EXPLORING A FIVE-LAYER MODEL TO ACHIEVE BENEFITS OF SEPARATION FOR WEB APPLICATION DEVELOPMENT

A Thesis

Presented to the

Department of Information Systems and Quantitative Analysis

and the

Faculty of the Graduate College

University of Nebraska

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

University of Nebraska at Omaha

by

Zachary D. Fowler

May, 2010

Supervisory Committee

Dr. Paul van Vliet

Dr. Ken Dick

Dr. Stacie Petter

EXPLORING A FIVE-LAYER MODEL TO ACHIEVE BENEFITS OF SEPARATION FOR WEB APPLICATION DEVELOPMENT

Zachary D. Fowler, MS

University of Nebraska, 2010

Advisor: Dr. Paul van Vliet

Abstract

As the sophistication of web applications continue to grow and as these applications start to rival desktop experiences, the complexity involved in creating these online applications is ever increasing. While literature began to support the idea that web applications can be engineered using traditional software engineering approaches, the requirements of producing web applications add speed and flexibility as a factor while maintaining quality. This study proposes the use of a five-layer model based on the separation of concerns found in web application development. The five-layer model leverages traditional software development paradigms such as object-oriented systems, model-view-controller patterns, and layered design. To evaluate the five-layer model, it is applied to a development project as a case study. Interviews were used to explore the usefulness of the model from the perspective of developers and project manager.

To my mother,

and the other 3%.

Acknowledgements

I would like to give my deepest thanks to Dr. van Vliet and my committee members Dr. Dick and Dr. Petter for their patience throughout the process which, at times, seemed like it might never end. I also wish to thank the many people who supported me throughout my studies as a UNO student. The fact that I have been so fortunate to find an area in which I love to work can be directly attributed to their enthusiasm for teaching. I would also like to express my sincere gratitude to Katie Nichols and Karen Stendal for sharing their thesis experiences with me. It was from the examples they set that I found the excitement to pursue this project. Finally, this project would not have been possible without the support of my family and friends. Thank you, ever so much.

Contents

List of Tables

List of Figures

1. Introduction

Can web application development be managed using traditional techniques, or must it be considered an entirely new art form within the project management discipline? Support for either side of the spectrum exists in current literature with most falling to one side or the other, but rarely is there a consensus. Some authors insist that web application development is an extension of current methodologies, while others insist that the speed requirements and volatile nature of the web require a new understanding of development management.

Without discounting either side, this study introduces a five-layer model for web application development that reflects the benefits of both arguments. Design and development practices in the web application arena are predicated on the separation of content, application logic, and presentation to the client. The model builds upon the separation of content, presentation, and application logic, the nominal layers of a web application. Interactive components are separated from the presentation layer and an additional structured markup layer is added between after the application logic layer. By designing web applications within this model, project managers are able to leverage their developers' unique skill sets and experience, enhance parallel development, and provide a foundation for future application development for other media types or services by only requiring the modification of upper layers.

The purpose of this study is to develop the proposed five-layer model and explore its usefulness of for web development based on the separation of concerns, by examining a case study of its application to a web development project in which data is collected by interviewing developers and project management.

To guide the research, the remaining sections of this study follow the design science research activities, as described by Peffers et al. (2006), to introduce and evaluate a proposed five-layer model for web application development as an artifact for information science research. Section 2 provides background in literature for the nature of web development and its unique needs. Section 3 introduces the five-layer model. Section 4 describes the activities and nature of design science research for information science. Section 5 enumerates the research methods used to demonstrate the artifact's implementation in a case study to solve a business problem. Section 6 evaluates the model by presenting the results of structured interviews. Section 7 concludes by summarizing the evaluation of the model and communicates limitations and further areas for research.

2. Literature Review

Over the last few years, the ad-hoc development that was once so dominant in the field of web development faded in favor of repeatable processes and abstractions familiar to traditional software engineering. During this time, a convergence on three web development principles emerged:

1. Methodologies and development must support flexibility and change
2. Quality applications must be developed within shortened time frames
3. The separation of concerns and object-oriented patterns are key

The following sections describe these principles as backed by current literature.

2.1 Flexible Methodologies

In a study on cost estimating techniques for web applications, Mendes, Mosley, and Counsell (2005) summarized factors that made estimating web development difficult, compared to estimating traditional development:

1. There were no standards in the development size of web applications
2. Processes for web development were substantially different from traditional approaches
3. The primary goal was to bring quality applications to market as fast as possible
4. Those involved in web development were typically less experienced programmers

5. Processes in general were ad hoc, though some companies were starting to use agile methods

Three of the five (1, 2, 5) reflect the need for consistency, while the other two (3, 4) are specific concerns of web development. The factors they presented rang true with researchers who agreed that web development started with very ad hoc processes (Altarawneh & El Shiekh, 2008; Mendes et al., 2005; Taylor, McWilliam, Forsyth, & Wade, 2002). Case studies and surveys revealed that the processes companies employed often reflected the unique skill set or knowledge of only a few individual developers (Altarawneh & El Shiekh, 2008, Taylor et al., 2002). Others proposed that the nature of web's ad hoc development stemmed from the internet's early purpose as a document publishing platform instead of an application platform (Gellersen & Gaedke, 1999; Murugesan, Deshpande, Hansen, & Ginige, 2001; Pressman, 2000)—that it just was not meant for complex applications and thus any which were created would be wrought with difficulty. "Standards" for developing web applications were based on experience and, in those days, such experience was hard to come by.

Following Mendes et al.'s (2005) next point, the understanding of whether or not web development was truly different than traditional information systems development was up for debate. With such a new set of technologies and computing paradigms, some questioned if web applications *can* be engineered at all (Pressman, 2000). In addition to introducing the term "web engineering," Pressman proposed that while some software engineering processes might not work well for the web and in practice may be labeled as failures, a solid engineering approach will stand the test of time. In what was deemed a "resurrected tradition of a written academic debate," Kautz, Madsen, and Nørbjerg (2007a, 2007b) and Baskerville, Pries-Heje, and Ramesh (2007) generally agreed with Pressman. Kautz et al. (2007a, 2007b) argued that traditional information systems development had always suffered from the same problems, while Baskerville et al. (2007) focused on the application of newer agile methodologies for the purposes of "internet speed" and shorter life cycles. Other work also suggested that while the platform is new, the underlying problems that challenge designers and managers were fundamentally the same as they were for traditional systems, but only looked new on the surface (Kautz et al., 2007a, 2007b; Lang & Fitzgerald, 2006). The author of this study found this debate to be an interesting one, and agrees with Kautz et al. (2007a), that software engineering principles can truly benefit web development.

Case studies illustrated that the incremental and iterative approaches of the Unified Process could be used for web development (Ambler, 2002; Motschnig-Pitrik, Karagiannis, & Reimer, 2002), but investigative surveys of industry practices by Taylor et al. (2002) and Garzotto, Perrone, Jeusfeld, and Pastor (2003) suggested that few practitioners were actually applying the conceptual models and processes proposed by researchers in their entirety. These exploratory studies suggested that the firms which employed *some* formal processes (if any) did so in a piecemeal manner in order to suit their needs. In terms of speed and flexibility, agile methodologies were thus presented as another panacea for the state of web development methodologies (Altarawneh & El Shiekh, 2008; Baskerville, Ramesh, Levina, Preis-Heje, Slaughter, 2003; Mendes et al., 2005; Reifer, 2002), such that web-specific agile methods like the AWE Process (McDonald & Welland, 2005) were created.

2.2 Concept of Speed

While the convergence on agile methods grew, speed and flexibility have been consistent qualifiers for web development processes and methodologies. To help quantify this, Pries-Heje, Baskerville, Levine, and Ramesh (2004) presented a comparative study of ten companies' practices measured in 2000 and then again in 2002. Key to their findings were five common elements that persisted across company processes before and after the dot-com bubble: parallel development, release orientation, components/reuse, prototyping, and the use of stable architectures (Pries-Heje et al., 2004, p. 40). A surprising outcome of their study was that the focus in 2002 shifted towards quality and away from the throwaway mentality applied to website projects that was rampant pre-bubble in 2000. Before the dot-com bubble, speedy delivery was considered too important to even perform QA testing—the first product to release wins the market. However, as the belt-tightening across companies cinched budgets after the bubble, quality became a shared part of web development measures: quality *with* speed. To these ends, firms needed to rely more on component re-use and parallel development than brute force programming.

2.3 Separation of Concerns, Layering

The concept of separation is not new to programming. According to Frick, Bude Su, and Yun-Jo An (2005), the separation of content and presentation is an important design consideration in any software project. In fact, the separation of concerns is identified as a key practice to facilitating software maintenance (Mens & Wermelinger, 2002) and allows functionality which would normally be developed by different professionals to remain separate in the application code (Ceri et al., 2003, p. 458). Applied to web development, it provides for a foundation for evolution and can maximize design and implementation reuse (Schwabe, Esmeraldo, Rossi, & Lyardet, 2001).

2.3.1 Benefits of Layered Design

Often, the separation of concerns leads to logically defined layers. In this study, the term "layer" generally reflects the instantiation of a separated concern. For example, an application may contain a database layer, logic layer, and an interface layer, each of which is separated from the others such that changing a form element does not require changes to the logic.

Knight and Dai (2002) define the layers of web applications as input, application logic, business logic, and presentation. Fowler's (2002) abstractions name data source, domain, and presentation as fundamental layers of application architecture. Traditionally, web development separation takes place across three layers: application logic, content, and presentation (Briggs, Champeon, Costello, & Patterson, 2004; Hall, 2009).

