Software Product Line Architectures, Libraries, Frameworks, and Technologies for Web-based Applications

Mehta, Rutul
A20476293
rmehta29@hawk.iit.edu
Department of Computer Science
Illinois Institute of Technology,
Chicago, US

Sharma, Sukanta
A20472623
ssharma32@hawk.iit.edu
Department of Computer Science
Illinois Institute of Technology,
Chicago, US

Sudharshana, Vidya
A20472468
vsudharshana@hawk.iit.edu
Department of Computer Science
Illinois Institute of Technology,
Chicago, US

Abstract— **The advanced portable world with its huge number of clients who are completely interconnected and constantly surfing the web, a series of new problems have arisen to fulfill the hunger for online services. Web packages that might be complicated client-server packages that run best on net browsers have emerged as a possible answer to this problem. They are built on a standard MVC architecture that lacks feature richness, modularization, and application structure. Web applications are needed to have dynamic substance with particular and versatile web application improvement methods. We propose to use a feature-rich, hierarchical WMD architecture as the base of the web application to develop which provides modularity and aids with maintainability issues. In this paper, we are discussing Frameworks, technologies, Version control, and project management. We propose and analyze some common solutions to web development. At long last, in the final segment, we will look at some of the recent developments in web technology that help to improve security, reliability, and availability.**

***Keywords*— Web Application, Software Product Line Engineering (SPLE), Product Domain Model or Domain-Driven Design (DDD), Dependency Injection (DI), Cross-Platform**

1. Introduction

A Web application (Web app) is an application program that is stored on a remote server and delivered over the Internet through a browser interface. Web applications commonly have short improvement cycles and can be made with little advancement groups. JavaScript, HTML5, and Cascading Style Sheets are used to build most Web applications (CSS). These languages are often used in client-side programming to help construct an application's front-end [4]. To build the scripts that a Web app would use, server-side programming is used. In server-side programming, languages like Python, Java, and Ruby are often used. Software Product Line can be defined as a type of software engineering method, tools, and technique used to create a set of software-reliant systems. The use of Software Product Line Architecture has greatly improved the quality of software. Software Product Line Architecture enables the creation of products and specifies the quality attributes that go with them. Software product lines are made up of components and recycled assets that are essential to the product's functionality [4].

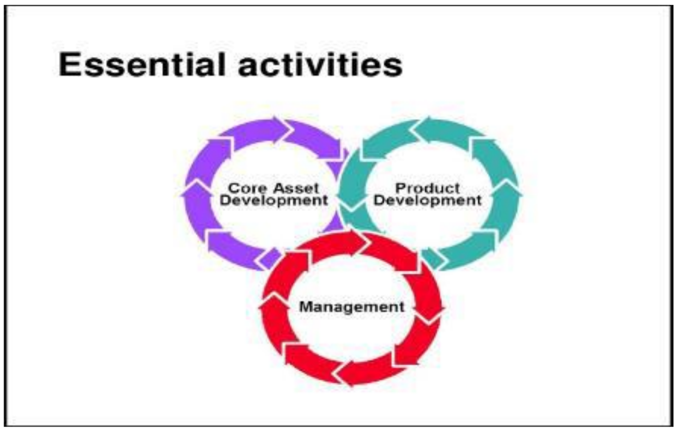


Figure 1: The three essential activities involved in an SPL (source: http://www.sei.cmu.edu/plp/frame_report/PL.essential.act.htm) [4]

The typical architecture of the web application is represented in figure 2, the applications which communicate with HTTP/ HTTPS (Secure HTTP) are categorized as a web application. These applications usually have a client and a server. The dedicated computers having high computing power is called a sever which provides various services to the client. The client sends the HTTP request to the webserver, webserver process the request and gives appropriate output in the form of the requested formats like XML, JSON, or HTML.

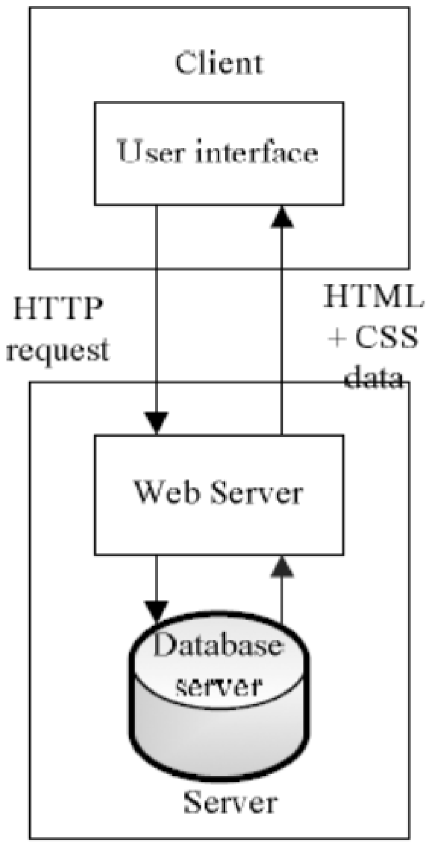


Figure 2:Web-based Application [4]

