2015

University of North Carolina Wilmington

Master of Science in

Computer Science and Information Systems

Proceedings

https://csbapp.uncw.edu/mscsis

A Calendar for Displaying Data from Objects on the Salesforce.com Platform

By Josh Kraszeski

Committee Members:

Chairperson: Douglas Kline

Eric Patterson

Thomas Janicki

Abstract

Although a comprehensive CRM platform, there is no native functionality in the Salesforce.com platform to display event information from any sObject in a responsive, web-based calendar. The purpose of this paper is to research the process of developing such a feature within the Force.com Platform as a Service (PaaS) in a manner such that it may be productized and deployed in a managed package to any Salesforce.com subscriber. This software package was developed using Apex, Visualforce, JavaSript, HTML, and CSS3. Furthermore, this paper will investigate the tools available to the Force.com developer for creating such a feature as well as the Software Development Life Cycle on the Force.com platform. This paper also includes an analysis of different methodologies for achieving the business requirements of this project as well as suggestions that may improve future versions.

Table of Contents

1 Introduction

1.1 Introductory Paragraph

This paper documents the planning, analysis, and design used in the creation of a responsive web-based calendar that can display data from any Salesforce.com object containing a date field. Additionally, it aims to describe the technologies being leveraged throughout the development process. This paper outlines the need for the system and its requirements as well as the challenges faced in the development process.

1.2 What is nCino and how do they leverage Salesforce.com?

nCino Inc. is a software development company that leverages the Salesforce.com Platform as a Service (PaaS) to build cloud applications. Currently, the target market of nCino is limited to financial institutions. Software applications are developed at nCino using a variety of methodologies similar to those in other software development companies.

Building applications on the Salesforce.com platform enables nCino to create product quickly without needing to engineer server-side infrastructure. The task of serving the application is handled entirely by the Salesforce.com platform, thus minimizing investment required. Furthermore, this promotes high-speed growth, a critical need for a software company aiming to maximize market share.

1.3 How would a calendar built on Salesforce.com benefit nCino?

nCino has a variety of internal and customer facing uses for calendar control. Internally, the calendar may be used for visualizing project deadlines to increase scheduling efficiency. When implemented with nCino customers, the calendar will be used to display relevant loan and collateral information. For example, a manager may use it to see how many loans are closed on a given day.

1.4 Summary of cost-benefit analysis

Cost benefit analysis was performed to determine the financial validity of this project. First, vendors providing similar products were analyzed by investigating their feature set and implementation cost. Then the cost of developing a solution was analyzed and compared with existing vendors.

1.5 Summary of how the project concluded

The project concluded with the successful implementation of a calendar deployed on the Salesforce.com platform that met all current business requirements. Stakeholders were satisfied with both the user experience and user interface.

The software was completed with the ability to be configured for any object in Salesforce as desired. Additionally, a configuration GUI was created to streamline the configuration regardless of the technical skillset of the implementer. The calendar contained views for month, day, and year.

Prior to working on this project my knowledge of AJAX and JavaScript was quite limited. Throughout this process I learned how to use JavaScript to create dynamic client-side applications. By leveraging AJAX and JavaScript remoting actions in Salesforce, I learned how to enhance my client-side applications with the ability to interact with backend Salesforce servers.

The rest of this paper is organized as follows: Section 2 discusses background information relating to the project and analyzes the technologies leveraged in

this project. Section 3 outlines the project environment and development methodologies utilized. Section 4 discusses the scope and deliverables specific to this project. Section 5 retrospectively reviews the project after its completion.

2 Background & Analysis

Currently, calendar functionality is limited on the Salesforce.com platform. This section will describe the current calendar on the Salesforce.com platform as well as opportunities for enhancement. A thorough cost benefit analysis will be performed to determine the appropriate course of action when selecting a solution to the problem described.

2.1 Statement of Problem

2.1.1 Current Date field interpretation process

Currently, date fields utilized in data objects on the Salesforce.com platform must be interpreted in their simple, plain text form. There is no native functionality to display this information on a visual calendar.

2.1.2 Opportunities for improvement

- Incorporate data into a visual calendar to enhance the users ability to interpret data and improve decision-making.

- Allow calendar integration so that it may be configured with any custom object.
- Enable calendar customization so only relevant data is visible in the calendar.
- Provide the user with the ability to interact with data in the calendar, so they may adjust dates or change information as needed.

2.1.3 Alternative Solutions

- A managed package calendar is available for installation into any Salesforce Org via the AppExchange for $8.00 a month. This cost may seem insignificant for only a few users, but quickly becomes substantial as more users are adopted. Furthermore, it lacks configurability desired by the client such as the ability to modify source code for custom implementations.

2.1.4 Business Value

- By developing a custom calendar owned by nCino, we will be able to offer much greater configurability thus insuring that it functions in accordance with the business needs.

- Cost will be significantly reduced. If you consider the calendars' utilization by a mere 100 employees, we save $800.00 a month or almost $10,000 a year.
- However, it is likely this functionality will be transferred to our clients in a variety of forms resulting in significant additional profit margins for the company. Currently, nCino has over 3,000 users on the platform. By building this in house, instead of purchasing the existing calendar on the app exchange, we will be able to offer this feature to our customers and save approximately $288,000 per year.
- Please keep in mind these are conservative estimations that don't consider additional customers to be gained or the unforeseen benefits that the configurability of the calendar will provide as it evolves. In reality, the cost savings is likely to be exponentially more than the figures described.

2.2 nCino Inc.

nCino Inc. is a cloud computing company founded in Wilmington that leverages the Force.com platform. Currently, nCino's core product focuses on streamlining the loan origination process for Financial Institutions. This is accomplished through a variety features including:

- Integrations with 3rd party vendors
    - Equifax
    - CBC Innovis
    - FileNet
    - Precision Lender
- Workflow and Process Automation
- Document Storage
- Credit Analysis

Since inception in 2012, nCino has grown steadily to become a leader in the financial services industry. Initially, small region banks (assets < 10 B) were the primary focus for nCino sales efforts. Recently nCino has expanded into the large financial institution space and signed such clients as SunTrust.

2.3 Force.com

Force.com is a platform as a service that allows developers to create applications that interact with Salesforce.com. It utilizes a proprietary language known as Apex to develop server-side business logic while HTML pages are generated via a proprietary markup language known as Visualforce. Force.com is a multitenant architecture where the application is housed in the Salesforce.com infrastructure.

2.3.1 Force.com PaaS Terminology

- Instance
    - This is a complete set of systems, storage, and network infrastructure that provides service to a subset of Force.com customers.
- Superpod
    - A set of systems including servers, storage arrays and network infrastructure, load balancers, and mail servers. Multiple instances may be run at a given data center and using Superpods provides isolation should one instance encounter problems.
- Org
    - This is the production environment with which the client interacts. It is highly customizable to the business needs of a given organization.

Currently, the Force.com PaaS handles more than 1.3 Billion transactions per day. Load balancers and tiered processing solutions based on the needs of a given customer allow the platform to operate smoothly. Application servers utilize Hotspot JVM.

Data is transmitted between the App server and Database tier by fiber optic connections. Oracle databases are run on the database tier in a multitenant architecture.

2.4 Bootstrap

Bootstrap is a free and open-source front-end framework that combines a collection of HTML, CSS, and JavaScript tools to enable more rapid development. The primary focus of the Bootstrap framework is to allow the developer to create a responsive web design with minimal coding.

Responsive web designs automatically adjust the output to the type of device accessing the website. For example, if a responsive website is viewed on a desktop, it will fill the screen area and appear much like a normal webpage. However, if this same site is accessed via a mobile device, it will be rendered in a mobile-friendly version optimized for touch interaction on small screens. The beauty of Bootstrap is that it allows the developer to code HTML and CSS only once for a given page and it will automatically be rendered appropriately on a viewer's device. A twelve-column grid layout is acts as the core methodology for creating responsive design.

Bootstrap relies heavily on LESS stylesheets to implement UI components. Developers can modify the stylesheets that are included in Bootstrap to customize the UI. By default a website using the Bootstrap framework will be visible optimized for 4 screen sizes:

2.5 Justification

2.5.1 Build vs. Buy vs. Customize

Within Systems Analysis methodologies there are three primary approaches to development and implementation of a new system, they are:

- Build a new system
- Buy a system
- Customize an existing system

In order for any of the above approaches to be considered, they must meet the requirements of the system. Various constraints such as time, cost, and resources must be considered when determining which methodology to implement.

2.5.2 Build

Generally new systems are built when alterative solutions don't meet the needs of the client. Also, one can follow this path when existing systems require a high level of customization to meet the needs of the client.

However, there are drawbacks to building a new system. For example it is often an expensive and time-consuming process. Furthermore, software development knowledge is required and may be beyond the skillset possessed by the client. Finally, support is often limited to those who built the system, meaning one relies on those who built the system to troubleshoot problems. This can be a significant problem if staffing changes result in a loss of the initial developers.

In the case of nCino, the option to build a new system resulted in the highest degree of flexibility for the solution. As a software company it is beneficial to have a system that is designed with your existing software in mind. Furthermore, intellectual property ownership is a primary concern of a software company and building the solution in house allows the firm to retain ownership of that property.

2.5.3 Buy

An alternative to building a solution is to buy an existing solution. There are many benefits to buying an existing solution that one must consider:

- It will likely be the most cost effective.
- Installation is generally much faster than building a new system.
- Support is usually available to those who purchase a pre-built solution.

  This may result in a higher degree of long-term satisfaction as well as user adoption.

  However, this is not to say that buying a system does not have drawbacks. For example:
- Pre-built systems may include functionality that is not needed by the user.
- Alternatively, it may be lacking in functionality required by the user.
- Source code is unavailable to the client.

2.5.3.1 Cost

- Many of these systems require additional fees for support, long-term maintenance, and upgrades.
- Once locked-in to a solution, subscription costs may rise over time, resulting in a significant financial burden on the client.

Although Salesforce.com includes a built-in calendar, it lacks the ability to display information based on dates contained in a record. Salesforce includes standard functionality known as tasks, which may be displayed on the calendar. However, this is severely limiting to the developer who is customizing the platform. Furthermore, the Salesforce.com calendar is not responsive and uses an extremely outdated UI.

2.5.4 Customize

Customization allows one to blend the features and both build and buy alternatives. By purchasing a pre-built solution and customizing to meet the needs of the client, cost savings can be realized while the benefits of a solution customized to business needs remain.