The benefits of using defined layers and the benefits of separation of concerns are often interchangeable. Layers allow the division of complicated systems into isolated systems, are good for standardization, and allow substitution of alternative implementations of the same services (Fowler, 2002). Specific references to the benefits of presentation separation are discussed in Section 2.3.2.2 below.

Another advantage of layered design is that every developer or designer does not need to possess skill sets for every component of a web application. Programmers may not be experts at design, just as graphic designers may not be knowledgeable in designing database systems. With a layered design it is possible to have developers apply their expertise to produce the best results at each layer.

2.3.2 Supporting Trends in Technology

Several tools, techniques, and technologies have been introduced to enhance the separation between layers of web application architectures. Object-oriented development, specific presentation technologies, and model-view-controller patterns are discussed in the next sections.

2.3.2.1 Object Oriented Development

Objected-oriented practices are known for their ability to create reusable components through abstraction. When applied to web development, these principles lead to a successful separation of concerns (Gellersen & Gaedke, 1999; Knight & Dai, 2002; Jazayeri, 2007), and modular design (Schwabe et al., 2001; Knight & Dai, 2002; Jazayeri, 2007; Hall, 2009).

Gellersen and Gaedke's early paper in 1999 (Gellersen & Gaedke, 1999) criticized the lack of such abstractions in web development at the time:

> *"Obviously, the lack of abstraction hampers construction of general components and frameworks. Code from user interface elements cannot be separated from code for page layout" (p. 64).*

Today, however, object-oriented programming is supported by almost all web application programming languages. As an example, reusable database classes like PHP Data Objects (PDO) (http://php.net/manual/en/book.pdo.php) provide a common query interface for several database products. Application code does not need to be rewritten if the underlying database product is changed when using these reusable classes.

2.3.2.2 Presentation Technologies

Accessibility, device independence, and mobile adaptation are often cited as benefits of separating presentation from content (Briggs et al., 2004; Hall, 2009; Laakko & Hiltunen, 2005; Yates, 2005; Zhang, 2005). Specifically, ever since the introduction of Cascading Style Sheets (CSS), web developers have been able to leverage the separation of presentation from the HTML pages for these purposes. CSS files typically define presentational elements such as fonts, colors, and layout. One CSS file can be used across many HTML documents, so that a change in one stylesheet will affect all attached HTML files simultaneously. This can save bandwidth by reducing the amount of markup required (Briggs et al., 2004) and allows website maintainers to update the look of many pages at once, saving time and effort. Alternate stylesheets can be created which adapts the presentation to alternate media types such as a screen reader or mobile browser.

Content adaptation is an automated form of technology that applies transformations to content in order to better suit end device presentation. Servers that perform content adaptation can also save bandwidth by stripping the amount of markup and content sent to handheld or mobile devices (Hall 2009; Zhang, 2007). The markup used for desktop-based browsers is often bloated with extra functionality or ads compared to content suited for mobile devices. Content adaptation performed for mobile devices will segment pages into meaningful chunks of information. More requests may be required to receive the entire page, but the amount of markup sent to the device will be far less per request (Zhang, 2007).

2.3.2.3 Model-View-Controller Patterns

Separation between the content, logic, and presentation layers is often cited as necessary for large, enterprise application development (Fowler, 2002; Knight & Dai, 2002). Web development has benefited from another technology for separation which is rooted in traditional software development. The model-view-controller (MVC) pattern separates an application into models (objects that represent data), views (display states of the information), and controllers that manipulate the models based on user input and update the views (Fowler, 2002). Web developers have realized MVC patterns can apply to web applications, and a number of MVC frameworks are available for various languages (Jazayeri, 2007).

Table 1 - Three-layer Web Application Model mapped to MVC Components

Web Application Layer	Model-View-Controller Component
Content, or Data	Model
Application Logic	Controller
Presentation	View

Benefits of separation and layered principles come from implementing an MVC framework. In MVC frameworks such as CodeIgnitor, CakePHP, KohanaPHP, Ruby on Rails, and Apache Struts, each component of the MVC pattern is a contained in a separate file so developers can edit them independently and in parallel. This creates a hard, physical separation between each layer of a web application. Table 2 shows how three-layer web applications can be mapped to an MVC approach.

2.4 Motivation for Further Study

Arguing that web development is an immature discipline and without direction would be difficult today. In the previous sections, we have seen that the need for speed and flexibility is a persistent quality which makes web application development unique and challenging. By using traditional software engineering techniques such as object-oriented design and the separation of concerns, web application developers can leverage parallel development and re-use in order to shorten development times and increase quality. By *designing* web applications based on the layered separation of concerns, it is this researcher's belief that project managers can leverage their developers' unique skill sets and experience to enhance quality and timely development cycles, and provide a foundation that supports application flexibility. Though the separation of concerns is a well talked about phenomenon in web application development, few studies have examined the practicalities of separation from the *perspective of developers and managers*.

Additionally, as quickly as web application development has matured, web applications today are more than just tools to retrieve organized information. The data that drives these applications can be consumed in formats which require no direct presentation to the user (e.g., XML-based web services, or RSS feeds). Furthermore, web applications (which are often considered Web 2.0 sites) encourage user participation and interactivity with rich UIs that rival those of desktop applications. A three layer model is no longer sufficient. The accepted layer model needs to be expanded to support data consumption and the interactivity of rich user interfaces in order to embrace a true separation of concerns.

This research therefore presents an updated five-layer model for web application development, as an artifact of design science research activities, which naturally aligns with object-oriented development, presentation technologies, and model-view-controller patterns. The research explores the model's effectiveness by designing and developing a web application based on the five layers.

3. An Expanded Five-Layer Model for Interactive Web Applications

Web applications have traditionally been divided into content, application logic, and presentation layers. This three layer model is as an evolution from the days when web pages were contained in one file. Originally, web pages were static; all parts of a page were coded into one file. During the first evolution and the introduction of CSS, presentation was separated from the content. As content sources became flexible and web sites relied on databases, application programming logic was used to retrieve content and build HTML dynamically, thus resulting in a second evolution. With the growing popularity of web-based applications and database-driven content, web services and XML-based data feeds allowed computer-to-computer consumption of dynamic content. The application logic layer was able to produce alternate forms of structured markup (e.g., HTML or XML) from the same content. Though this evolution is not usually referenced, alternate forms of markup are widely used in practice for RSS feeds and WSDL documents for Web Services. In this researcher's experience, however, web applications today do more than just *display* information or provide data feeds, so another evolution of the model is necessary for the complete separation of concerns.

Using the established division between content, application logic, and presentation layers as a foundation, this study introduces a five-layer model for web application development that separates the behavioral and interactive concerns of an application from the presentation. Figure 1 illustrates the progression of web applications from a static, single file web site, to the five-layer model. From the accepted three-layer model and observed four-layer model, the five-layer model (also depicted in Figure 2) divides the presentation layer into presentation and interaction components.

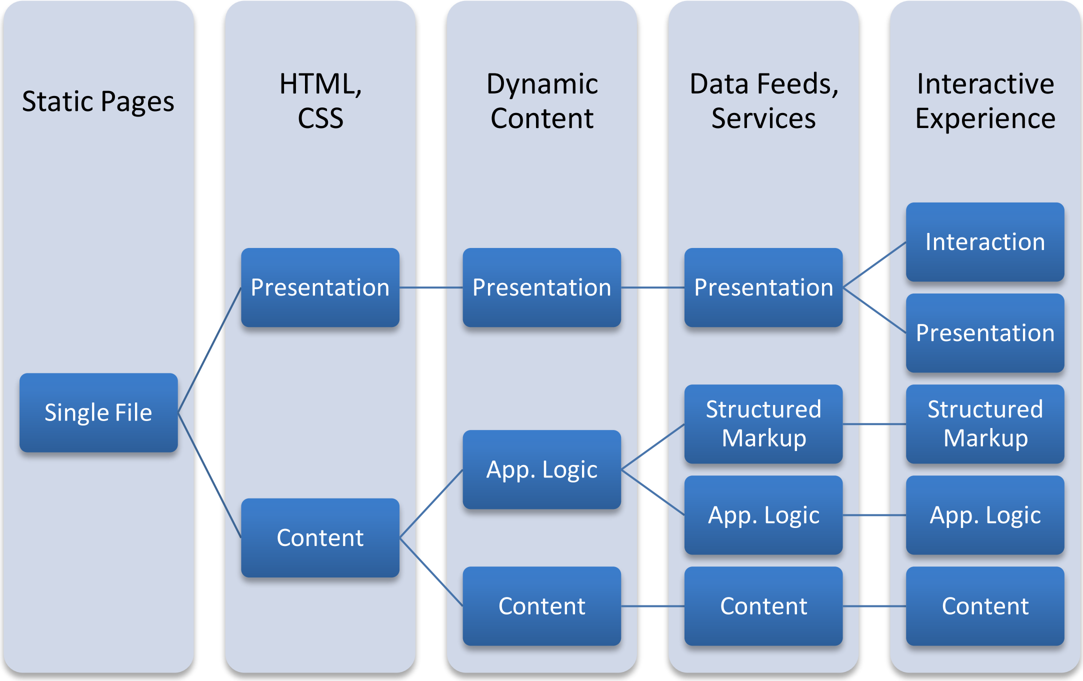


Figure 1 - Evolution of Layered Web Applications

The following sections describe each layer in detail, and propose design elements which should be produced to facilitate separation among developers. A basic understanding of web technologies such as HTML, CSS, JavaScript, and XML will be helpful in these sections. For each layer, required “input” and “output” elements which will make separation successful are noted.