2. PRODUCT LINE ARCHITECTURE

A software product offering normally comprises a Product Line Architecture (PLA), a bunch of reusable artifacts, and a bunch of items got from the common resources. Software product lines have arisen as another new and critical software product development worldwide. A software product offering is a bunch of programming concentrated frameworks sharing a typical, oversaw set of highlights that fulfill the necessities of a specific market section or domain and that are created from a typical arrangement of center resources in an endorsed way.

The Software Engineering Institute (SEI) characterizes the Product Line as [3] "A Software Product Line (SPL) is a bunch of software concentrated frameworks that share a typical, oversaw set of highlights fulfilling the particular requirements of a specific market portion or mission and that are created from a typical arrangement of center resources in a recommended way."

Items inside a software product family are regularly evolved in stages that will in general be asynchronous, for example, domain engineering and concurrently running application engineering, respectively [1].

2.1. Domain Engineering:

Domain engineering includes, among others, recognizing shared traits and contrasts between item relatives and executing a bunch of shared programming relics (for example components and artifacts) so that shared traits can be abused financially, while simultaneously the capacity to differ the items is saved. During this stage, a variety of new artifacts are planned, and a bunch of variations is related to every one of them. Work results of the software designing interaction are programming segments, reusable and configurable necessities, examination, and configuration models, etc. All in all, any reusable work item is alluded to as a reusable resource.

2.2. Application Engineering:

During application engineering, individual products are gotten from the product family, built utilizing a subset of the common programming ancient artifacts. On the off chance that essential, extra, or substitution item-specific assets or artifacts might be made. In this stage, every variation point, as defined in the past stage, is bound to a specific variation, chosen from the arrangement of variations related to it.

However, for Web application development, we can develop a variety of products only via domain engineering without using application engineering.

To solve the issue, we propose another designing methodology for Software Product Line Engineering (SPLE) for a Web application. We defined a model of domain engineering as a Domain-driven design (DDD) that expresses individual products, using a UML-based metamodel with notations of variability. At runtime, dependency injection technology is used to inject the logic of the product, known as "Instance Product". Domain-driven design (DDD) is an input to generate the resources related to Instance Product [4].

2.3. Domain-Driven Design (DDD):

Domain-driven design (DDD) centers around what matters in enterprise applications. With the help of object-oriented principles, the domain model can be developed by a developer which understandable by all the team members including business experts and technical specialists. Domain-Driven Design extracts the scope of the target application effectively but excludes the specific methods of diversity design [4].

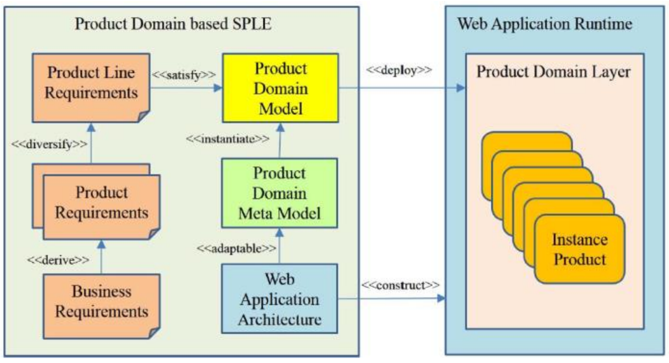


Figure 3:A concept of product domain-based SPLE for Web application [4]

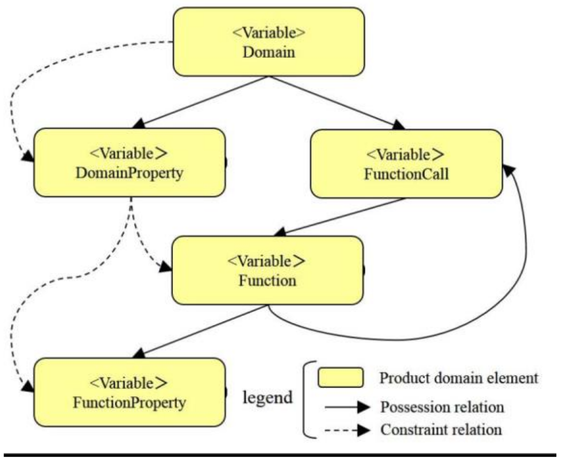


Figure 4:An overview of Product Domain Meta-Model [4]

