simple SharePoint 2010 Workflow Error Notifications. Retrieved from http://dave-sampson.blogspot.ca/2012/06/simple-sharepoint-2010-workflow-error.html