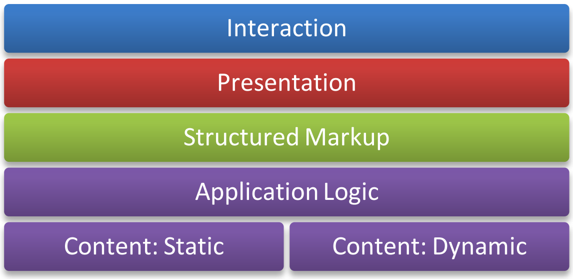


Figure 2 - Five-Layer Model for Web Applications

3.1 Content

Content that will be used by the application or presented to the user of the website belongs in its self-titled base layer. Figure 2 shows two types of content: static and dynamic content. Static content does not require state information and is usually in the form of text, images, video, documents or even XML documents such as RSS feeds or iCalendar (ICS) files.   The location and format of such content is always known to the application ahead of time as this is an early design decision. Static data is usually used but rarely modified by the application.

Content which can be queried or requires state information (such as application state or user state) is considered dynamic. Dynamic data is usually stored in databases and is generally updated or maintained be the application itself. The application itself may alter dynamic data or it may be altered by a trusted third-party data provider.

Web applications make generous use of both types of data, but should access them in a consistent and abstracted manner. Objects are most appropriate for building data-abstraction layers, and they become models in an MVC pattern. The application logic layer only needs to be aware of the public methods and properties, not any underlying code. Object abstraction also benefits static data such as pictures. As the application grows, images can be moved from the local file system to a content delivery network, and modifications to the image-retrieving class files never affect the application logic code.

3.1.1 Design Documentation Considerations

In website design, one of the most common ways to store dynamic content is in a database. Entity Relationship Diagrams (ERDs) are an excellent way to model the *relationships* and cardinalities between data objects. When modeling the content layer, however, class diagrams and object documentation (public methods, properties, descriptions) can also go a long way to supplement ERDs, and provide working class names for application programmers. Data designers may also find CRC cards useful in modeling the data domain.

3.1.2 Specific Input and Output Expectations

In order to separate the content layer from the application logic layer, developers should have a full understanding of the application domain and its data elements. These can be obtained from requirement documents, paper-based forms that need to be converted into electronic format, and knowledge of the customer processes.

When fully designed, a set of data documents that describe the available classes and their public methods and properties should be provided to the application layer developer. Additionally, if static files are to be used, the naming conventions or URL patterns which locate them should also be documented.

3.2 Application Logic

This layer is the heart of any website application. Developers work at this level to ensure the application meets the business requirements of the system. The application logic layer produces structured markup which will be provided to the browser for presentation and interaction processing, or to another system for consumption via XML-based technologies.

3.2.1 Design Document Considerations

This layer deals with programming logic more than any other layer. Use cases show goals and activities that the logic layer must handle and provide great artifacts for developers. Detailed descriptions that go along with the use cases provide the set-by-step logic while the use cases themselves present the overall situation in which these steps are applied.

3.2.2 Specific Input and Output Expectations

In order to program the application logic, the developer must have access to all design documents. They must also be aware of which data types are required by the structured markup files and templates (Views, in the MVC pattern).

As output to the structured markup layer, the application logic layer provides a list of variable names and data types per defined view to the structured markup developer.

The application logic developer also provides a URL structure for the web application, as URL structure is often determined by application architecture.

Fowler (2002) presents various enterprise application architectures which can be applied to web applications (MVC, Front Controller, Page Controller), but each may require a different URL structure. For example, the Page Controller pattern (Fowler, 2002, pp. 333-343) may be used if a website organizes its application logic into logical pages that use physical file hierarchies:

/books/view.php?id=32

/books/list.php

/cards/view.php?id=2241

/cards/list.php

whereas the Front Controller (Fowler, 2002, pp. 344-349) will handle all requests for a website with one director file, negating the need for physical file hierarchies:

/app.php?section=books&action=list

/app.php?section=cards&action=view&id=2241

The URL structure differs depending on which application architecture is applied, and thus must be documented here at the layer which handles URL request routing: the application logic layer.

3.3 Structured Markup

Structured markup provides application data in a format that is understandable by browsers, or formats data in a way that can be understood and consumed by third party services. Most frequently, this is in the format of HTML[1] or a variety of XML, depending on the endpoint device.

XML documents are typically used to structure data in a way that it can be interpreted by other computing devices. The structure of data has little to no bearing on the presentation to the user and therefore only requires development of the bottom three layers of the five-layer model. The HTML structured markup used by browsers is meant to be presented to the user, and thus requires the additional presentation and interaction layers. Example instantiations of the five-layer model for these scenarios are shown in Figure 3. The transparent "Interaction" and "Presentation" layers signify their absence when designing XML for consumption as a service.

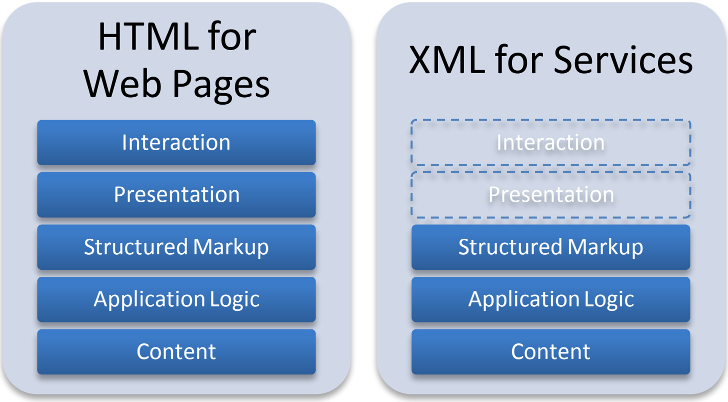


Figure 3 - HTML Stack vs. XML Stack in the Five-Layer Model

When planning structured markup for web pages and HTML, designers must consider page elements such as headings, navigation, content areas, multimedia

[1] For the purposes of this paper, references to HTML are inclusive to its counterpart which conforms to XML syntax, XHTML.

sections, and forms. Markup for XML data should be planned so that it conforms to a document type definition (DTD) or other internal and documented standard.

3.3.1 Design Document Considerations

Web applications are often designed using prototypes, screen mockups, or wireframes. Based on these documents, block diagrams for major sections should be identified so that semantic markup can be applied. The block diagrams can be modeled using WebML (http://webml.org) tools or developed into functional website wireframes and prototypes using software such as Microsoft Visio (http://office.microsoft.com/en-us/visio/default.aspx) or Axure RP (http://axure.com/). In several literature cases, prototypes and ad hoc diagrams were most commonly used to graphically model markup (Amant, 2005; George, 2005; Shucha, 2003; Steele & Carter, 2001), though wireframe tools seem to be gaining in popularity on the web (Kahn, 2010) and for information architecture design (Rosenfeld & Morville, 2002).

3.3.2 Specific Input and Output Expectations

The structured markup developer takes variable names and data types from the application logic layer and applies them as variables in the HTML or XML templates.

The developers at this layer must also create the foundation for presentation layer elements by defining class names and IDs for page elements and tags, as well as specify the semantic markup for headings, content areas, multimedia sections, and forms. Ultimately, this layer will produce the actual HTML or XML markup which will be sent to the browser or end device.

3.4 Presentation

The presentation layer is where transformations of the markup are performed so that the content is presented in a visually appealing and organized way. Closely coupled to the structured markup layer, the presentation dictates how content contained in structured markup will be shown to the end user.

The presentation layer artifacts represent the first user-centered design element of this model.

3.4.1 Design Document Considerations

There is very little work in literature which suggests how to *best design* the presentation of web pages. Adobe Photoshop has been the tool of choice for most designers (Burdman, 1999, p. 29), and as noted in the literature review, ad hoc prototypes using paper sketches, whiteboards, wireframes and mockups are commonly used. Full color mockups can provide direction to designers just as sketches or mockups can provide layout information. As alternative CSS files can be used to alter the presentation of structured markup to various devices or media types, documents which indicate the differences between them may be helpful.

3.4.2 Specific Input and Output Expectations

Presentation developers receive CSS-specific documentation as defined by the structured markup layer. Class and ID names for each of the HTML sections of a page are specified so that the presentation developer can apply the appropriate font, color, imagery, and position styles to the elements.

The presentation layer developers produce CSS files associated to structured markup for each required media type (e.g., screen, handheld, or print).