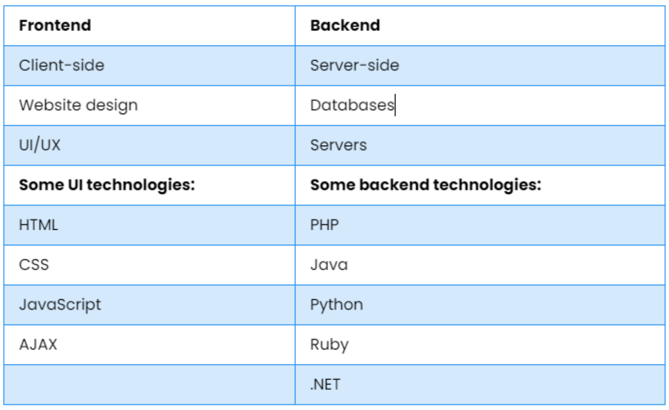
2.4. Dependency Injection (DI): For the runtime, we characterized an application design to receive dependency injection technology. The objective of dependency injection technology is the logic of a software product item. We characterized the objective as an Instance Product item. To produce a lot of assets concerning Product Items, we made a generator that inputs product domain model or Domain-Driven Design (DDD). Dependency Injection (DI) is an innovation for diminishing part dependency in application engineering with supporting application runtime. DI empowers loosely coupled components, which are accordingly exceptionally reusable. DI can move the code for starting up sub-segments from the program of a segment to a segment of the structure, for example, the Spring framework, which makes occurrences of sub-segments determined by a different arrangement document and consequently stores them in the segment. Concerning applying DI to deal with a ton of components and how to characterize relations among components is much important [4].

3. Web Technologies

Web technologies are the various tools and techniques that are utilized in the process of communication between different types of devices over the internet. To understand this term in a better manner, let us break it down into two pieces: 'web' and 'technology'. The web, in this case, refers to the World Wide Web, more commonly known as WWW and Technologies which are core to the functioning of the World Wide Web. All web pages, documents, and any other resources are identified and located with the help of their URLs. These collectively form what we refer to as the World Wide Web and these all kinds of resources are possible to view via web browsers. The process of making web development is mainly divided into two parts frontend development and backend development. Frontend refers to all those parts of a website that a user can see on their screen and interact with. Backend refers to the exact opposite of that. It involves the hidden mechanisms that make a webpage function. A typical user is generally unaware of what goes on at the backend [7].

Figure 5: Frontend vs Backend [7]

Here is a brief overview of the differences between the two categories of web development.



Frontend	Backend
Client-side	Server-side
Website design	Databases
UI/UX	Servers
Some UI technologies:	**Some backend technologies:**
HTML	PHP
CSS	Java
JavaScript	Python
AJAX	Ruby
	.NET

Figure 6:Frontend vs Backend [7]

3.1. Databases:

All the data that is exchanged on the web needs to be stored somewhere. For this purpose, most websites have their databases associated with them. Below is a list of some databases – some relational and others, non-relational – that are commonly used for web applications.

MySQL, SQL Server, Postgres, Oracle, MongoDB, Redis are some of the examples.

3.2. Data Formats:

Whenever there is a need to exchange data between two devices on the web, a proper procedure is followed. Data is packaged properly for transmission from the source to the destination. Special APIs (Application Programming Interfaces) are designed and integrated into websites for convenient data exchange. They arrange the data in such a way that the receiver can easily decode and understand it.

XML stands for Extensible Markup Language. XML breaks down data into elements identified by various types of tags. However, with XML, you can invent your tags to describe your data better. This data, upon reaching a web app or server, can be easily understood and analyzed.

JSON is a lightweight alternative to XML. After years of experiencing XML's bulkiness and heavy consumption of bandwidth, experts came up with JSON (JavaScript Object Notation). Unlike XML, JSON focuses more on quick and easy data exchange rather than detailed data definition and modeling. It also eliminates all the extra load that XML carries in the form of repetitive tags.