In the case of nCino, the existing solutions lacked the customization required by the organization as the ability to manipulate the underlying source code was desired. Also, existing solutions lacked a responsive design, which is a primary focus for nCino's vision of mobile compatibility for all features.

3 Project Environment & Processes

This section describes the development environment and process utilized in this project. First, we examine the process used for feature development. For this project a hybrid approach leveraging both the spiral iterative model and agile development process was used. Secondly, this section examines the environment where the methodologies were practiced. This includes tools used both client and server side as well as the IDE and source control tools.

3.1 Review of Development Methodologies

3.1.1 Basic Development Methodology

Throughout the development of this project a spiral iterative model was utilized in tandem with Agile Software Development methodologies. The spiral model was used for the design and creation of features, while Agile methodologies were used in planning each step of the implementation process.

3.1.2 Spiral Iterative Model

The Spiral iterative model contains three primary stages: Add feature, test feature, re-factor. For each feature the Software Development Life-cycle (SDLC) is used for the planning and implementation. The first three stages of the SDLC

(Modeling, Analysis, and Design) were accomplished during the fall of 2014. Code generation and testing stages were then completed in the spring of 2015. The final stage, maintenance, will be completed in the coming months as the system is implemented in a more widespread environment.

3.1.3 Agile Development Methodologies

While coding this system based on the analysis conducted in the Fall of 2014, agile methodologies were utilized. Agile methodologies use an iterative model that encouraged the rapid development of features. Specifically, scrum methods were used that enabled high levels of collaboration among team members. Although this was an individual project, I still kept my team informed about my progress on a daily basis and reached out to them for guidance when platform specific questions arose.

3.2 GitHub

3.2.1 Source Control

GitHub is a free online tool used for source code management and revision control. There are public, open-source repositories, to which anyone may contribute. Also, GitHub offers paid private repositories that may be used for individual or organizational use. nCino uses a paid repository to house its source

code and allow team collaboration. GitHub tracks all changes made to source code enabling one to revert back to previous versions if needed.

3.2.2 Collaboration

Collaboration is encouraged on GitHub because all source code is available to any team member. One may contribute modification and then have those modifications merged into the master branch following code review.

3.2.3 Continuous Integration testing

By utilizing GitHub in combination with Bamboo, continuous integration testing was performed. Whenever source code was uploaded to GitHub, Bamboo used an Amazon EC2 instance to run all test cases relevant to the platform. This provides the assurance that unit tests will not fail unexpectedly, but rather fail on the initial upload so the error may be quickly corrected.

3.2.4 Backups

GitHub operated as the primary platform for backups of source code. However, backups were also performed on the Salesforce.com org used for development. Furthermore, backups were performed locally.

3.3 Development Environment

The development environment utilized in this project consisted of laptops running Windows, Linux, and Mac OSX. Due to the cloud based servers running all software relating to the application, reliance on specific hardware was minimal.

Sublime Text with the Mavens Mate plugin was primary IDE for this project. Code was written on the local machine then pushed to Salesforce.com servers for compilation and feature testing. Unit tests were executed by the Mavens Mate test runner UI on the Salesforce.com server.

Salesforce.com provides a comprehensive developer website. This was the primary source of knowledge for new principles employed in this project that were new to the developer and proprietary to the Salesforce.com platform.

3.4 Development Methodology

The primary development methodology utilized was the spiral iterative model in tandem with agile development methodologies. The spiral model provided guidance for the development of individual features while agile development provided the flexibility to respond to changing requirements.

This spiral iterative model was used to produce small prototypes that satisfied a specific feature request for the project. During each iteration, requirements for the feature to be developed were determined. This included user acceptance criteria that would be used to determine if the feature development was successful. A development plan was then created for the feature that described how the feature would be developed. During this planning phase, any risks that may prevent the feature from being developed were also identified. Once planning was complete, the feature was developed to meet the specified acceptance criteria. This was followed by unit testing that helped to ensure future development did not break any existing features. Some suggested practices from the iterative approach were not used. For example, comprehensive test plans were not developed, as this was an internal feature prototype that was not intended for production release at this time.

Throughout each iterative cycle of feature development, a specific development workflow was enforced. This provided a consistent process that ensured code met high standards and maintained appropriate levels of testing and documentation.

The first step in the process was to synchronize the local code branch used for feature development with the remote branch stored on GitHub. This reduced the likelihood of merge conflicts and ensured the developer had the latest code to work with. At this point, feature development was initiated. The developer would then work on the feature for multiple iterations until it was complete. In some cases, feature development began with the writing of automated unit tests. These tests were used to drive feature development and ensure business logic was functioning appropriately. For example, unit tests were used to confirm the functionality of Visual Force Controller methods. After features were completed, including comprehensive unit tests, code relating to the feature was then pushed to the remote branch. After code was pushed, it was reviewed by an experienced developer to ensure best practices were employed. If the reviewer suggested changes, modifications were made to the feature and pushed to the remote branch. This cycle was repeated until all parties were satisfied.

3.5 Force.com

Within the Force.com platform, a Model View Controller (MVC) design pattern was used. The database layer acted as the model portion of the MVC.

Visualforce was then used for the view portion. Finally, Apex was functioned as the controller for the MVC architecture.

3.5.1 Database

The Force.com platform provides a persistent database layer that can be accessed via Apex methods. This is where all data relating to Salesforce objects is housed in a relational architecture.

3.5.2 APEX

Apex is the proprietary language used to create server-side business logic on the Force.com platform. It utilizes similar syntax to that of the Java language, although it is not case sensitive like Java. Apex also blends additional elements not found in Java that are unique to the Force.com platform. For example, in Apex one may code queries that interact with the underlying database without the need to create Stored Procedures or SQL statements. In Apex this is known as the Salesforce Object Query Language (SOQL). Other features unique to Apex on the Force.com platform include:

- The ability to interact with user permission set information
- Data manipulation language calls like INSERT, UPDATE and DELETE with built-in exception handling

- Locking syntax to prevent record update conflicts
- Custom public API's that can be built in Apex methods
- Built-in support for test creation and execution

3.5.3 Visualforce

Currently, Visualforce is the primary markup language used on the Salesforce.com platform. In many ways it functions similarly to traditional HTML, however it provides additional syntax that enables the end user to better interact with Apex controllers. Things such as JavaScript and CSS function normally in Visualforce.

3.5.4 JavaScript remoting

JavaScript will be used to dynamically generate the calendar. However, platform specific methods were used to pass data JSON data to the working JavaScript. This is known as JavaScript remoting on the Salesforce.com platform. By utilizing Visualforce syntax with JavaScript, one is able to call specific controller methods that provide the JavaScript with data desired for the page. Additional benefits of JavaScript remoting include its ability to serialize and deserialize the data being transmitted, thus minimizing the coding required by the developer.

3.5.5 JSON

JavaScript Object Notation (JSON) is a lightweight interchange format used for the transmission of data across the Internet. This is the primary format for data encapsulated by the controller and then sent to a JavaScript application to be read. Also, it is easy to debug as it can be easily read by humans.

3.5.6 Apex Unit Tests

Apex unit tests are used to test methods created in Apex classes. Platform specific syntax is used to define tests. These tests are used to verify that specific components of code are working correctly throughout the development process. Salesforce requires 75% test code coverage to deploy code to an org, however nCino requires 100% coverage meaning all lines of code have been hit in the testing process.

4 Project Scope & Timeline

In this section we will discuss the scope of the project as agreed to by the stakeholders. Items both in and out of scope are detailed here. Additionally, deliverables are outlined. This section also contains a timeline for completing the deliverables referenced in this section.

4.1 Included

- Functional calendar built on the Salesforce.com Platform
- Capable of being configured to support any custom object in Salesforce that has a date field
- Configurable to display desired additional fields in the calendar
- All data will be up-to-date upon page refresh

4.2 Additional items to be investigated

- Allow the user to interact with data in the calendar view and synchronize interaction with custom object (two-way binding)
- Provide mobile support
- Allow multiple users to manipulate data in the calendar at the same time and immediately see each other's changes
- Support for beginning and ending date
- Support for exporting .ics files for events

4.3 Outside of scope

Create a GUI for calendar configuration.

4.3.1 Explanation:

- This would greatly increase the time required to develop the calendar while simultaneously reducing the future customizability.
- The calendar will be used by nCino, a software development company, and thus many of our users prefer direct source code manipulation for implementation.

4.4 Deliverables

- A working prototype calendar, developed on the Salesforce.com platform, which may be configured for any custom object containing a date field.
- It will be deployed in the form of a managed package.
- Code specific to the presentation of the calendar will be contained in wrapper to ensure minimal configuration is needed.
- Code must be reusable.
- Hope to test in the nCino sandbox to ensure full functionality.
- A written investigation of two-way binding and mobile support for the calendar describing my findings through the development process.

4.5 Timeline

- HTML mockup and design finished by January 20th
    - Set up environment
        - 1 hour
    - Set up Source Control
        - 1 hour
    - Set up Libraries
        - 5 Hours
    - Create base page
        - 1 hour
    - Pull in and configure calendar
        - 8 hours
- Working calendar using SF custom object by March 1st (Unit testing included in estimation)
    - Create controller
        - 1 hour
    - Build methods to retrieve objects
        - 12 hours
    - Set component up to make remoting request for the data
        - 12 hours

    - Configure Javascript to read data from controller
        - 20 hours
- Calendar configurable for any custom object by April 10th
    - Test with several objects
        - 12 hours
    - Investigate how to handle beginning and end dates
        - 12 hours
    - Investigate 2-way binding
        - 12 hours
    - Investigate multi-user interaction
        - 6 hours
    - Investigate Mobile support
        - 4 hours
    - Investigate downloadable .ics for calendar item
        - 8 hours
- Prepare documentation and written explanation of investigation by April 20th
    - 15 hours

4.5.1 Revisited Project Timeline

In this timeline actual completion dates are shown and can be used for comparison with the original timeline. Dates and times shown in red vary from the original timeline.