3.5 Interaction

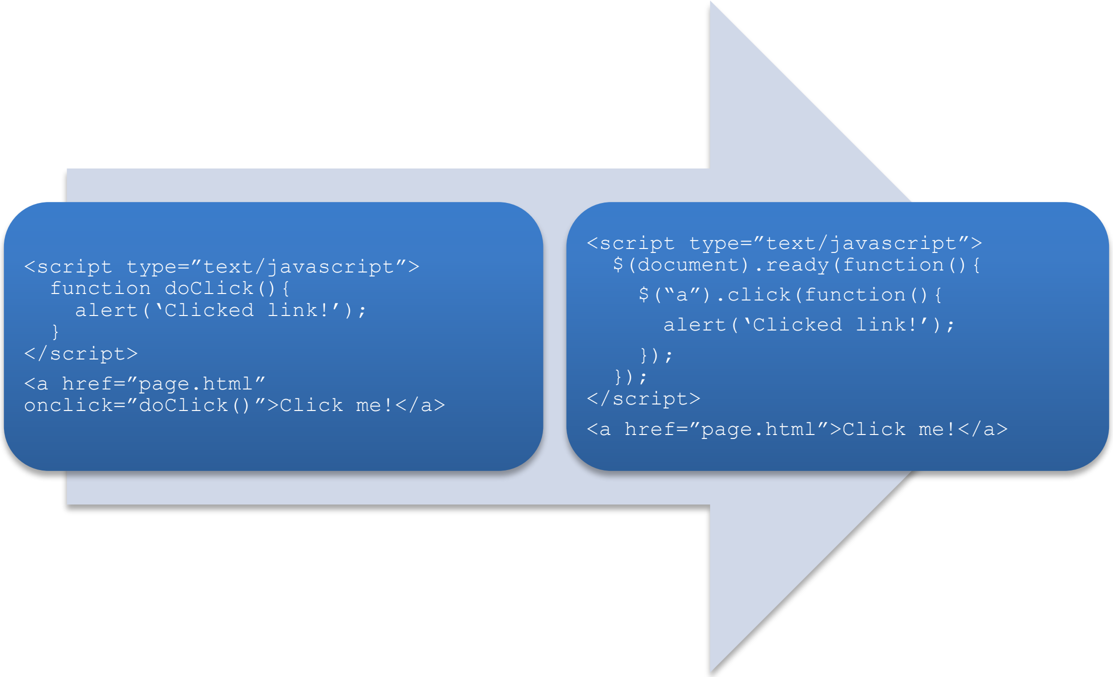
Any event that changes the state of the current presentation can be classified as interaction. The distinction between navigating and interacting is that navigation describes movement between pages of the application while interaction only modifies the current screen. As the technologies of client-side scripting have advanced over the years, the need to separate interaction from what was once only considered presentation must be addressed.

3.5.1 Separation of Interaction Concerns from Presentation

Dynamic HTML (DHTML) uses client-side JavaScript code to manipulate page elements, or parts of the Document Object Model (DOM), without requiring web server interaction. DHTML was often used to script rollover images or expanding menus on a web page by modifying the DOM in real-time. Unfortunately, scripting was tightly coupled with structured markup during the early days of DHTML. As browsers supported different features in an effort to gain market share, it was difficult for developers to create code that would behave consistently across all platforms. Additionally, not all versions of client-side scripting languages like JavaScript or VBScript had the same feature set, so some scripts which would work in Internet Explorer would fail in Netscape, Mozilla, and Safari browsers, or vice versa.

As user interface innovation on the web using Asynchronous JavaScript (AJAX) enables developers to build web applications that rival the rich experience of desktop applications (O'Reilly, 2007), frameworks such as Prototype (http://www.prototypejs.org/) and jQuery (http://jquery.com/) provide developers with a consistent platform for scripting across all major browsers. They support standard cross-browser functionality and graceful degradation so that developers spend less time creating browser-specific code. These frameworks often use CSS-style selectors to apply scripts and behavior to DOM elements, taking advantage of a technology already familiar to web developers and designers.

Furthermore, as seen in Figure 4, these frameworks allow developers to apply functions and event-handlers to DOM elements in a way that can be completely separated from markup. Event handlers such as "onclick" do not have to be hard-coded into the markup for buttons or links, as seen on the left half of Figure 4, in order to perform client-side interactive scripting. Instead, they are applied to DOM elements using framework supported CSS-style selectors as shown on the right half of Figure 4. If for not any other reason, these frameworks are the key to separating UI interaction from the presentation of a page.



```
<script type=”text/javascript”>
  function doClick(){
    alert(‘Clicked link!’);
  }
</script>
<a href=”page.html”
onclick=”doClick()”>Click me!</a>
```

```
<script type=”text/javascript”>
  $(document).ready(function(){
    $(“a”).click(function(){
      alert(‘Clicked link!’);
    });
  });
</script>
<a href=”page.html”>Click me!</a>
```

Figure 4 - Traditional Embedded JavaScript (left) and jQuery-based JavaScript (right) applied to a click event

3.5.2 Design Document Considerations

Application requirements for behavioral functionality may be described using story boards, flow charts, or state charts. Animations, mouseover effects, or other UI interactions are difficult to describe using current tools, so textual descriptions of these items may be the best way to convey the desired level of interaction.

If AJAX requests are made to modify areas of page content, request/response information should be described so that application logic and structured markup layers can prepare URLs and response data.

3.5.3 Specific Input and Output Expectations

At the interaction layer, developers require documentation of HTML structure and CSS class and ID names so that website behavior can be mapped to

DOM elements. This is provided by the structured markup layer in combination with the presentation layer outputs.

The output of this layer is a file, or series of files, which use client-side scripting to *add* interaction to the structured markup in a way which is unobtrusive to the markup.

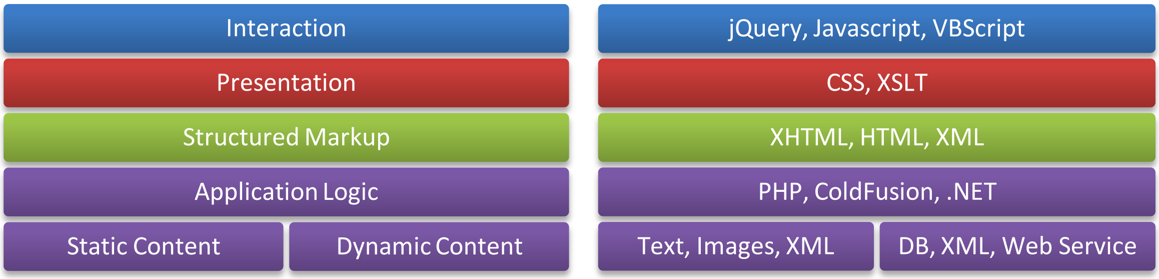
3.6 Practical Applications

When considering the design for the five-layered model, the MVC framework allows for file-based separations of each layer. Note that Structured Markup, Presentation, and Interaction layers are all part of the MVC "view" component, but can be defined in individual files.

Table 2 - Five-layer Model for Web Applications Mapped to MVC Components

Web Application Layer	Model-View-Controller Component
Interaction	View (e.g., JS Files)
Presentation	View (e.g., CSS Files)
Structured Markup	View (e.g., HTML or XML)
Application Logic	Controller
Content, or Data	Model

The five-layer model for web applications discussed in this study is not specific to HTML, CSS, and JavaScript. It can be applied to other web technologies such as Flash or XML+XSLT. Figure 5 gives examples of technologies which can be used at each layer. Some technologies, such as XML, can be used at multiple layers, depending on the use.



Interaction
Presentation
Structured Markup
Application Logic
Static Content | Dynamic Content

jQuery, Javascript, VBScript
CSS, XSLT
XHTML, HTML, XML
PHP, ColdFusion, .NET
Text, Images, XML | DB, XML, Web Service

Figure 5 - Example Technologies for Each Layer of the Five-Layer Model

To summarize the design considerations and input/output expectations for each layer, Table 3 is provided as a reference. The foundational layers are listed first in the table, followed by the higher level layers which augment them.

Table 3 - Reference for Five Layers to Design Documents, Input and Output, and MVC Component

Layer	Design Documents	Requires	Produces	MVC Component
Content	Class Diagrams, CRC Cards, ERDs	Application Requirements, Domain knowledge, Forms	Object Class Files, Public methods and properties to access and change data, URL patterns for static content	Models
Application Logic	Application Requirements, Use Cases with Step-by-step descriptions,	Object documentation, Needs of presentation templates	Variable/type definitions for presentation templates, URL structure for application	Controller
Structured Markup	Block level diagrams, mockups	Variable/type definitions for content provided by application logic	HTML or XML files with named CSS classes and ID values	View (HTML)
Presentation	Mockups, prototypes	Class and ID names from templates, HTML structure	CSS files to apply to templates	View (CSS)
Interaction	Storyboards, flowcharts, state charts, textual descriptions	Class and ID names from templates, HTML structure	JavaScript files to add interaction	View (JS)

4. Design Science Research

To test the usability of the proposed five-layer model, traditional research activities such as comparative case studies and questionnaires would not be appropriate. An alternative approach is necessary to establish the validity of new tools as they are presented within a discipline. Hevner, March, Park, and Ram (2004) identifies two such paradigms for information science research: behavioral science and design science. Behavioral science research seeks to create or justify theories which predict organizational and human phenomena within the information system domain of knowledge, while design science research seeks to create and evaluate IT artifacts which are intended to solve identified organizational problems (Hevner et al., 2004, pp. 75-76). The differences between behavioral and design science information systems research are described as complimentary and inseparable. Behavioral science research builds truth by developing and justifying theory. Most quantitative theory-building studies found in information science research follow behavior science research patterns. Design science research on the other hand builds utility through constructing and evaluating artifacts designed to meet business needs.

Building on this work, Peffers et al. (2006) propose six activities for producing information science research using a design science process. The activities are summarized in the Table 4.

Table 4 - Design Science Activities (Peffers et al., 2006)

Design Science Activity	Description
1. Problem Identification and Motivation	Define the problem and justify the value of a solution. Knowledge of the state of the problem and importance of its solution is required for this activity.
2. Objectives of a Solution	Objectives for the solution are inferred from the problem definition and can be qualitative (a new artifact which presents solutions not already explored) or quantitative (the solution is measured as better than current alternatives). Knowledge to the efficacy of current solutions is required for this activity.
3. Design and Development	The artifact is created here in the form of constructs, models, methods, or instantiations by determining its functionality and architecture. The actual artifact is created in this activity. Knowledge of theory which can move objectives to design and development is required in this activity.
4. Demonstration	Demonstrate the efficacy of the artifact to solve the problem by experimentation, simulation, case studies, proofs, or other activity. Knowledge of how to use the artifact to solve the problem is required for this activity.
5. Evaluation	Observe and measure how well the artifact supports the solution by comparing objectives to the actual observed results. It may be necessary to iterate back to step 3 to improve the effectiveness of the artifact.
6. Communication	The problem and its importance, the artifact, its design, and effectiveness must be communicated to relevant audiences.