3.3. Open-Source vs Closed Source:

Open-source software is available for the general public to use and modify from its original design free of charge. It is distributed under a licensing agreement which allows code to be shared, viewed, and modified by other users and organizations. The source code is not shared with the public for anyone to look at or change. Closed source software can be defined as proprietary software distributed under a licensing agreement to authorized users with private modification, copying, and republishing restrictions. Closed source is the sort of arrangement that you would expect from most businesses, protective of their product and keen to maintain control over their brand and the user experience offered to their customers. Think Apple rather than Android. Many companies prefer open-source like Java, PHP, etc. for the development, deployment of their web application as they do not have to worry about the licensing cost additional to the development cost. Though it does not give them any support; the only way to get out of any issue faced in the development process is the open-source community. Whereas the closed-source technologies like the .NET framework cost an additional budget for licensing but in return, they have additional official support from Microsoft.

4. Framework

Web Application Framework or just "web structure" is a software framework that is intended to help the improvement of web applications including web services, web resources, and web APIs. Apart from this, Web technology provides one means to develop Google's Android and Apple's iOS via cross-platform frameworks. Both the platforms remain incompatible in terms of user mobility between device vendors and apps. Using such frameworks, code is written once but the app can be deployed to multiple platforms [5].

The comparison of various cross-platform development frameworks is given in Table 1 where it's quite notable that the first peak is in 2012 and 2013 after proposals to understand the proliferation of cross-platform development frameworks [5].

Year	Evaluated Frameworks	Particularities
2015	WebWorks, PhoneGap, Titanium; native Android and iOS for comparison	Extensive work including a performance evaluation and GUI considerations
2015	AngularJS, jQuery Mobile, HTML5/JS, RhoMobile, PhoneGap, Sencha Touch	Criteria definition and qualitative comparison
2015	PhoneGap, Smartface App Studio Titanium, Xamarin	Rather short paper; does not cite prior work on the topic
2015	PhoneGap, Titanium	Special focus on energy consumption
2014	Intel XDK, PhoneGap, Titanium	Evaluation by two independents teams; focus on User Experience
2014	jQuery Mobile, MoSync, PhoneGap, Titanium	Focus on animations
2013	(PhoneGap, Titanium)	The paper has a more general focus; proposing much fundamental work
2013	PhoneGap, Sencha Touch, Titanium	Assessment criteria; apps developed for performance evaluation
2013	PhoneGap (and Sencha Touch), Rhodes, Titanium	Comparison based on a sample application
2012	PhoneGap, Titanium; Webapps and native apps for comparison	Proposing criteria for evaluation; widely cited paper
2012	DragonRad, MoSync, PhoneGap, Rhodes	One of the first comprehensive studies
2012	(none)	Theoretical assessment of cross-platform possibilities in general

Table 1:Cross-Platform Framework Comparisons [5]

Native, Hybrid, and Web Apps are built using Cross-Platform Tools and these tools have come up in several technology flavors: JavaScript frameworks, App factories, Web-to-native wrappers, Runtimes, and Source code translators. Hybrid Mobile Application is one of the widely used approaches taken to build Cross-platform Applications. So, you build a Web application, which is embedded within a native application container. This application is installed, launched, and operated like any native application and can access device APIs (with some exceptions/hurdles) – but it's written in HTML, JavaScript, and CSS. Cross-Platform Web Application development reduces development time & cost with rapid design, development, and management of cross-platform web applications. With the consideration of all the aspects and according to the current trend, we are focusing on three frameworks Angular, React Native, Ionic Framework.

4.1. Choice of Framework

To guarantee that the correlation and assessment of the frameworks were pretty much as evenhanded as could be expected, just frameworks utilizing JavaScript as their primary programming language was picked. Our examination in this manner is intentionally unique to some different works that unequivocally thought about such methodologies that were paradigmatically extraordinary. With the rise of new frameworks, technologies decision is not simple. Because of the chance of basic contrasts in structures utilizing different languages, those utilizing anything besides JavaScript were prohibited.

There are mainly a few frameworks for cross-platform improvement which are picked because of their oddity, recentness, and large developer community. In conclusion, React Native, the Ionic Framework, and Angular were chosen for the assessment. [5]

4.2. Information on Frameworks

React Native is created and kept up by Facebook and was dispatched with iOS-just help in January 2015; Android support was delivered soon thereafter along with Web Application Support came into the picture. React Native, utilizing the view-delivering library React.js, is assembled and utilized effectively underway by Facebook [5].

Ionic Framework is a broad open-source hybrid application development suite kept up by Drifty Co. The application development system is just one piece of their environment; notwithstanding the structure, they likewise offer types of assistance, for example, prototyping devices, logical apparatuses, and a pop-up message administration through their foundation Ionic.io [5].

