GO ABER - FITNESS TRACKING APPLICATION

SEM5640 - Developing Advanced Internet-Based Applications

Connor Goddard (clg11), Samuel Jackson (slj11), Helen Harman (heh14), Daniel McGuckin (dam44), Craig Heptinstall (crh13)

January 13, 2016

Aberystwyth University

INTRODUCTION

Project: Develop an application that acquires activity data from 3rd party sources and keep users engaged through challenges and progress updates

Two systems produced: One Java EE, One .NET

OVERVIEW

- Project Status
- Methodology
- Design
- Implementation & Issues
- Live Demo
- Evaluation & Conclusions

Project Status

PROJECT STATUS

- Majority of requirements completed, with a small number of exceptions
  - Fitbit connectivity in JavaEE
  - Email sending from application
- Application tests
  - Unit & Regression Testing - Undertaken by component author. Included the use of mocked test doubles.
  - Acceptance Testing - Undertaken either: manually as part of integration process; or automatically via Cucumber framework.

Project Management

PROJECT CHARACTERISTICS

1. Team Resources
    - All members expected to hold development responsibilities.
    - Delegation of tasks manageable at the scale of an *individual developer*.
2. Technical Complexity
    - Lack of prior team knowledge and experience in primary technologies (ASP.NET & JavaEE).
    - Required flexibility with respect to work (re-)scheduling and design changes.
3. Project Budget
    - Prioritisation of work to be crucial given expected technical complexity.
    - No particular expectation for customer-led requirements changes, but important to provide some buffer for this.

DEVELOPMENT METHODOLOGY

SCRUM agile-based approach selected as the primary development methodology.

Provided *regular* opportunities to re-evaluate work priorities, and to discuss technical issues as a team.

Up-front planning of work allowed for clear visibility in terms of task ownership and bug management.

SCRUM does not enforce a fixed set of development practices. *This allowed for flexibility with respect to testing, design & refactoring*.

DEVELOPMENT APPROACH

Work Planning

- 6-day Sprint cycles (12-day for first sprint).
- Weekly sprint retrospective and sprint planning meetings. - In collaboration with project manager (Nigel)

Design

- Weekly SCRUM meetings complemented evolutionary design, providing regular opportunities to re-visit the design *all together as a team*.

Development

- Continuous integration (Visual Studio Online).
- All individual development work undertaken in both JavaEE & ASP.NET.

Testing

- No “official” testing strategy was adopted, however aspects from both TDD and BDD were utilised (e.g. Acceptance testing and regular re-factoring).

TEAM PERFORMANCE

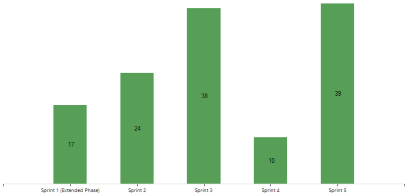


Figure 1: Team velocity performance over project duration.

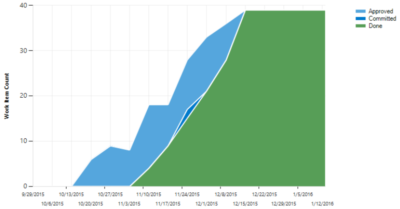


Figure 2: Cumulative flow of features over project duration.