The research in this study is based on the design science research process with the intent of producing a five-layer model as an artifact for web development. The following sections address how the research meets each of the six activities of design science research identified by Peffers et al. (2006).

4.1 Problem Identification and Motivation

Section 2, the Literature Review, provides background for the problem: demanding quality "internet-speed" application development with enough flexibility for change. The separation of concerns through current development paradigms offers a solution to this problem, but few studies have addressed the feasibility of such separation. The nominal three-layer approach to separation does not support the expanding amount of interactivity found in web applications of the Web 2.0 era.

4.2 Objectives of a Solution

As shown in the literature review, the separation of concerns provides for a possible approach to solving the business problem of designing and rapidly building high-quality flexible web applications. The objectives of the design science research artifact should therefore support the separation of concerns via object oriented development, presentation technologies which allow the separation of presentation from content, and MVC patterns which provide a clear division between the components of an application.

4.3 Design and Development of the Artifact

The actual design and development of the artifact, the five-layer model for web application development, was described in detail in Section 3. The evolution of the model from traditional web application models was described, followed by a detailed description of how the artifact can be implemented. The practical applications describing how the model can be applied to meet the objectives stated above concludes the section.

4.4 Demonstration

Demonstration of the artifact is done by using it in a web application project as described in the research design and methods of Section 5.

4.5 Evaluation

Based on post-study interviews, the usefulness of the artifact from the perspective of developers and project managers is addressed in Section 6. Results from interviews compare the perceived objectives to actual observations in applying the artifact to solve a business problem.

4.6 Communication

This study itself, and specifically the results and conclusions presented in Section 6 and 7 respectively, communicate the model’s validity to the web application development research community.

5. Research Design and Methods

To demonstrate the utility of the proposed artifact to solve a problem, a case study was developed in which the five-layer model could be applied to a web application development project.

5.1 Strategy of Inquiry

A case study was chosen as the strategy of inquiry for this research because of its ability to explore processes, activities, and events (Creswell, 2003, p. 183) as a qualitative approach of study. Since design science research commonly introduces a new artifact and evaluates its efficacy, a case study that employs the artifact is most appropriate for demonstration and evaluation.

5.1.1 Case Selection

In order to study the usefulness of a five-layer model for web development, case study participants must possess specific skills in web development which include any of the following: database design, object-oriented programming, specific web technologies and programming (HTML, CSS, PHP, JavaScript), or graphic design. Participants who possess these skills will be able to provide the most relevant feedback on usefulness of the proposed model.

5.1.2 The Project

Based on professional and personal connections, the researcher identified a group of web developers available at the University of Nebraska at Omaha's College of Information Science and Technology. After contacting the project manager via email and finding them to be agreeable to such a project, the researcher met with the project manager to discuss the nature of the five-layer model. The researcher helped build documentation for a pre-existing project so that it could be used to evaluate the proposed model. This included class diagrams and ERDs, use cases with step-by-step descriptions, and wireframe documentation.

A portfolio project highlighting the group's work was chosen for the case study. The web application centered on publishing previous development work done by the group, and share information about the team members themselves. Once a team of five developers had been selected by the project manager, the researcher asked all members, including the project manager, to review and sign consent forms relating to the post-implementation interview. All potential subjects were made aware that their participation in interviews was voluntary, and they could ask to be withdrawn at any time. Potential subjects were also made aware that their participation would not affect their employment nor directly benefit them in any way. This research has been approved by the Institutional Research Board (IRB) as Category 2 Exempt research, #198-10-EX. A copy of the exempt status notification is included as Appendix A.

This particular project was well suited for research as it had components to address each part of the five-layer model: dynamically driven content from a database (projects, team members), use of static content such as photographs and text, and application logic that builds on an abstracted object model for data access to create structured markup. Presentation of the site content included stylized graphics and positioning, while an interaction layer enabled website visitors to seamlessly browse the portfolio. Additionally, back-side management components were built to update stored object information, and an XML-based RSS data feed was built to explore alternate output at the structured markup layer. The “front-side” included four main pages of information, and the back-side used Create, Read, Update, Delete (CRUD) , and List modules for each main object type. The project was complex enough to present each layer, but simple and familiar enough to the team that the information domain did not require additional training.

To support development based on the separation of concerns and object-oriented paradigms, the project team used the KohanaPHP MVC framework to develop objects as models, structured markup as views, and contain application logic in controllers. External Cascading StyleSheets and client-side jQuery tools were used for the presentation and interaction layers respectively.

The project manager assigned tasks to a developer at each layer, based on their skills and experience. The researcher is unaware if the details of the input and output considerations of Section 3 were used to further guide development. Informal communication among team members was encouraged throughout the project.

5.2 Data Collection

At the conclusion of project development, the researcher interviewed each developer and the project manager separately to collect and record their opinions and observations from applying the five-layer model to the project. If the researcher received permission, the interview for each participant was recorded using a digital recording device. All participants granted permission to have their interview digitally recorded. During the interview, the researcher asked the questions provided in this Appendix B, and posed follow-up questions when needed to elaborate on participant responses.

5.3 Measurement

Post-implementation interviews were conducted using closed- and open-ended questions to gain qualitative insight into the use of a five-layered model. Conclusions were drawn based on Creswell's (2003, pp. 191-195) six steps of qualitative data analysis:

1. Organize and prepare data for analysis.
2. Obtain a general sense of the information.
3. Code material into chunks or categories.
4. Describe and interconnect themes which organize the major findings.
5. Advance how the description and themes will be presented.
6. Interpret the meaning of the data.

Steps 1 and 2 were done by transcribing the interviews and reviewing them to get a general sense of ideas or themes. A tentative list of themes was produced from these steps. By summarizing the main points of each response and placing these into a grid, the researcher began to find connections between them. Steps 3 and 4 were performed repetitively by enumerating the connections until the major findings were identified and organized. To complete the final two steps of qualitative data analysis, the resulting themes were identified and interpretations given in narrative form throughout Section 6.

5.4 Outcomes

By analyzing the interview data, the researcher intended to evaluate the usefulness of the five-layer model as a new artifact for web application design and development. The following section presents the results of the evaluation.

6. Results

Based on the coding techniques of Section 5.3, data collected from participant interviews revealed several trends and groupings. Responses were coded and grouped into the following categories:

- Opinions on the validity and usefulness of the model
- Effects of using a new method for the first time
- Differences compared to previous development styles
- Difficulties found in the model's practical application

A note to the reader: To protect the identity of responses, interview participants are referenced by a number. The number does not correspond to the layer in which an individual was assigned nor their role in the project (developer or project manager). Moreover, any reference to a specific person, a person's name, or their gender in the interview text was replaced by keys and gender-neutral pronouns. The keys "Dev 1," "Dev 2," etc., were used to reference developers, where the number corresponds to the assigned layer. Because the project manager has unique insight into the overall development process and was asked alternate questions during the interview, the key "PM" was used.

Also for the reader: The actual verbiage used by the interviewee was maintained as much as possible, even with grammatical mistakes or poor sentence structure, as the interviews were conducted in person as a conversation. As necessary, additions which clarify the meaning of statements are enclosed in square brackets.

6.1 Opinions on the Validity and Usefulness of the Model

All participants were asked the question, “Do you feel the separations are useful?” and “How do you believe using the five-layer model can affect the speed at which web applications are developed?”

All participants stated that they felt the separations of the model were in fact useful. Several used the words “more efficient” to describe overall productivity, or other speed-related words to describe the collaborative work. As one participant stated,

> *“[I]t can allow for very quick implementation of a web site that’s very clean, well styled, very interactive with complete database and everything all wrapped up in one.” (Interview Participant 1)*

Others specifically agreed that the speed was positively affected by using the model.

> *“So start up time is a little bit, but then it goes really fast because everyone knows what they’re doing, everyone’s getting it done.” (Interview Participant 2)*

> *“It’s easier to divide work between people, and it’s easier to integrate, so I think this is faster than working like previously, [in] the old style.” (Interview Participant 4)*

> *“I think it will take less time if we follow this five layer architecture, compared to any architecture.” (Interview Participant 5)*

> *"Because with everybody doing a different piece of it, one person isn't doing everything, you don't necessarily have to wait until someone has to stop." (Interview Participant 6)*

In terms of usefulness to a project team, one of the proposed benefits of separation is the ability to take advantage of an individual's skill sets and apply them to a specific layer. The participants of the study also found this to be true.

> *"The interaction [person] was really really good at doing [their] part, so [their] stuff was done very quickly. The style [person] was really good at doing [their] part, so we were able to hone in on everybody's experiences, or their experience." (Interview Participant 6)*

> *"And if people do what they are good at, the application would be good. I think that is the positive thing. Well, yeah, and you can't expect every single body to do everything, so separation is good within layers." (Interview Participant 5)*

> *"Also, I think it's good to—it's nice to have five different people who have a forte in whichever layer they may be placed working together, again, makes it more efficient than say myself if I don't have a lot of experience with coding, or something like that. It's going to take me a lot longer to figure out how to solve a problem, or to learn about different aspects that need to be implemented, whereas having somebody who is more of an expert right from the get-go allows things to move along a lot more quickly." (Interview Participant 1)*

Expanding on this idea, Interview Participant 6 also stated that the quality can improve as developers with certain abilities can be used for each layer separately.