- HTML mockup and design finished February 15th
    - Set up environment
        - 1 hour
    - Set up Source Control
        - 1 hour
    - Set up Libraries
        - 10 Hours
    - Create base page
        - 1 hour
    - Pull in and configure calendar
        - 12 hours
- Working calendar using SF custom object March 20th (Unit testing included in estimation)
    - Create controller
        - 1 hours
    - Build methods to retrieve objects

        - 12 hours
    - Set component up to make remoting request for the data
        - 20 hours
    - Configure Javascript to read data from controller
        - 20 hours
- Calendar configurable for any custom object by April 15th
    - Test with several objects
        - 12 hours
    - Investigate how to handle beginning and end dates
        - 2 hours
    - Investigate 2-way binding
        - 12 hours
    - Investigate multi-user interaction
        - 6 hours
    - Investigate Mobile support
        - 4 hours
    - Investigate downloadable .ics for calendar item
        - 8 hours
- Prepare documentation and written explanation of investigation by April 21st

- 20 hours

5 Project Review

This section describes the scope and deliverables that resulted from the project. It investigates whether scope creep was encountered and additional deliverables that resulted from the project.

5.1 Scope review

Scope expanded slightly throughout the project by way of investigations suggested by the committee and additional functional requirements that were determined by user testing. At no point did scope expansion result in missing deadlines or project commitments. Additional deliverables that were built by way of slight scope expansion included support for start and end dates as well as an administration tool for configuration.

Overall, the application functioned in the manner outlined in the initial planning. Additional features were added as previously discussed, but all features committed to were completed. Additional investigations recommended by the

committee resulted in the additional features that significantly improved the delivered product.

5.2 Deliverables Review

All deliverables committed to were delivered and tested. Additional deliverables, such as start and end date functionality and administration tools, were completed within the time allowed for the project as outlined in the project timeline.

5.3 Challenges

5.3.1 JavaScript and reserved words in Apex

Initially, the JavaScript in this calendar was configured to read JSON data using the naming attributes “start” and “end” for beginning and end dates of a given event. However, these are reserved words in Apex and cannot be used in the formation of an object for JSON formatting. This required extensive reworking of the JavaScript libraries. Changing “end” to “endTime” and “start” to “startTime” throughout the entirety of the JavaScript libraries pertaining to the calendar enabled the Apex remoting methods to format the data in a manner that could be consumed by the calendar.

5.3.2 Storing configurations

Additionally, storing configuration settings proved challenging on the Salesforce platform. Although Salesforce provides a wealth of tools for storing settings and configurations relative to organization settings and native Salesforce functionality, little is provided for custom configurations.

This dilemma resulted in the need to create a custom sObject in the managed packaged created for the project. This object was comprised of a simple table that stored information relevant to configuration of the calendar specific to your implementation. This was comprised of the sObject to be represented on the calendar, the fields to be mapped to the start and end dates of events, and the description field whose data would be displayed on the calendar.

After this architecture was put in place, additional research led to the discovery of an alternative solution. In the Summer 2015, the Salesforce platform released new features known as Custom Metadata Types and Custom Settings. This could have been used as an alternative to creating a configuration table and may be a more viable long-term solution.

5.3.3 Learning Unfamiliar Languages and Technologies

Perhaps the most significant challenge was learning new programming languages, Apex and JavaScript, while attempting to incorporate features from multiple frameworks in a single product. Apex is the proprietary Salesforce programming language that functions in an object-oriented nature much like Java. Also, it incorporates inline queries, known as SOQl queries, that function much like traditional SQL.

Combining the features of Apex, Bootstrap, and Javascript libraries provided a significant challenge because their combined use is not well documented. Individually they each provide comprehensive documentation on their use and implementation, but collectively leveraging them to build an application proved to be quite challenging.

The effort to combine different frameworks resulted in an increased need to understand software architecture and patterns, such as the MVC model. The more thorough one's knowledge of these core concepts, the better one will be able to leverage multiple frameworks to achieve a result that is greater than the sum of its parts.

6 Conclusion

This project resulted in a useful calendar that expanded Saleforce functionality and provided a tool to improve planning throughout an organization. Personally, it greatly increased my knowledge of not only the Salesforce platform, but also software architectures in general. Principles learned here will be greatly beneficial to my professional career in the future, both in Salesforce development and software development as a whole.

This document discussed the entire software development process for this project. Starting with background research, business needs and requirements assessment, and initial scope commitment. Furthermore, it discusses the actual process of creating the software to fulfill the business needs as well as a reflective outlook to determine the strengths and weaknesses of the project.

nCino benefited from this project by using it as an internal tool for both planning and scheduling purposes. This calendar also provides a valuable codebase for future development if similar functionality is determined necessary for our core product.

This project provides a great example to the SF developer community because it demonstrates the flexibility of the platform. It demonstrates how external frameworks, developed with no Salesforce knowledge, can be leveraged to greatly enhance functionality and features. To me, this was the greatest learning benefit of the whole project. Discovering how to blend multiple frameworks to create a single and unified feature was a true eye opener. This concept enables developers to create the best possible user experience and use the most efficient technologies providing incredible flexibility and feature capability.

7 Appendix

7.1 Project Charter

- My project committee which includes:
    - Dr. Douglas Kline (Chair)
    - Dr. Eric Patterson
    - Dr. Thomas Janicki

- The project's client is nCino Inc.
    - Jonathan Rowe (EVP Marketing)
    - Nathan Snell (EVP Product Development)

Stakeholders

- Customer – a company that has purchased nCino's software
- User – a user of the software that has been purchased by Customers
- Developer – a developer working at nCino authoring software
- Company - nCino

A brief discussion of the topics covered in the project charter is found below.

- Business Purpose and Objectives
    - Create a reusable calendar component for Salesforce.com for use at nCino in their products
    - Save company money
    - Save developer time
    - Make products clearer and more convenient

- The purpose and objective of this project is to provide an online application developed on the Salesforce.com platform that will interpret data stored in any object and display relevant date information on a graphical calendar.

- Critical Success Factors
  - This section describes the items that will determine whether or not this project is a success
  - Critical success factors are as follows:
    - Provide working calendar on the Salesforce.com platform
    - Configurable to work with any custom object that contains date fields
    - Accurately represent data in the custom object at the time of page load.
    - Continued ability for nCino support and maintenance

  - Risks
    - This section describes those things that may be a risk to the project's success
    - Primary risks being considered are the long-term sustainability of the software architecture given that Salesforce.com is a dynamic platform with frequent updates
    - Additional risks include a lack of configurability and compatibility with any Salesforce.com object.

7.2 Use Case 1

Use Case ID:	1		
Use Case Name:	Calendar for visualizing project deadlines		
Created By:	Josh Kraszeski	Last Updated By:	Josh Kraszeski
Date Created:	10/10/2014	Date Last Updated:	11/15/2014

Actor:	Project Database User
Description:	The user would like to see records with date fields displayed in a calendar view
Preconditions:	A Salesforce data object must be created
Postconditions:	The data will be displayed in a visual calendar
Priority:	Medium
Frequency of Use:	Daily
Normal Course of Events:	1. User accesses VisualForce page with calendar 2. Records are displayed on dates corresponding to their configured field 3. User clicks a specific record to access it in more detail
Special Requirements:	Salesforce org with calendar package installed
Assumptions:	Records exist with date fields

Notes and Issues:	Ability to work with any custom object may be challenging

7.3 Use Case 2

Use Case ID:	2		
Use Case Name:	Calendar for visualizing loan closing dates		
Created By:	Josh Kraszeski	Last Updated By:	Josh Kraszeski
Date Created:	2/1/2015	Date Last Updated:	2/4/2015

Actor:	Loan Officer
Description:	The user would like to see records with date fields displayed in a calendar view
Preconditions:	A loan must be created as a Salesforce data object
Postconditions:	The data will be displayed in a visual calendar
Priority:	Medium
Frequency of Use:	Daily

Normal Course of Events:	1. User accesses VisualForce page with calendar 2. Loans are displayed on dates corresponding to their closing dates 3. User clicks a loan to access it in more detail
Special Requirements:	Salesforce org with calendar package installed
Assumptions:	Records exist with date fields
Notes and Issues:	Ability to work with any custom object may be challenging

7.4 Use Case 3

Use Case ID:	3		
Use Case Name:	Calendar for visualizing account openings		
Created By:	Josh Kraszeski	Last Updated By:	Josh Kraszeski
Date Created:	2/1/2015	Date Last Updated:	2/4/2015

Actor:	Sales Representative
Description:	The user would like to see account openings with date

	fields displayed in a calendar view
Preconditions:	A account must be created as a Salesforce data object
Postconditions:	The data will be displayed in a visual calendar
Priority:	Medium
Frequency of Use:	Daily
Normal Course of Events:	1. User accesses VisualForce page with calendar 2. Accounts are displayed on dates corresponding to their closing dates 3. User clicks an account to access it in more detail
Special Requirements:	Salesforce org with calendar package installed
Assumptions:	Records exist with date fields
Notes and Issues:	Ability to work with any custom object may be challenging

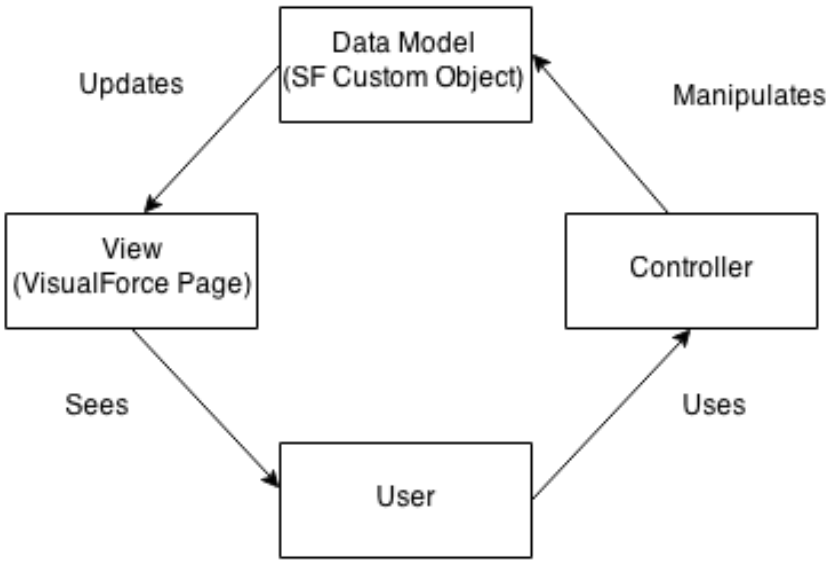
7.5 Use Case 4

Use Case ID:	4		
Use Case Name:	Calendar for visualizing loan durations from open to close		
Created By:	Josh Kraszeski	Last Updated By:	Josh Kraszeski