Design

DESIGN

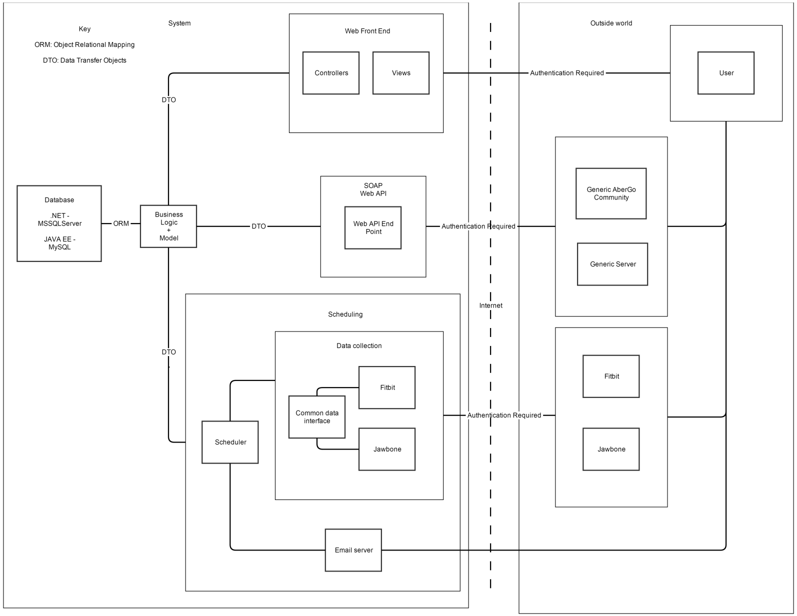
From requirements analysis we had a couple of areas where we felt needed to be discussed in more depth.

- Conceptual system architecture
- Entity relationship diagram

DESIGN - ARCHITECTURE

Key points:

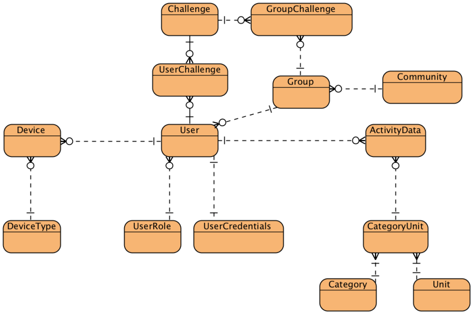
- Our first thoughts on a very high level conceptual overview of the system.
- Should be essentially the same for both. Technology specific decisions left to developer.
- Three distinct client facing parts.
- Business logic deliberately left vague.


Key
ORM: Object Relational Mapping
DTO: Data Transfer Objects
System
Web Front End
Controllers
Views
Authentication Required
Outside world
User
DTO
Database
.NET - MSSQLServer
JAVA EE - MySQL
ORM
Business Logic + Model
DTO
SOAP Web API
Web API End Point
Authentication Required
Generic AberGo Community
Generic Server
Internet
Scheduling
Data collection
DTO
Fitbit
Common data interface
Jawbone
Scheduler
Authentication Required
Fitbit
Jawbone
Email server

DESIGN - ERD

Key Points:

- Artefact from our first discussion about the database model
- Activity data: One table for all types, one row per entry
- User can only have one role
- Our data model very closely mimicked this.



Implementation

IMPLEMENTATION

	JavaEE	ASP.NET
Authentication	JDBC Realm	Identity Framework
ORM	JPA (EclipseLink)	Entity Framework (Code First)
Scheduling	Managed Execution Schedular	Hangfire Library
UI	Primefaces Library	ASP.NET Razor

Table 1: Key technology choices between JavaEE & ASP.NET platforms.

IMPLEMENTATION

Key Issues:

1. JavaEE Realms
   - Lack of clear vendor documentation.
   - Glassfish configuration (`asadmin` tool for scripting)
2. Change of RDBMS
   - Switched from MySQL to SQL Server for ASP.NET - default support by Identity Framework.
3. Multi-tier Architecture
   - Significant proportion of business logic residing in controller classes or service classes organised in WAR.
   - Partial support for Service Layer pattern (via dedicated service classes).

Demonstration

EVALUATION

- Starting development
  - Key concerns and questions addressed
  - Setting up development machines was time consuming, including a database change
- During development and testing
  - Use of TFS was hugely useful during sprints
  - Tests were created, though more could have been done, and more efficiently
- Team collaborated successfully, and put in an even amount of effort from the requirements stage through to the documentation and demonstration stage of the project

CONCLUSIONS

- Produced two working applications
- Followed methodology & improved as we went along
- Major issues were time, testing, and lack of experience with technology.

Any Questions?