> *"Instead of having one person that's really good at programming but not that good at design, for programming they work really good and be really efficient, but the design they may not, or vice versa. So having somebody do each layer could increase the quality." (Interview Participant 6)*

One participant stated that because they were *not* selected to work in a certain layer, the project was better off, due to their lack of knowledge in the other layers.

> *"And other layer, like, JavaScript and jQuery layer or the graphics layer, or the programming layer, well, those two layers like JavaScript and design layer would have completely not suited me. This project wouldn't have been completed by now if I was in that layer." (Interview Participant 3)*

The idea that developers were assigned to work in layers that matched their skill sets resonated through all responses.

6.2 Effects of Using a New Method for the First Time

Though participants found the division of work and separation between layers useful, they were clear in stating that they had not seen or used a five-layer model before in this way. There was a period of adjustment before they could understand the entire project. This was the first time such a highly collaborative project was taken on with such a clear division of labor.

> *"I have never heard of the five layer model, so it was something completely new." (Interview Participant 1)*

> *"So I think if everyone's experienced and knows what they're doing and worked on projects before, I think there will be very little startup time. But like this is our first project we did separated like this, and everyone wasn't necessarily, with the layers around them, wasn't necessarily knowing too much about the layers around them, of how the work is done. " (Interview Participant 2)*

> *"At first it was like pretty hard to understand how to work." (Interview Participant 4)*

> *"The first thing I want to say is as I was following this five layer for the first time, so I faced a few difficulties communicating with my teammates." (Interview Participant 5)*

> *"As I am working on this architecture for the first time, I was facing some problem regarding the model part." (Interview Participant 5)*

Another reported that using the new framework itself posed challenges, when asked if they encountered difficulties.

"I think the framework, let me say, because I didn't understand it at the beginning, it was a little difficult." (Interview Participant 3)

Interestingly, the same respondent also stated that after a team would apply the five-layer model the first time, subsequent projects could be completed even faster.

"The learning curve in the beginning can get really high, because you have to make employee know what the thing is, the five layer system. After that, it can go really fast because they are expert at what they are doing and if good coordination is there within the team members it can really excel." (Interview Participant 3)

6.3 Differences Compared to Previous Development Styles

Participants were asked how the new approach was different from their current methods. In addition to using the model as a new tool for development, the responses by participants converged on the idea that work was not separated like this in the past. This was the first time they were responsible for only one piece of the puzzle.

"Well usually I'm working by myself mostly, or have been so far. So this is the first time really, really truly collaborating in real time with somebody else." (Interview Participant 1)

"I normally am assigned to a project, … . If there is problem with JavaScript, I do it. If there is problem with HTML, I do it; if there is

> *problem with graphics, I do it. With the help of project manager, there are only two person. So this is completely different thing than from what I'm doing." (Interview Participant 3)*

> *"Normally I work, if I have to work on a page, in the first there will be database connection and stuff, on the second there will be designing and stuff, and then there will dynamic stuff like PHP. In this case, there is one controller that will be all working with logic and programming. There will be models that will be controlling databases and other HTML views will be controlling all those views. Rather than combining everything on one page, it's differentiated." (Interview Participant 4)*

> *"How I normally work, is maybe one or two people pretty much do the whole thing. … It was different because we were able to separate everything, and we were able to put people, who their skills were honed in on that one particular area." (Interview Participant 6)*

One participant revealed that the *concept* of separation could be found in previous projects, but that it was never executed with such formality (Interview Participant 6).

Another participant stated that it while it was difficult to transition to the separated model at first, it was "obviously useful" and that once they got used to it, developers would "hate to work like previously," using their previous approaches (Interview Participant 4).

6.4 Difficulties found in the Practical Application

The five-layer model was derived as an evolution from the traditionally layered web architectures supported in literature. As defined by the design science research activities used to evaluate the model, the objectives of the proposed solution must support speed and flexibility through the separation of concerns and supportive trends in technology. Overall, the data gathered from participant interviews seem to positively affirm the usefulness of the five-layer model. However, practical difficulties in applying the model to a project were consistently identified in the same data. The difficulties or issues which received the most attention were the sequencing of tasks, the stronger need for coordination and communication, and specific concerns relating to the separation of the structured markup layer from presentation and interaction layers.

6.4.1 Sequencing

The experiences on sequencing and the practical dependencies between layers affected each participant's perspective of their work. Interview Participant 1, for example, found that some parts were dependent on others enough that the "lag" kept the team from being consistently productive.

> *"I think that's what needed to happen, is a lot of things: more of the other four layers needed to come together before the jQuery could kinda be put on top and work with the other layers." (Interview Participant 1)*

Other participants echoed that idea when asked about dependencies.

> *"Also another negative is that it was sorta expected that we all start at the same time. Where I could see eventually … where one needs to start a little bit earlier." (Interview Participant 2)*

> *"OK, maybe I was waiting for the directions, and maybe I was not doing the work because I was not given the things that I needed. I think that may have reduced a little bit of productivity." (Interview Participant 3)*

> *"In some areas yes, especially between the HTML and the styling, kinda getting things to work: big dependencies between the controller, between the programming and the views. The views could not be really done much with, until the controller person produced that. … Waiting for somebody else to finish their part before you could continue, would be probably the biggest [drawback]" (Interview Participant 6)*

6.4.2 Coordination and Communication

Because of the clearly defined separation between team member's responsibilities, many participants stated that communication and coordination were extraordinarily significant to support the split activities.

> *"The positive factor, what I told you earlier only, that the tasks will be separated into various layers so one can concentrate on a specific task. But the negative part is that we have to be more well communicated, well versed, with each and everyone because I am coding for the controller part, and I have to communicate with those guys who will be coding the view part and who will be doing the database stuff. If*

> *communication is not there, then we can't do the project successfully. And, that is the only thing I feel that is negative about this five layer structure." (Interview Participant 5)*

> *"I don't think that there will be any such problem or so if the team members are well communicated with each other, and if they know their task well, then I don't think there will be any such problem working on this architecture layer." (Interview Participant 5)*

When asked if the five-layer model can positively or negatively affect the quality of web application, several respondents stated that poor communication will negatively affect quality.

> *"That can be negative factor too, coordination. If I am doing [it] myself, we don't need coordination. If people are around, you need good interpersonal skills within employees. So I'll take that as a minus point because coordination." (Interview Participant 3)*

> *"The only thing is that if communication doesn't happen correctly, the wrong things could get done." (Interview Participant 6)*

Relating to the actual application of the five-layer model, Interview Participant 5 expressed that because this was the first time working with the separations, communication was extremely important.

> *"The first thing I want to say is as I was following this five layer for the first time, so I faced a few difficulties communicating with my*

> *teammates. ... So there I faced some kind of problem. The problem was not of coding, but the basic problem was about the communication part." (Interview Participant 5)*

Additionally, communication was the deciding factor on whether or not to judge how the five-layer model positively or negatively affected productivity.

> *"The thing is that, I can say that there is a hair line difference between positive and negative part, and that difference is treated due to communication." (Interview Participant 5)*

6.4.3 Separation of Structured Markup from Presentation and Interaction

During interviews, it became clear to the researcher that the separation between structured markup and the presentation and interaction layers proved to be very difficult for developers. The developer in charge of building the structured markup was able to create valid markup, but had little experience with CSS. The presentation layer developer found that the structured markup developer's lack CSS of knowledge negatively affected their own ability to apply CSS elements to the HTML: the HTML needed to be modified in order to support the presentation layer aspects. The presentation layer developer normally works with HTML and CSS in combination. They believed that the developer assigned to the structured markup layer knew HTML well, but did not know how to prepare it for the proper use of CSS.

> *"If the HTML person understood more about CSS and how you align things, it would go a lot easier." ... "The HTML person needs to know*

> *what's appropriate for DIVs and etc. for aligning things." (Interview Participant 2).*

The interaction layer developer found similar problems.

> *"Sure, well I guess for instance, I know [Dev 4] was waiting on [Dev 3] to finish up some of the coding, er some of the HTML, so [they] could apply the styles to it. I mean [they were] ready to go but [they] couldn't do anything until [Dev 3] was done, and I couldn't do anything until [Dev 3] was done as well, with the classes and IDs and that kind of thing." (Interview Participant 1)*

Because the interaction layer developer was also well versed in structured markup, they found it difficult to not have full control.

> *"Uh, it was tough not to change some of the HTML, or add CSS styles, and that sort of thing." (Interview Participant 1)*

According to the interview participants, the project manager stepped in at this time to assist. They provided support and assigned another developer to work with the structured markup.

At the end of the study, when asked about ways to improve the model, the project manger suggested that the presentation layer responsibilities be divided into graphics and layout. The graphical components would be handled by the presentation layer developer, while the layout components would be developed by the same person who creates the structured markup. The dependencies between layout and the structured markup were considered too tightly wound to unravel from each other.

6.5 Other Results

Not all results gathered from the interview could be grouped into the major category themes, yet were found to be interesting by the researcher.