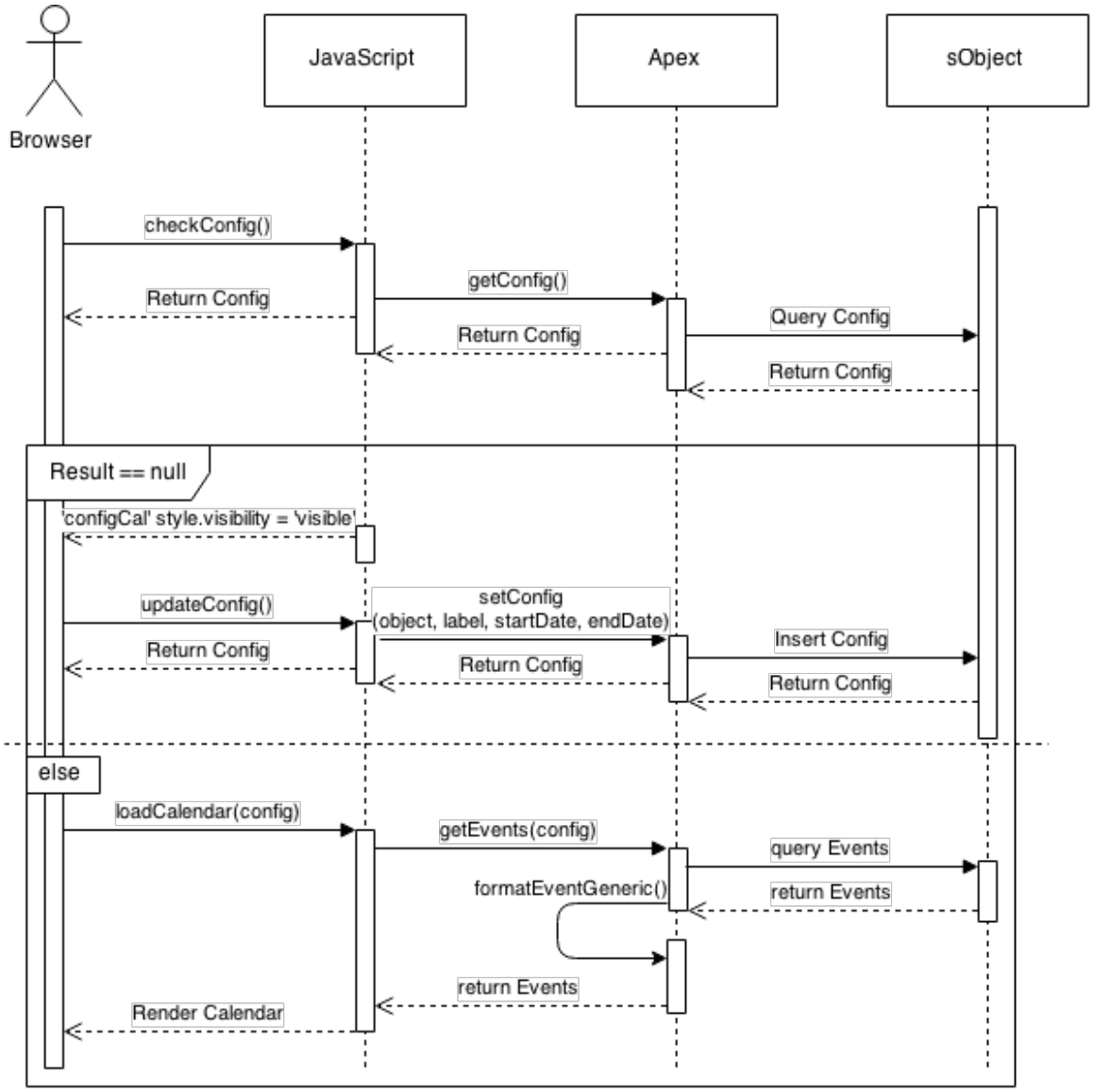
Date Created:	2/1/2015	Date Last Updated:	2/4/2015

Actor:	Loan Officer
Description:	The user would like to see loan durations displayed in a calendar view
Preconditions:	A loan must be created as a Salesforce data object
Postconditions:	The data will be displayed in a visual calendar
Priority:	Medium
Frequency of Use:	Daily
Normal Course of Events:	1. User accesses VisualForce page with calendar 2. Loans are displayed by showing the opening date and closing date for each loan 3. User clicks an account to access it in more detail
Special Requirements:	Salesforce org with calendar package installed
Assumptions:	Records exist with date fields
Notes and Issues:	Ability to work with any custom object may be challenging

7.6 MVC model as used in this project

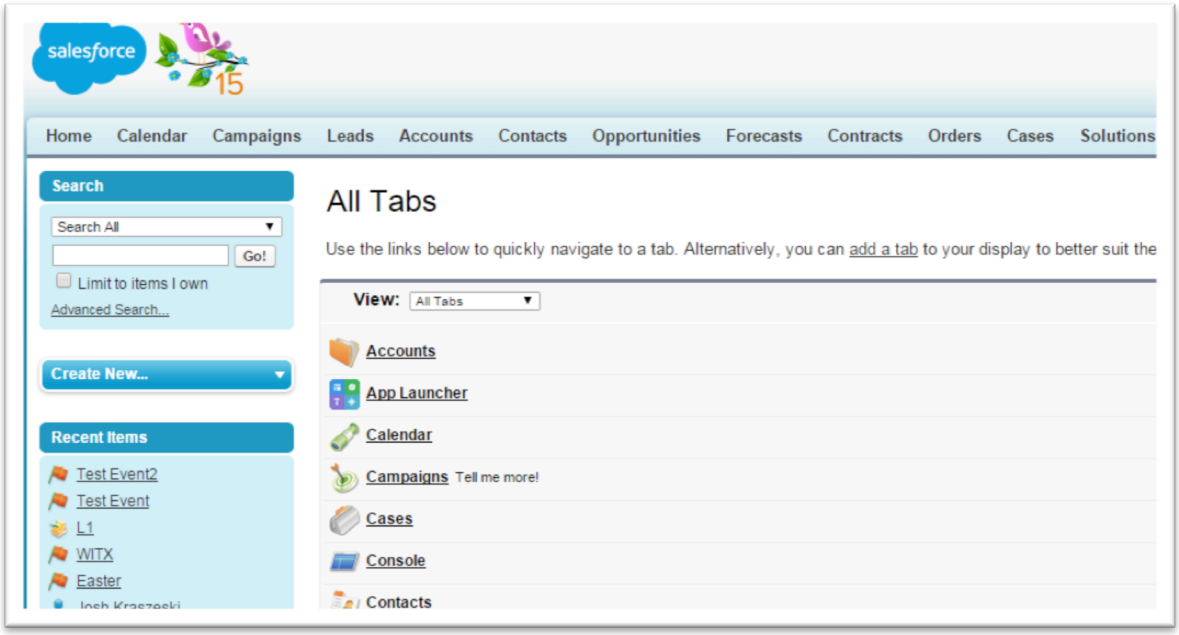


7.7 Sequence Diagram

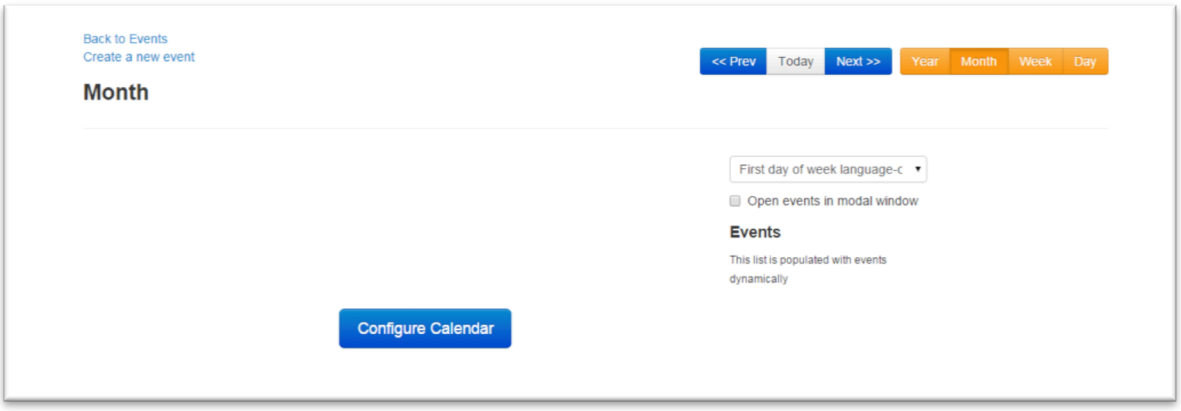


7.8 Screenshots of Calendar Implementation

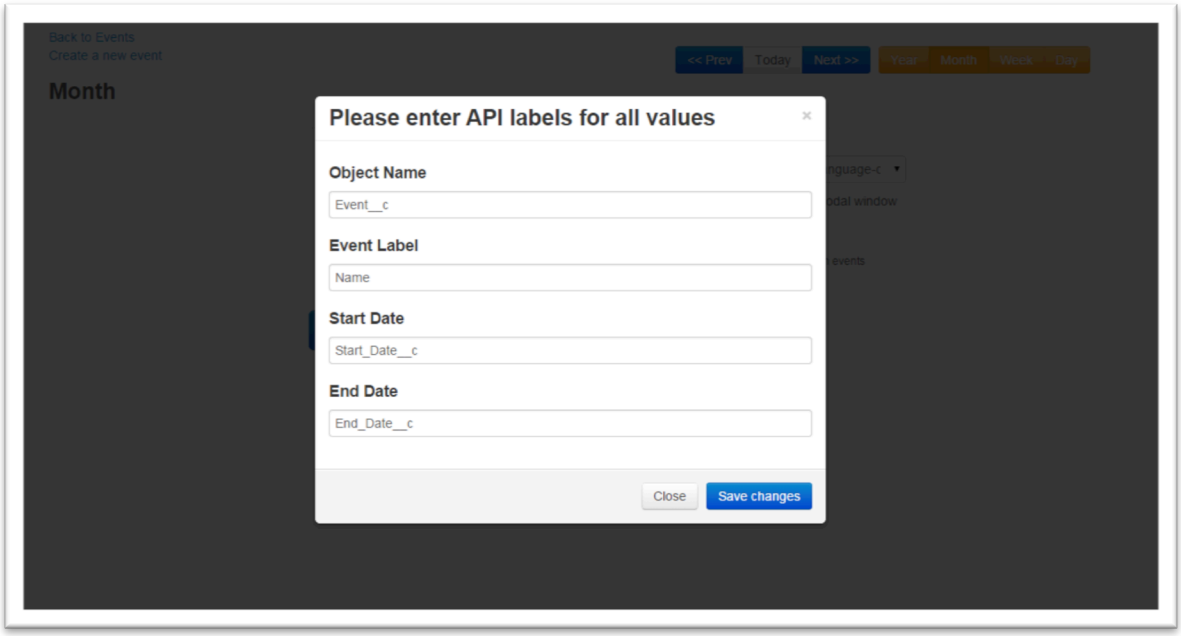
Please note the calendar tab displayed in the navigation bar below:



Upon selection of the calendar tab in the navigation bar, the user is presented with the following screen:

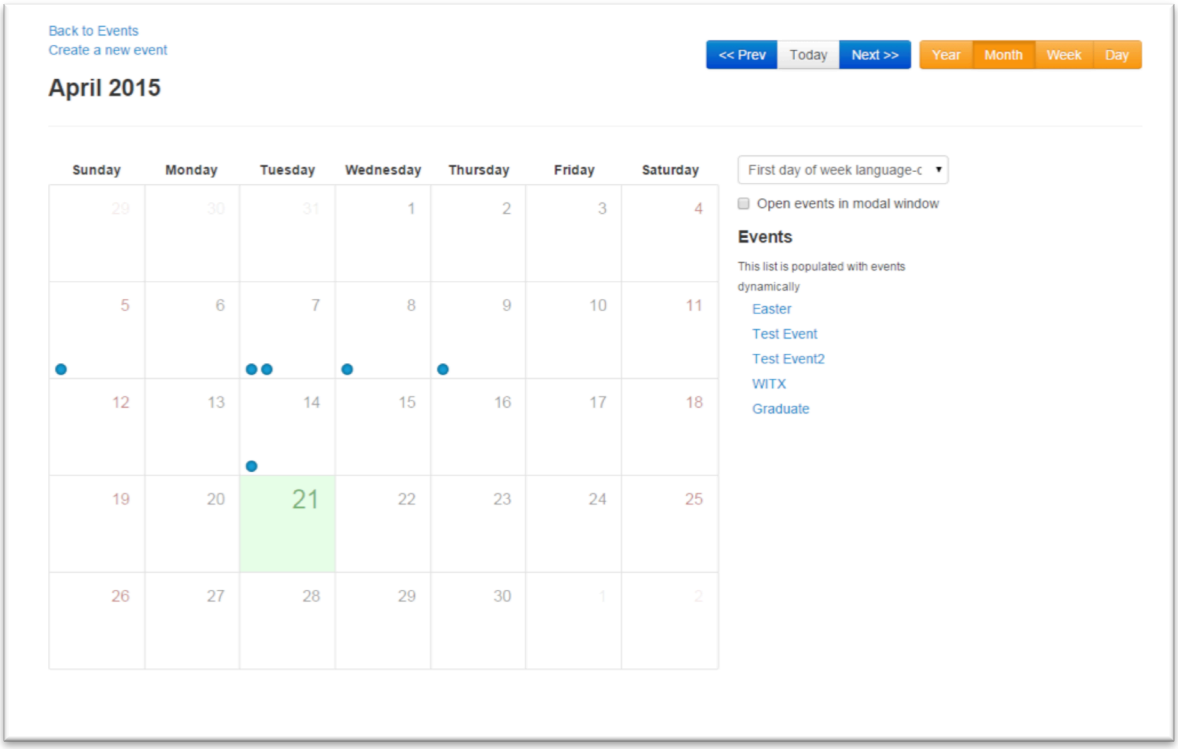


When the "Configure Calendar" button is clicked, the user is presented with the following modal:

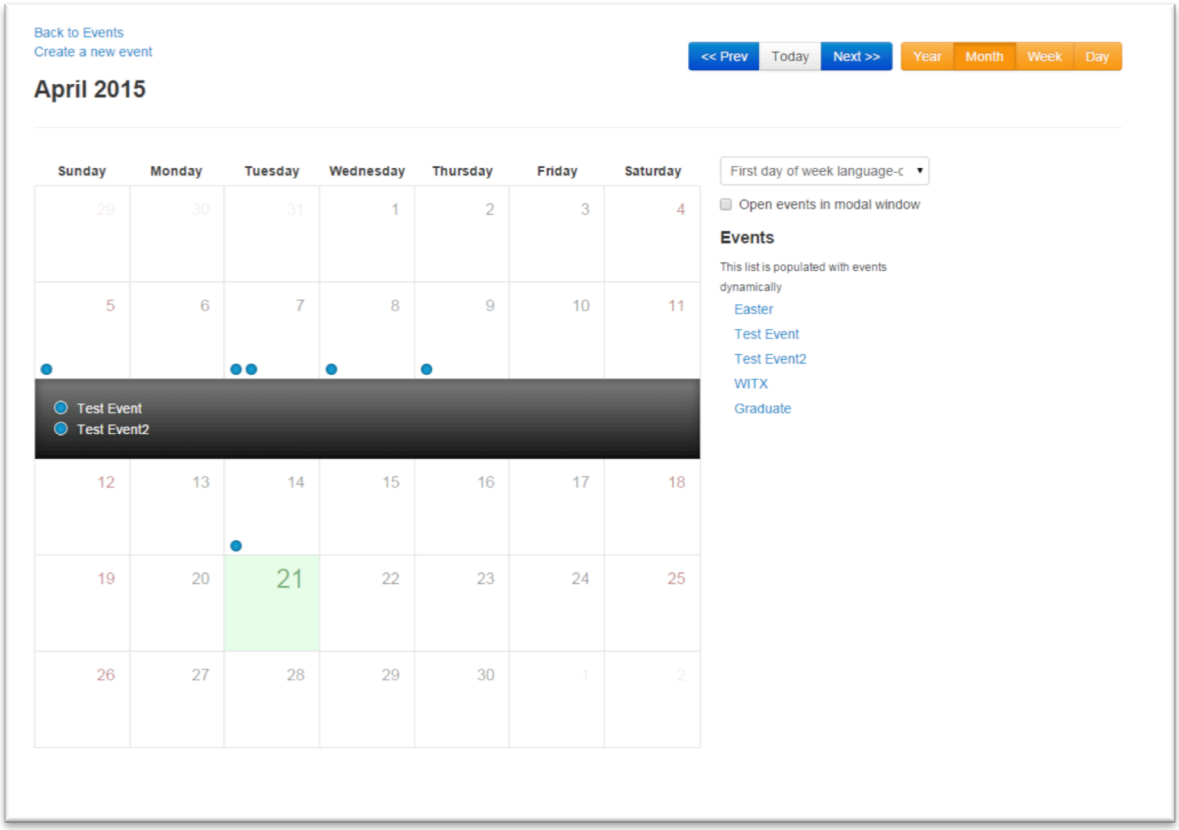


* Please note this is where the user can select the object, label, and date fields that he or she would like displayed.

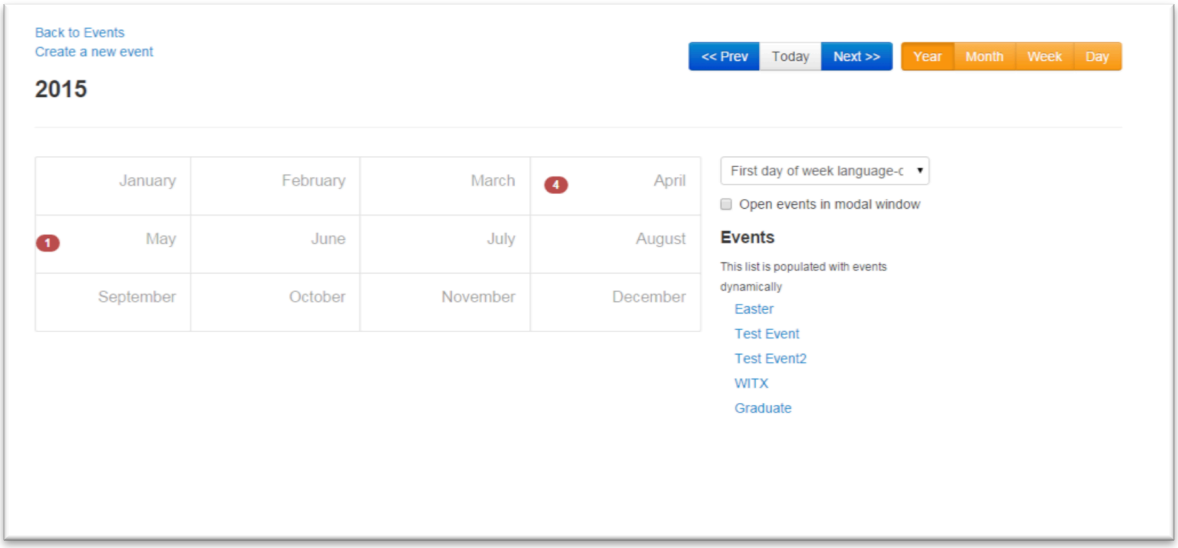
After configuration, events are displayed in a month view by default:



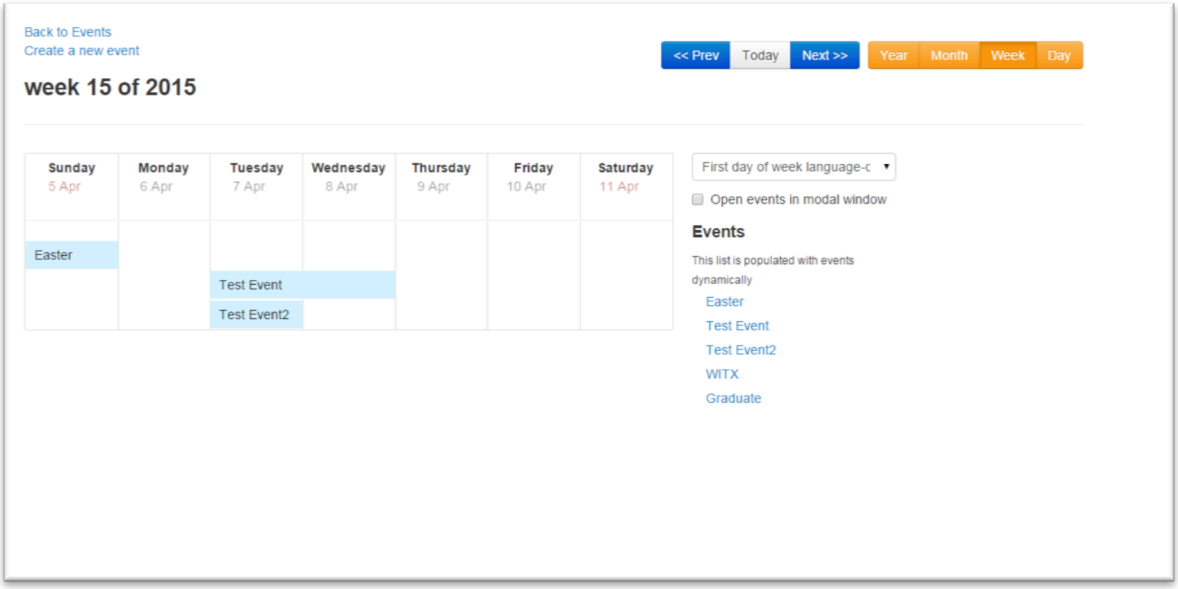
When a particular date is clicked, a more detailed UI presents the events on that date:



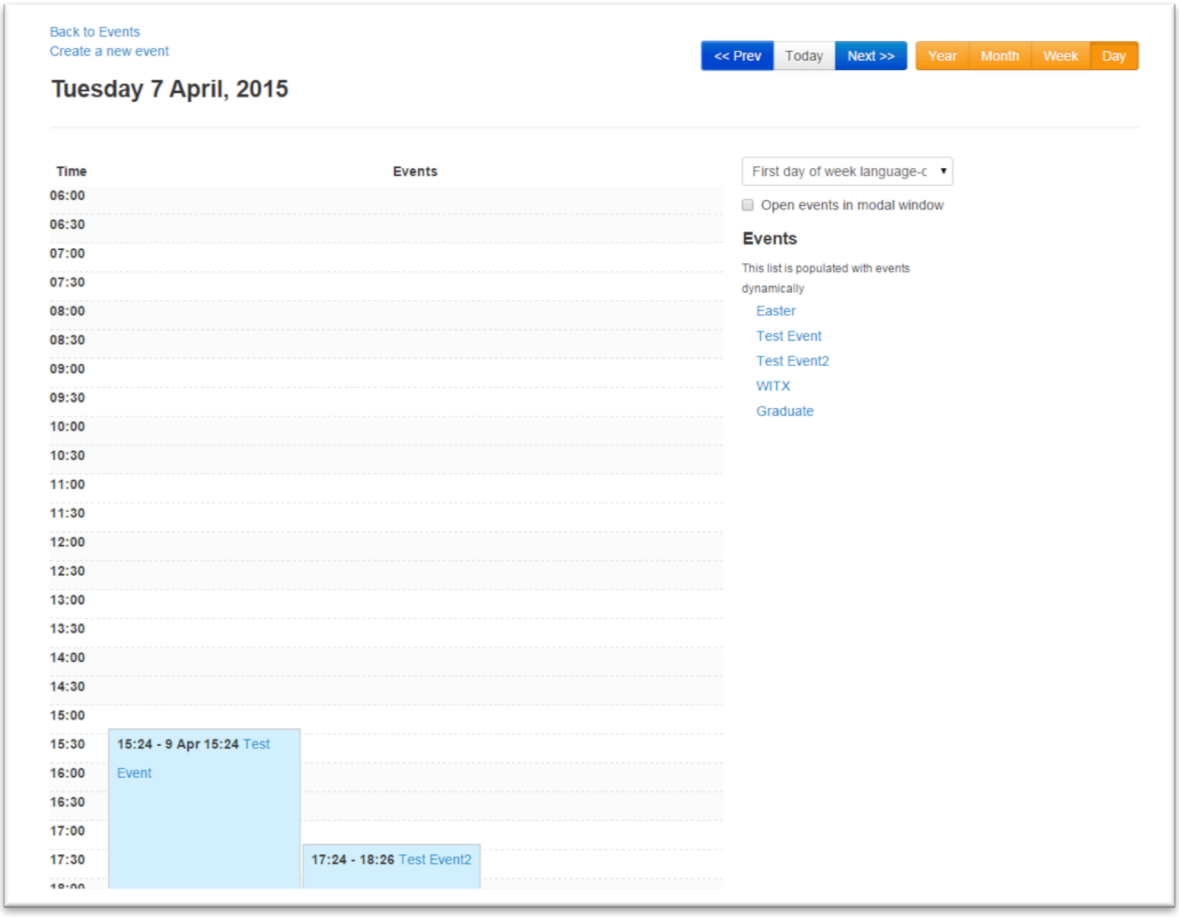
Also available is a year view that displays all events for a given year:



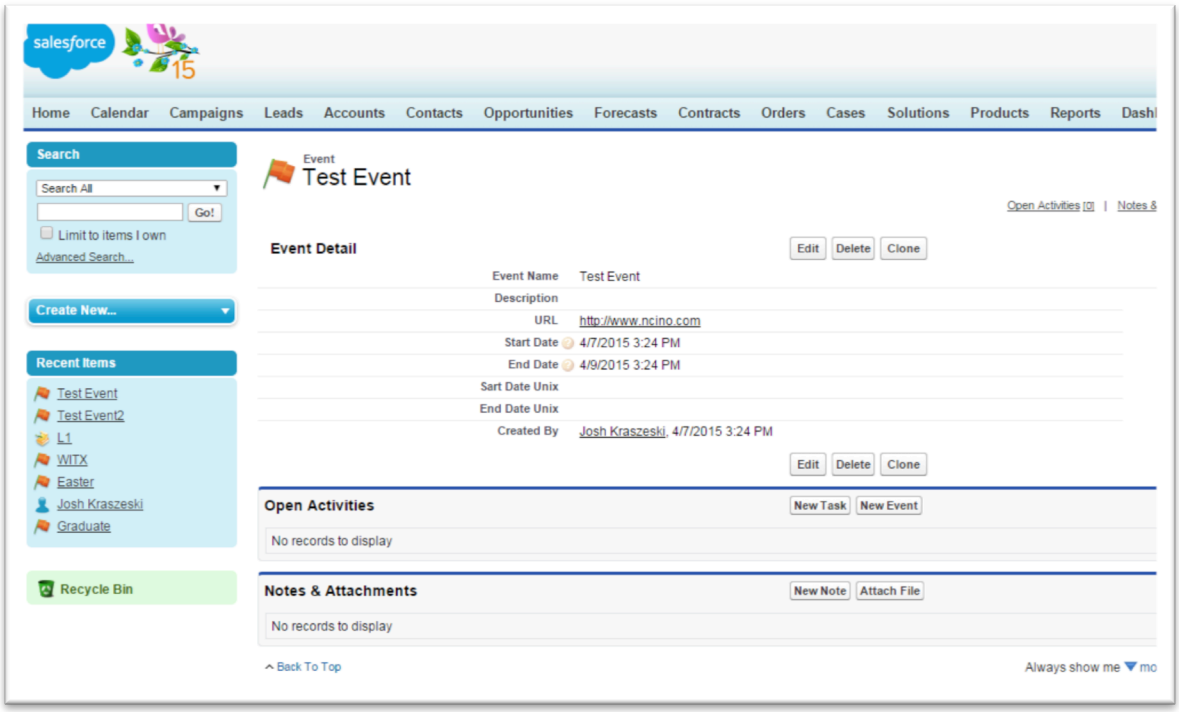
Week view:



Day view provides the most detail representation of the data:



Upon selection of an event the user is directed to the detailed record associated with that event:



7.9 Code used in controller:

```
public with sharing class EventRemoter {

        public String eventName { get; set; }
        public static sObject event { get; set; }
        public EventRemoter() { } // empty constructor

@RemoteAction
public static List<formatEventGeneral> getEvents(String objectName,
                                        SString title,
                                        String startDate,
                                        String endDate) {

                List<formatEventGeneral> formattedEventsGeneral = new
                List<formatEventGeneral>();

                List<sObject> eventList = Database.query('SELECT id, ' + startDate +
                ', ' + endDate + ', ' + title + ' FROM ' + objectName);

                for (sObject e: eventList){
                        formattedEventsGeneral.add(new
                        formatEventGeneral(String.valueOf(e.id),
                        String.valueOf(e.get(title)),
                        Datetime.valueOf(e.get(startDate)),
                        Datetime.valueOf(e.get(endDate))));
                }

                return formattedEventsGeneral;
        }

        public class formatEventGeneral{
                public String id;
                public String title;
                public String url;
                //public String class;
                public Datetime startTime;
                public Datetime endTime;

                public formatEventGeneral(String objId, String label, Datetime
                startDate, Datetime endDate){
                        id = objId;
                        title = label;
                        url = '/' + id;
                        //class = 'event-info';
                        startTime = startDate;
```

```
                    endTime = endDate;
                }
            }

}
```

7.10 Written Investigations

7.10.1 Two-way Binding

Multiple solutions for implementing two-way binding were considered throughout the development of this calendar. Both KnockoutJS and AngularJS frameworks would have accomplished the task with significantly more development. Each framework would have required that the majority of the JavaScript used in the current version of the calendar be re-written to utilize the framework. Had the calendar been developed with this in mind from the beginning, this may have been accomplishable. However, it was not possible given the time constraints of this project. Furthermore, due to the ability of each

event to link to an editable record, business requirements did not require two-way binding functionality. All business requirements were fulfilled without the need to integrate this functionality.

7.10.2 .ics Calendar File Download

This was determined to be technically feasible, but development time was instead allocated to build a configuration UI as it was seen by stakeholders as a more valuable asset. For this to be accomplished server-side Apex methods to generate the .ics file, regardless of object data type, would have been required. Additionally, simple HTML download links did not function well in the Visualforce environment because links would vary from org to org. This meant additional JavaScript methods would be required in the Visualforce page to provide links.

7.10.3 UI for Calendar Configuration

Although viewed initially as out of scope, the feasibility of this requirement and its perceived value made this an additional priority for this project. If the calendar is not configured, a "Configure Calendar" button is provided. Upon clicking this button a modal appears with text boxes for the various fields required for configuration. After entering the required values they are stored in a

configuration object. If the calendar has been previously configured, no "Configure Calendar" button is shown and the previous configuration is loaded. If more than one configuration is save in the configuration object, only the first entry will be used to render the calendar. In future iterations of the calendar this functionality could be uploaded to allow the user to select his or her desired configuration. The user may update delete a configuration from the configuration object at any time if desired.