Knowledge of Nearby Layers

As proposed in the definition of the five-layer model, it was stated that the application could be better developed by assigning individuals to each layer based on their expertise. According to the interview results, the idea of placing an "expert" into the relevant layer is promising and valid. However, while they may not be experts in other areas, they should at least be experienced and familiar with the needs of nearby layers. For practical applications, while an expert in interface development may not need to know how the CSS-specific elements or HTML structure were created, they should at least be familiar with the technology to fully understand how other layer's output affects their own layer's input.

Amount of Effort for Application Logic Perceived to be Greater

One participant believed the application logic layer required much more effort than the other layers.

> *"Well, I think the separation of layers is good, but I think if you are doing the big system, I think for one graphic designer or one HTML programmer, you need to have two web programmers. That's what I*

> *think, because I know programming is zero or one. Either it comes up or it doesn't." (Interview Participant 3)*

They went on to explain that all layers are affected by the controllers and the programming logic, and therefore will require more effort even for small projects.

Large vs. Small Projects

Several participants suggested that the five-layer model would be beneficial for large scale projects. One participant in particular did not believe the five-layer model would be appropriate for smaller, "brochure sites," or sites which did not require back end programming.

7. Conclusions

In this study, a five-layer model for web application development was designed, demonstrated, and evaluated as an artifact using design science research activities. By exploring the data collected through post-study interviews, the researcher found the artifact to be valid, but it must be developed further to address some of the practical challenges which were discovered.

The increased separation between developers' work, combined with the fact that most developers usually are in control of entire web projects, requires that the collaboration and communication between all parties is maintained at an exceptional level. While work *can* be done in parallel, individuals cannot start their work until elements from adjacent layers are at least defined and communicated among team members. Additional layer-by-layer documentation may need to be designed *before* work can begin instead of *during* development. The experts assigned to each layer should be involved in this phase of project design in order to take advantage of their skill sets.

Furthermore, the dependencies between the structured markup layer and the presentation and interaction layers appear to be stronger than the connections between other layers. The developer who is responsible for the structured markup used by higher layers needs to not only be an expert at structured markup, but also know how it must meet the technical needs of the presentation and interaction layers.

7.1 Limitations

Several limitations to the research were made apparent during interview analysis. Since this was the first time such a formally separated model was used by the team of developers, many of the results spoke to the confusion and learning curve that comes from trying something new. Developers who were used to working on their own, for example, were placed into a highly collaborative project when they were not used to communicating so frequently with other team members.

Moreover, the project manager was as unfamiliar with the nature of the separation as developers, and thus has learned more about how to appropriately select team members for each layer. The lack of experience in applying such a formally separated model revealed "learning curve" as an unexpected variable which should be addressed in future quantitative work.

This was the first project to apply the five-layer model to a web development application, so the results and conclusions at this point are based on a very limited sample.

7.2 Future Work

Future research in the area of applying a separation of concerns to development projects can be conducted using this exploratory case study as a foundation. In order to take advantage of the separation prescribed in this model, quantifiable measurements to establish the qualifications of an "expert," as applied to a particular layer, must be investigated. The research in this study found that having a skill set that matches the layer may not be the only requisite; certain levels of knowledge in the other areas may be necessary for the best practical implementation. An explanation of how much knowledge is required would guide managers in selecting team members.

While input and output expectations for each layer were suggested, they were not formally used in project development. Ad hoc communication and informal updates served as the team "documentation" instead. Collaborative tools that support real-time updates to wireframes and other visual documentation may assist developers in the future.

This study explored the application of the five-layer model in one project. In order to create the most effective tool for rapid and flexible application development, modifications to the model should be validated and applied to additional web application projects, in organizations that are familiar with separation as well as organizations which normally create smaller applications.

Once the tool is applied to additional web application development projects, further work can be completed to quantitatively validate its usefulness by comparing the five-layer approach to traditional approaches.

8. References

Altarawneh, H., & El Shiekh, A. (2008). A theoretical agile process framework for Web applications development in small software firms. In *2008 6th International Conference on Software Engineering Research, Management and Applications (SERA)*. IEEE.

Amant, K. S. (2005). A Prototype Theory Approach to International WebsiteAnalysis and Design. *Technical Communication Quarterly*, *14*(1), 73-91.

Ambler, S. W. (2002). Lessons in Agility from Internet-Based Development. *IEEE Software*, *19*(2), 66.

Baskerville, R., Pries-Heje, J., & Ramesh, B. (2007). The enduring contradictions of new software development approaches: a response to 'Persistent Problems and Practices in ISD'. *Information Systems Journal*, *17*(3), 241-245.

Baskerville, R., Ramesh, B., Levina, L., Pries-Heje, J., & Slaughter, S. (2003). Is Internet-Speed Software Development Different? *IEEE Software*, *20*(6), 70-77.

Briggs, O., Champeon, S., Costello, E., & Patterson, M. (2004). *Cascading Style Sheets: Separating Content from Presentation, Second Edition* (2nd ed.). friends of ED.

Burdman, J. R. (1999). *Collaborative Web development*. Addison-Wesley.

Ceri, S., Bongio, A., Fraternali, P., Brambilla, M., Comai, S., & Matera, M. (2003). *Designing data-intensive Web applications*. Morgan Kaufmann Pub.

Creswell, J. W. (2003). *Research design: Qualitative, quantitative, and mixed methods approaches* (2nd ed.). Thousand Oaks, CA: Sage Publications, Inc.

Fowler, M. (2002). *Patterns of Enterprise Application Architecture*. Addison-Wesley Professional.

Frick, T., Bude Su, & Yun-Jo An. (2005). Building a Large, Successful Website Efficiently through Inquiry-Based Design and Content Management Tools. *TechTrends: Linking Research & Practice to Improve Learning*, *49*(4), 20-31.

Garzotto, F., Perrone, V., Jeusfeld, M., & Pastor, O. (2003). On the acceptability of conceptual design models for Web applications. In *Conceptual Modeling for Novel Application Domains. ER 2003 Workshops ECOMO, IWCMQ, AOIS, and XSDM. Proceedings (Lecture Notes in Comput. Sci. Vol.2814)*. Place of Publication: Berlin, Germany; Chicago, IL, USA. Country of Publication: Germany.: Springer-Verlag.

Gellersen, H. W., & Gaedke, M. (1999). Object-oriented web application development. *IEEE Internet Computing*, *3*(1), 60–68.

George, C. A. (2005). Usability testing and design of a library website: an iterative approach. *OCLC Systems & Services*, *21*(3), 167-180.

Hall, C. A. (2009). Web presentation layer bootstrapping for accessibility and performance. In *Proceedings of the 2009 International Cross-Disciplinary Conference on Web Accessibililty (W4A)* (pp. 67-74). Madrid, Spain: ACM.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *Mis Quarterly*, 75–105.

Jazayeri, M. (2007). Some trends in web application development. In *2007 Future of Software Engineering* (pp. 199–213).

Kautz, K., Madsen, S., & Nørbjerg, J. (2007a). Continuing the debate: a response to a response – persistent problems and practices in Information Systems Development as enduring contradictions of new software development approaches? *Information Systems Journal*, *17*(3), 247-249.

Kautz, K., Madsen, S., & Nørbjerg, J. (2007b). Persistent problems and practices in information systems development. *Information Systems Journal*, *17*(3), 217-239.

Khan, S. (2010, February 3). Guidelines, Tools and Resources For Web Wireframing. *W3 Avenue*. Retrieved April 19, 2010, from http://www.w3avenue.com/2010/02/03/guidelines-tools-and-resources-for-web-wireframing/.

Knight, A., & Dai, N. (2002). Objects and the Web. *IEEE software*, 51–59.

Laakko, T., & Hiltunen, T. (2005). Adapting Web content to mobile user agents. *Internet Computing, IEEE*, *9*(2), 46-53.

Lang, M., & Fitzgerald, B. (2006). New Branches, Old Roots: A Study of Methods and Techniques in Web/Hypermedia Systems Design. *Information Systems Management*, *23*(3), 62-74.

McDonald, A., & Welland, R. (2005). Agile Web engineering (AWE) process: perceptions within a fortune 500 financial services company. *Journal of Web Engineering*, *4*(4), 283.

Mendes, E., Mosley, N., & Counsell, S. (2005). Investigating Web size metrics for early Web cost estimation. *Journal of Systems and Software*, *77*(2), 157-172.

Mens, T., & Wermelinger, M. (2002). Separation of concerns for software evolution. *Journal of Software Maintenance*, *14*(5), 311-315.

Motschnig-Pitrik, R., Karagiannis, D., & Reimer, U. (2002). Employing the Unified Process for developing a Web-based application - a case study. In *Practical Aspects of Knowledge Management. 4th International Conference, PAKM 2002. Proceedings (Lecture Notes in Artificial Intelligence Vol.2569)*. Place of Publication: Berlin, Germany; Vienna, Austria. Country of Publication: Germany: Springer-Verlag.

Murugesan, S., Deshpande, Y., Hansen, S., & Ginige, A. (2001). Web engineering: A new discipline for development of web-based systems. *Web Engineering, Software Engineering and Web Application Development*, 3–13.

O'Reilly, T. (2007). What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. *Communications & Software Strategies*, (65), 17-37.

Peffers, K., Tuunanen, T., Gengler, C. E., Rossi, M., Hui, W., Virtanen, V., & Bragge, J. (2006). The Design Science research process: a model for producing and presenting information systems research. In *Proceedings of the First International Conference on Design Science Research in Information Systems and Technology (DESRIST 2006)* (pp. 83–106).

Pressman, R. S. (2000). What a Tangled Web We Weave. *IEEE Software*, *17*(1), 18.

Pries-Heje, J., Baskerville, R., Levine, L., & Ramesh, B. (2004). The high speed balancing game: how software companies cope with Internet speed. *Scandinavian Journal of Information Systems*, *16*, 11-54.

Reifer, D. (2002). Ten Deadly Risks in Internet and Intranet Software Development. *IEEE Software*, *19*(2), 12.

Rosenfeld, L., & Morville, P. (2002). *Informatin Architecture for the World Wide Web* (2nd ed.). Sebastopol, CA: O'Reilly & Associates, Inc.

Schwabe, D., Esmeraldo, L., Rossi, G., & Lyardet, F. (2001). Engineering web applications for reuse. *IEEE Multimedia*, *8*(1), 20–31.

Shucha, B. (2003). The circle of life: managing a law library Web site redesign project. *Law Library Journal*, *95*(1), 47-67.

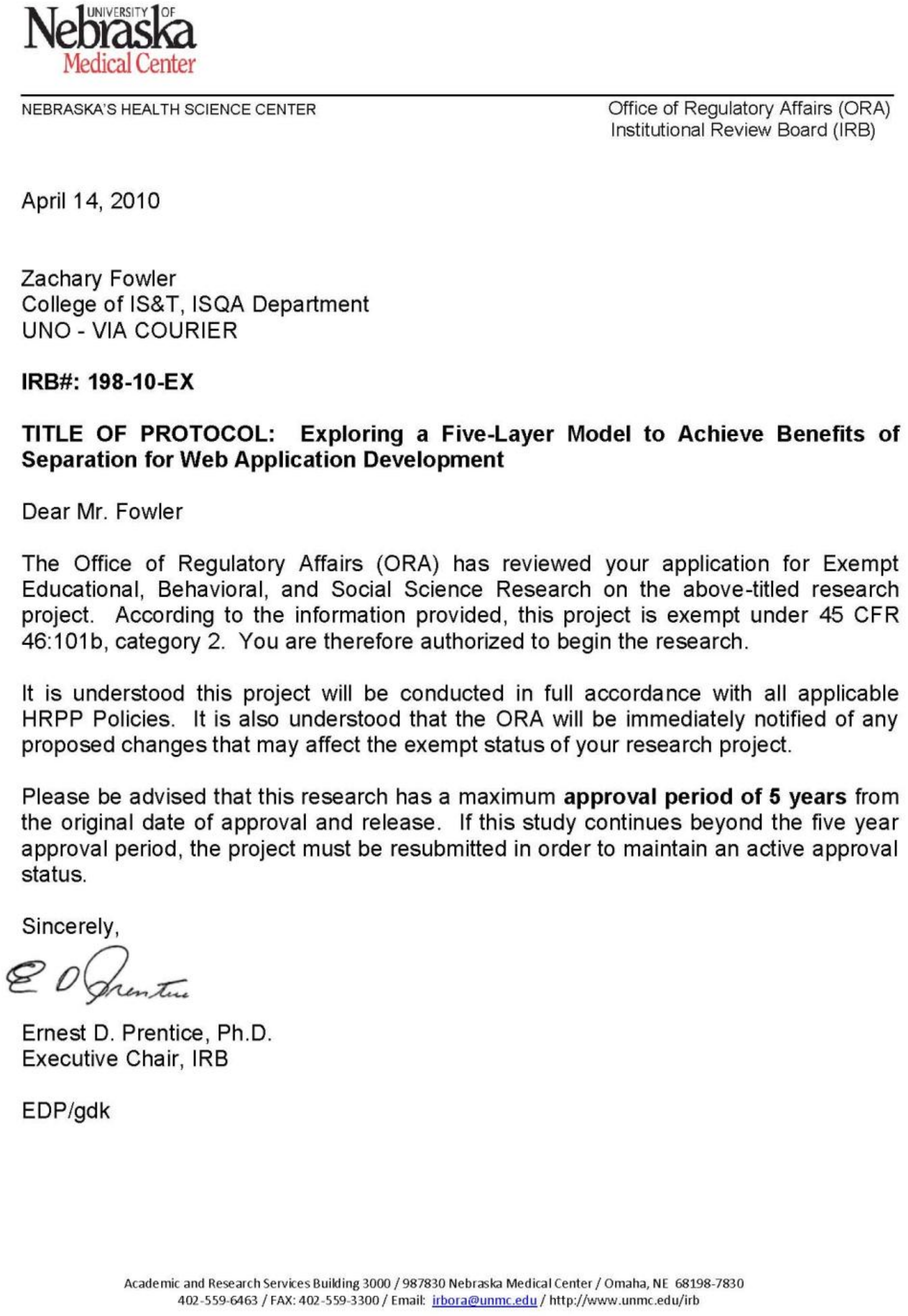
Steele, A., & Carter, R. (2001). Designing process: taming the Web development cycle. In *25th Annual International Computer Software and Applications Conference. COMPSAC 2001*. Place of Publication: Piscataway, NJ, USA; Chicago, IL, USA. Country of Publication: USA.: IEEE IEEE Comput. Soc IEEE Comput. Soc.

Taylor, M. J., McWilliam, J., Forsyth, H., & Wade, S. (2002). Methodologies and website development: a survey of practice. *Information and Software Technology*, *44*(6), 381–391.

Yates, R. (2005). Web site accessibility and usability: towards more functional sites for all. *Campus-Wide Information Systems*, *22*(4), 180-188.

Zhang, D. (2007). Web content adaptation for mobile handheld devices. *Commun. ACM*, *50*(2), 75-79.

Appendix A – IRB Approval



Nebraska UNIVERSITY OF
Medical Center

NEBRASKA'S HEALTH SCIENCE CENTER

Office of Regulatory Affairs (ORA)
Institutional Review Board (IRB)

April 14, 2010

Zachary Fowler
College of IS&T, ISQA Department
UNO - VIA COURIER

IRB#: 198-10-EX

TITLE OF PROTOCOL: Exploring a Five-Layer Model to Achieve Benefits of Separation for Web Application Development

Dear Mr. Fowler

The Office of Regulatory Affairs (ORA) has reviewed your application for Exempt Educational, Behavioral, and Social Science Research on the above-titled research project. According to the information provided, this project is exempt under 45 CFR 46:101b, category 2. You are therefore authorized to begin the research.

It is understood this project will be conducted in full accordance with all applicable HRPP Policies. It is also understood that the ORA will be immediately notified of any proposed changes that may affect the exempt status of your research project.

Please be advised that this research has a maximum **approval period of 5 years** from the original date of approval and release. If this study continues beyond the five year approval period, the project must be resubmitted in order to maintain an active approval status.

Sincerely,

E D Prentice

Ernest D. Prentice, Ph.D.
Executive Chair, IRB

EDP/gdk

Academic and Research Services Building 3000 / 987830 Nebraska Medical Center / Omaha, NE 68198-7830
402-559-6463 / FAX: 402-559-3300 / Email: irbora@unmc.edu / http://www.unmc.edu/irb

Appendix B – Interview Guide and Script

The following pages give closed- and open-ended interview questions which are to be asked of study participants at the end of the development period. At no time may the interviewer ask the name of the interviewee such that it is recorded.

INTRUCTIONS TO INTERVIEWER

1. Request to digitally record the interview. Start recording if the interviewee allows it. Prepare to take notes if the request to digitally record the interview was denied.
2. Read the welcome statement below to the interviewee.
3. Read each question in its entirety to the interviewee, taking notes as appropriate.
4. At the completion of the interview, read the thank you statement and thank the interviewee for their time.
5. Stop recording.

WELCOME STATEMENT

Good [morning/afternoon/evening]. As part the study on web development using a five-layer model for separation, you have been asked to participate in this post-study interview. The interview is expected to last approximately 30-60 minutes. You will be asked a series of closed- and open-ended questions relating to your experiences on the project. You may choose not to answer any question, or stop the interview at any time. Your identity will be kept confidential and not used in presenting the results of this study. We will begin when you are ready.

GENERAL QUESTIONS

1. Explain the separations of the five-layer model, as you understand them, from your experience with this project.

2. Do you feel the separations are useful? Can you elaborate?

3. Explain how the way you normally work was different because of using the five-layered model.

4. What do you believe are the negative and positive factors in developing web applications using this five-layer model?

5. How do you believe the five-layer model can positively or negatively affect the quality of web applications?

6. How do you believe using the five-layer model can affect the speed at which web applications are developed?

PROJECT MANAGER QUESTIONS

Only ask these questions to the project manager.

1. Do you feel the five-layer model is compatible with the way you or your team develops web-based applications?

2. In what ways do you believe the separations of the five-layer model were compatible with your current methods?

3. In what ways do you believe the separations of the five-layer model were incompatible with your current methods?

4. How do you believe the separation positively or negatively affected the management of the project?

5. Describe any benefits you observed in using the five-layer model.

6. Describe any drawbacks you observed in using the five-layer model.

7. Would you attempt to use this five-layer model for web development projects in the future? What would you change?

DEVELOPER QUESTIONS

Only ask these questions to developers.

1. In this study, you were asked to work in one layer only. Please explain the difficulties you encountered because of this.

2. How well do you feel that your knowledge and skillsets matched the needs of the layer in which you developed?

3. How do you feel that using the separation of five-layer model positively or negatively affected your ability to be productive? Please elaborate.

CLOSING QUESTIONS

1. What other thoughts do you have about the model and its separation?

2. Is there anything else relating to the study that you would like to add or discuss in this interview?

THANK YOU STATEMENT

Thank you for your time and answers in this interview. Your participation in the study is complete at this time. Thank you.

[Stop recording]