7.10.4 Handling Objects with Beginning and End Dates

Initially, the project scope only required the display of one date field per record. However, after reviewing and manipulating the JavaScript of the selected Bootstrap calendar it was determine that two date fields were acceptable. JavaScript updates included integrating support for "EndTime" and insuring the JSON payload containing this information could be interpreted. Furthermore, Apex changes in the remoting methods were incorporated that enabled this information to be interpreted and processed appropriately.

7.10 Background Analysis of Cloud Computing and Justification for Leveraging the Technology

7.10.1 Introduction

Currently within the business community there is a significant push to adopt cloud solutions. New cloud providers are entering the market frequently and offering a wide array of cloud services. Cloud providers argue that their service provides significant competitive advantage and cost reductions to businesses.

We will examine cloud computing through a thorough analysis of its advantages and disadvantages. Also, we will discuss the history and infrastructure of cloud computing. Finally, we will combine this knowledge to determine security threats and benefits of cloud computing. Through this analysis we hope to determine if cloud computing truly provides the advantages promised.

7.10.2 What is Cloud Computing and how did it come about?

Cloud computing is defined by The National Institute of Standards and Technology *"as a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers,*

storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." (Shen)

Cloud computing is marketed in many variations including, but not limited to, software-as-a-service, infrastructure-as-a-service, on-demand computing, and internet-as-a-platform. Regardless of the name chosen to categorize a given cloud service, they all share some characteristics. For example, all of the above services are location independent, a key feature of cloud computing. Also within cloud computing, services are generally pay as you go, not a one-time upfront fee for a given service.

This is somewhat similar to timeshare computing services offered in the 1960's. With these services an entity (usually a business or individual) could "rent" computing power from a device they did not own. This reduced the cost of utilizing computers and made them accessible to those who only needed computing power on a limited basis. Then, during the 1980's, the personal computer revolution occurred. This greatly reduced the cost of owning a computer and made it possible for one to harness computing power from their home or office. Also, this resulted in the transition of the physical location of data from the mainframe to the local PC. Then came the Internet, allowing most

computer driven devices to communicate with each other. Cloud computing is essentially the marriage of mainframe computing power with local devices, all accomplished through the connection provided by the Internet. This enables the end user to harness the power of servers and mainframes from any Internet connected device. (KAY) (Tasneem)

Another key advantage that spurred the growth of cloud computing was the ability of providers to locate data centers at cheap locations. As we previously discussed in cloud computing location is irrelevant, this enabled businesses to consider low-cost options and translate this cost reduction to the end user. As stated in the article "A View of Cloud Computing", cloud computing has resulted in:

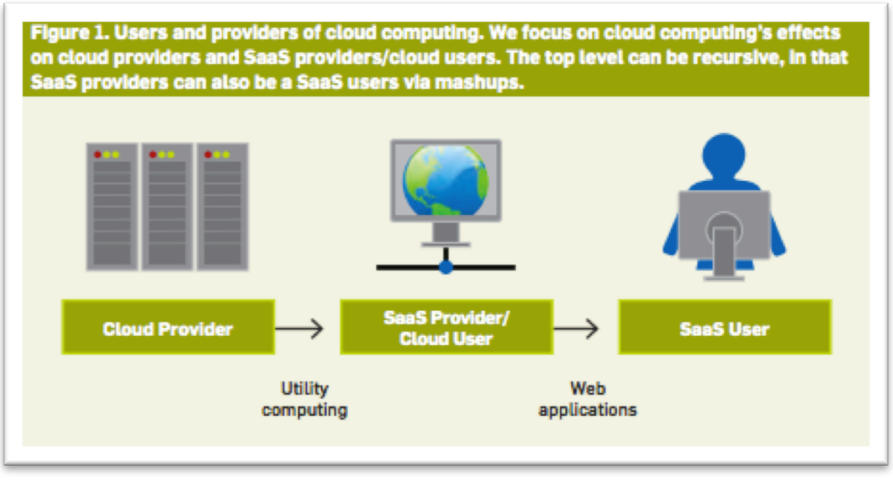
> *"factors of 5 to 7 decrease in cost of electricity, network bandwidth, operations, software, and hardware available at these very large economies of scale. These factors, combined with statistical multiplexing to increase utilization compared to traditional data centers, meant that cloud computing could offer services below the costs of a medium-sized data center and yet still make a good profit."* (Ambrust)

7.10.3 What are the proposed benefits?

All this sounds great, but how does it benefit the individual? First of all, it makes the local devices (Mobile or PC) computing power irrelevant to the task, as the processing will take place in the cloud. This reduces the cost to the end user, as less powerful hardware is needed. Secondly, cloud computing virtually eliminates installation and configuration issues as the primary interface is the web browser. Many of the benefits to the individual are also benefits to software vendor. For example the reduction in installation and configuration issues also reduces the maintenance required by the vendor. Also, software vendors are free to choose the cloud platform that best fits their application. When maintenance is required or updates are needed, the vendor simply pushes them to their cloud platform and all users of the product instantly receive the changes. Enterprise customers benefit from practically all of the advantages listed above as well as a few others. Primarily, the investment required to utilize cloud services is relatively low. Through the cloud an enterprise customer can access the best technology available without ever investing in physical hardware. (Martinez)

7.10.4 What are the key components?

The high-level structure of cloud computing can be broken down into three parts. The first of these parts is the data warehouse. The data warehouse is the location where the physical data is stored and where the majority of cloud based processing happens. There are many companies that provide cloud-based platforms utilizing data warehouses. In addition to the data center there is the Software-as-a-Service (SaaS) provider or "Cloud User". The SaaS provider develops software that utilizes the power of the data center. This software is then resold to the SaaS user who uses the SaaS product that is built on the cloud platform. There are many variations to this concept, but in general these are the key pieces of the cloud structure. (Armbrust)



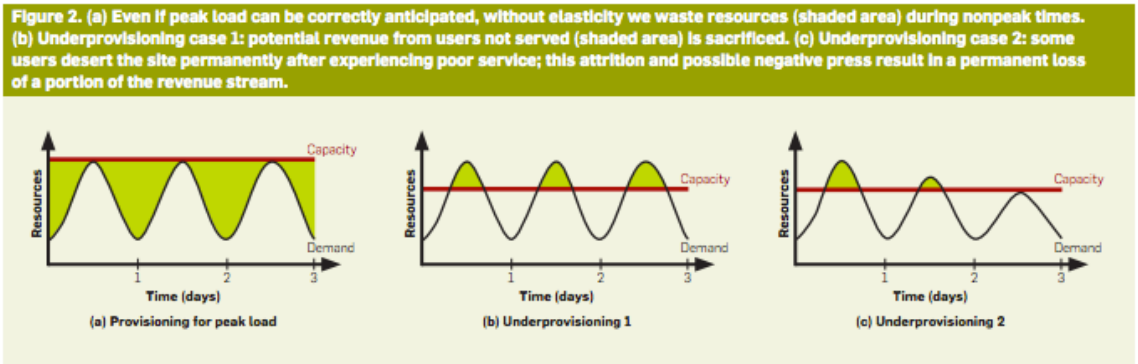
7.10.5 What types of cloud platforms are available?

Within cloud computing there are a variety of development platforms available. Windows Azure, Force.com, and the Google App Engine are just a few of these platforms. Within these platforms a developer can create interactive and dynamic applications that provide incredible levels of performance with minimal maintenance. However, the services offered by each cloud provider do vary significantly. For example Amazons EC2 product allows control much like physical hardware. "Users can control the entire software stack, from the kernel upward" says Armbrust. This level of control makes it difficult for Amazon to offer automatic scaling and failover. Other platforms are domain specific like the Google App Engine that exclusively targets web applications. This provides a greater level of scalability and failover protection because of the increased constraints. Microsoft Azure, on the other hand, utilizes the .net framework to provide more program flexibility to their users. However, Azure does not offer the low level programming functionality offered by Amazons EC2 platform. This makes Azure an intermediate platform, with Amazon and Google at opposite ends of the spectrum. (Armbrust)

7.10.6 Elasticity in Cloud Computing

A key benefit of cloud computing, regardless of the platform chosen, is elasticity. For example, a cloud provider can add or remove resources in minutes or less. In a traditional computing environment this can take months. This results in a better matching of resources to workload and increases the usage of services purchased. Armbrust has outline a terrific example of the cost benefits of this high data utilization:

> *"For example, Figure 2a assumes our service has a predictable demand where the peak requires 500 servers at noon but the trough requires only 100 servers at midnight. As long as the average utilization over a whole day is 300 servers, the actual cost per day (area under the curve) is 300 X 24 = 7,200 server hours; but since we must provision to the peak of 500 servers, we pay for 500 X 24 = 12,000 server-hours, a factor of 1.7 more. Therefore, as long as the pay-as-you- go cost per server-hour over three years (typical amortization time) is less than 1.7 times the cost of buying the server, utility computing is cheaper."*
>
> (Armbrust)



This example may even underestimate the challenges of provisioning as it fails to take into account other factors like the seasonal nature of some services. For example, Bestbuy.com may require more resources as Christmas approaches. This makes it incredibly challenging to accurately predict demand and provision accurately. The elasticity of the cloud in addition to the pay-as-you-go model makes it a highly compelling resource for online business needs. Also, this benefit transfers to entrepreneurial businesses as well. With this elastic model start-ups can purchase a small amount of resources initially and then instantly scale as their business begins to improve. (Ambrust)(Chen)

7.10.7 Developing Software in the Cloud

Agile development methodologies are particularly effective in a cloud environment as they cater to the rapidly evolving nature cloud computing. Agile

software development has no standard definition but does have many qualities generally accepted as the principles. The primary goals of agile methods are to develop software quickly while accommodating frequent changes in requirements. Agile methods generally favor an iterative approach with frequent releases to customers. Also, customers are a crucial part of generating system requirements and their collaboration with developers is encouraged. Continuous improvement is a key feature of agile methods as well. (Greer)

Many of the principles utilized by agile software development can be enhanced through cloud computing. Common Wealth Bank is Australia's leading provider of integrated financial solutions a strong example of the benefits of leveraging agile development and cloud platforms. They utilize more than 300 Oracle databases that have been consolidated into on-demand demand instances. This has reduced lead time for development teams provisioning a production quality environment from 3 months to 2 minutes. Other companies like Salesforce.com have used cloud computing combined with agile development methods to increase the frequency of their product updates. Through cloud computing Salesforce.com is able to maintain a single, unified source for code. This enables globally distributed developers to utilize agile methods to develop software with an iterative and unified approach. These are but two of many examples of how a

business can leverage virtualization and cloud computing to enhance agile development. Cloud computing can enhance agile development in several key ways: (Kannan)

First, as stated by Kannan, "cloud computing provides an unlimited number of testing and staging servers". Traditionally, one physical server is needed on site to test applications relying on server connectivity. Furthermore, multiple servers may be required if the application utilizes multiple servers. This can result in significant development costs and cause delays in production time as resources are provisioned. Through the cloud and virtualization hardware resources like servers and databases can be instantly spun-up resulting in huge time and cost savings for the agile developer. There is no longer a need to wait for physical devices to be configured and developers are free to continue developing software. (Kannan)

Another advantage discussed by Kannan is "It Encourages Innovation and Experimentation." Due to the reduced time and cost required to provision additional resources, developers are more likely to code and test new features. Also, because the resources available are essentially unlimited, developers aren't required to wait for an existing build and testing cycle to end prior to

receiving hardware to test. Resources can be instantly provisioned and are not limited by the hardware on hand. This results in a development team that is more willing to test new ideas due to the reduction in effort required. Ultimately this will translate into more innovative creative development environment resulting in a higher quality software output. (Kanaan)

Kanaan goes on to state that cloud computing "Enhances Continuous Integration and Delivery." As previously discussed, agile methods generally employ an iterative approach. This means that for each version new features are tested and added until the entire software package is complete. Through cloud computing, updates can be pushed to the end user for each iteration without the need to download software and install. Also, because the software is designed to operate on a cloud platform, new software integration and operating system compliance issues are dramatically reduced. (Kanaan)

7.10.8 Obstacles to Consider in Cloud Computing

There are several obstacles to those considering cloud computing. One of the primary obstacles is data lock-in. Currently proprietary API's are used to interact with the data stored on a cloud solution. This means that extracting data can be

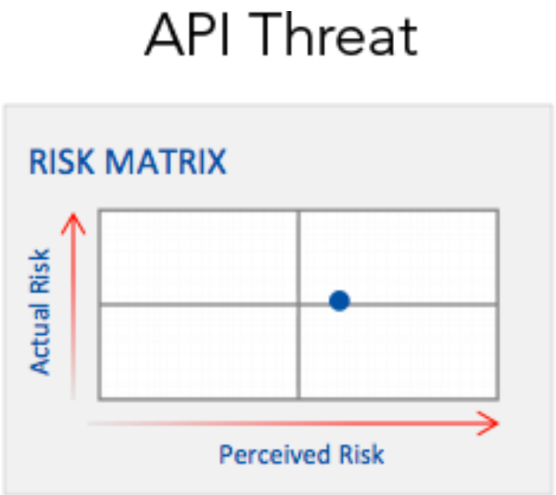
challenging. Even more challenging is extracting data and moving it to another cloud platform because each platform has its own proprietary API's. Also, because of this lock-in, cloud customers are subject to price increases, reliability issues, or even providers going out of business. One example of this occurred in 2007 when the online storage firm LinkUp was shut-down because it lost over 45% of customer data. In response to this LinkUp simply told there customers that their data could not be recovered and they should investigate other storage solutions. An incident such as this could be devastating to an individual or organization. (Hackers)(Mezgár)

7.10.9 Security in Cloud Computing

So now we know how cloud computing applications are developed and how they benefit the end user, but are they safe? Will our data, being constantly shuffled across the Internet, be protected? The security of a given cloud solution is dependent on those who developed it and the security methodologies utilized. Services such as those provided by Google and Microsoft are regarded as quite secure as they have highly experienced security teams supporting them. This is a huge advantage for a small business without the means to have their own dedicated security team. Through cloud computing, small businesses can

leverage the security technologies developed and honed but much larger companies. (Kamal)(Wei)

Data in the cloud is generally more secure than if stored locally. Some argue that the cloud puts all your "eggs" (data) in one basket, but at least it's a heavily protected basket. Also, data stored in the cloud is generally much more redundant as it is automatically backup at multiple sites. This makes data recovery a much easier process in the event that important data is lost. However, as with all data, its security is dependent on the developers. One potential security threat the developers have less control over are the API's used to interface with the cloud platform. If the API's they utilize have security flaws, then the end user is vulnerable. (Notorious)



Cloud based systems are also vulnerable to Denial of Service attacks. A DoS attack causes system resources to be overwhelmed and forces them to go offline. For the end user, this means the cloud service you subscribe to may be unavailable for the duration of the attack. Depending on the duration, the downtime may result in significant financial losses for those depending on the cloud service. (Notorious)

As with most business functions, malicious insiders are a significant security threat to cloud computing. A malicious employee may have the ability to delete crucial data or interrupt business functions. However, this risk can be mitigated with proper security controls and adequate logging. Through this process the business can limit the destructive ability of any employee and thereby reduce this risk significantly. (Notorious)

Another, often overlooked risk to those who utilize cloud computing is due diligence during cloud vender selection. It is the responsibility of the entity purchasing the cloud solution to verify its security. Often those purchasing cloud solutions simply trust the provider because they have heard so much positive publicity surrounding cloud computing. However, this is a significant security risk

because the security of the cloud should be independently verified. As stated in *The Notorious Nine*:

> *"Without a complete understanding of the CSP environment, applications or*
> *services being pushed to the cloud, and operational responsibilities such as incident response, encryption, and security monitoring, organizations are taking on unknown levels of risk in ways they may not even comprehend, but*
> *that are a far departure from their current risks" (Notorious)*

Ultimately there are inherent risks in computing regardless of platform. Many argue to the risks of cloud computing outweigh the benefits, but for most businesses I must disagree. It would be nearly impossible for any small business to supply the same level of security as that of a cloud provider. The investment in human and technological resources required to achieve this would be significantly more expensive than utilizing a cloud vendor. Furthermore, small businesses and individuals don't have the experience provided by cloud services. Cloud providers are exposed to nearly every known security threat,

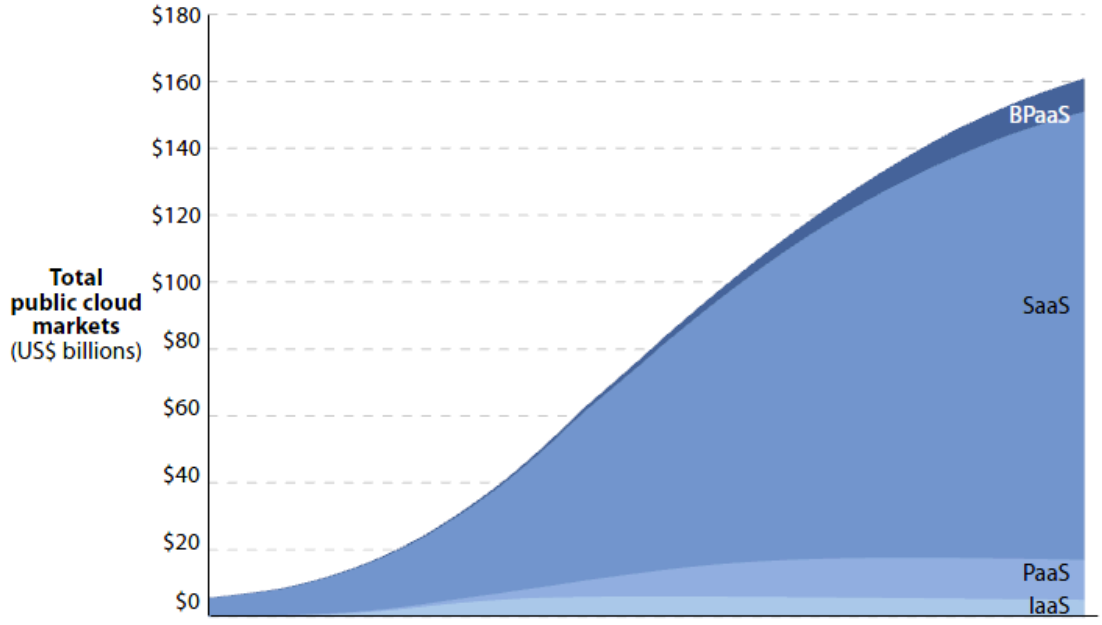
thus making them much more experienced at defending against malicious attacks.

7.10.10 Using the Cloud for Evil

However, cloud providers need not only be concerned with the security of their system, but they also need to protect it from being used maliciously. For example, in 2009 Sony's servers were made unavailable due to a Denial of Service attack that leverage cloud computing. The computing power of the cloud is available to any who wish to access it, therefore hackers can harness this power as well. A German researcher named Thomas Roth utilized Amazon's EC2 cloud service to demonstrate brute force cracking of Wi-Fi passwords. Roth was able to try passwords at the rate of 400,000 a second, which enabled him to break into the network in 20 minutes. These resources are available to anyone at a cost of pennies per hour making this a likely alternative to more traditional hacking methods. (Hackers)

Figure 3 Forecast: Global Public Cloud Market Size, 2011 To 2020

The spreadsheet detailing this forecast is available online.



	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
BPaaS ($)	0.15	0.23	0.35	0.53	0.80	1.26	1.95	2.93	4.28	6.00	7.66	9.08	10.02
SaaS ($)	5.56	8.09	13.40	21.21	33.09	47.22	63.19	78.43	92.75	105.49	116.39	125.52	132.57
PaaS ($)	0.05	0.12	0.31	0.82	2.08	4.38	7.39	9.80	11.26	11.94	12.15	12.10	11.91
IaaS ($)	0.06	0.24	1.02	2.94	4.99	5.75	5.89	5.82	5.65	5.45	5.23	5.01	4.78

58161

Source: Forrester Research, Inc.

7.10.11 Conclusion

Does the cloud provide a substantial competitive advantage and cost savings? For the right business candidate it does. For example, start-ups can benefit heavily by the decreased entrance cost. Furthermore, start-ups can leverage this technology to scale and grow rabidly as their business develops. However, for those heavily invested in traditional computing infrastructure cloud computing

provides less advantages. Upgrading existing infrastructure that has not fully depreciated may simply expose you to many of the risks of cloud computing without the main advantages. Almost more important than deciding whether or not to use the cloud, is choosing appropriate cloud provider. Due diligence is key in this process and should investigate the functionality and security of the platform to ensure it operates as planned. However, when leveraged appropriately, cloud computing undoubtedly provides significant competitive advantage to traditional platforms.