Angular is a platform and framework for building client applications using HTML and TypeScript. The basic building blocks of the Angular framework are known as an Angular component are organized into ng-modules. The ng-modules collect related code into functional sets; an Angular app is defined by a set of ng-modules. An app always has at least a root module that enables bootstrapping and typically has many more feature modules [8].

- Components define views, which are sets of screen elements that Angular can choose among and modify according to your program logic and data [8].
- Components use services, which provide specific functionality not directly related to views. Service providers can be injected into components as dependencies, making your code modular, reusable, and efficient [8].

5. Comparative Analysis of Different Frameworks

Here we will try to analyze and will try to conclude which framework can be chosen for a web application by looking at their advantages and disadvantages [9].

5.1. React Native:

5.1.1. Advantages:

- Updating DOM is usually very slow, when a big dataset is updated, react helps for fast rendering views in this situation.
- Limiting re-render for the complete web page, rather it updates the individual components based on their need.
- Creates a virtual DOM, which helps in better security [9]

5.1.2. Disadvantages:

- Full-featured server-side applications cannot be created.
- Complicated to move complex Photoshop or another sketch to a JSX file. So, it is not usable for sketch related web application [9]

5.1. Ionic Framework:

5.1.1. Advantages:

- A single codebase across various platforms. Ionic allows building web as well as mobile applications. Even we can transform Ionic application into a desktop app or PWA.
- Ease of maintenance via built-in browser instruments and debugging tools
- Reduced costs on development, hiring native developers, codebase maintenance.
- Wide range of integration capabilities and plugins.
- An extensive choice of UI elements and quick prototyping [10].

5.1.2. Disadvantages:

- Performance is lacking compared to other frameworks.
- Plugin-dependent system, it bounds to use plugins to access native functionality.
- Absence of hot reloading - Hot reloading allows you to apply changes without reloading the whole app.
- Writing using this framework and HTML, CSS, and JavaScript means writing a lot of code and adding libraries, plugins, dependencies, etc., which make an app reasonably heavier than the native ones [10].

5.1. Angular JS:

5.1.1. Advantages:

- As it is a combination of HTML and TypeScript Code, it helps to define the webpage design at a very granular level.
- Easy to implement because used keywords are self-explanatory and work well with HTML tags.
- It uses dynamic data rather than using static, more interactive sites can be made easily [9].

5.1.2. Disadvantages:

- It does not have any backward compatibility with Angular 1.x.

- The developer needs to have a good understanding of TypeScript.
- It is not easy to make SEO friendly website as the indexing test is not directly coded in HTML tags [9].

So, looking at the above comparison we can conclude that there is no framework good or bad framework, they are just a tool to solve a specific set of problems. The choice of the correct framework is dependent on the requirement of the web application considering security, budget, targeted audience, and time to develop.

6. Conclusions

In this paper, a broad examination on how the Software Product Line approaches for web-based applications and concentrated on some of the methodologies in detail and its connected models which carry out these product offering procedures for a simpler turn of events and standard approach to develop web applications which are the main part of the web today's world. There are emerging technologies on the horizon that use or change SPL technologies to create a reliable, reusable product. As such we cannot conclude which SPL approach is better as it entirely depends on the requirement of our web- applications developed. many open-source communities and organizations are doing a collaborative approach for the development of reusable artifacts which will eventually help any software development project to use as it is or customize the version of the artifacts.

References

[1] Luca Balzerani, Davide Di Ruscio, and Alfonso Pierantonio; A product line architecture for web applications; Proceedings of the 2005 ACM Symposium on Applied Computing (SAC), Santa Fe, New Mexico, USA, March 13-17, 2005.

[2] Makoto Yoshida and Noriyuki Iwane; An approach to the software product line system for web applications; 2006 International Conference on Computing & Informatics, Kuala Lumpur, Malaysia, 6-8 June 2006

[3] Takashi Nerome and Masayuki Numao; A Product Domain Model-Based Software Product Line Engineering for Web Application; 2014 Second International Symposium on Computing and Networking, Shizuoka, Japan, 10-12 Dec. 2014

[4] Revati Gandhi, Sukhada Pande, and Mayuree Nandi; Software Product line Technologies for Web-Based Application; International Journal of Engineering Research & Technology (IJERT), ICSITS – 2020 (Volume 8 – Issue 05), 19th March 2020.

[5] Tim A. Majchrzak, Andreas Biørn-Hansen, and Tor-Morten Grønli; Comprehensive Analysis of Innovative Cross-Platform App Development Frameworks, Hawaii International Conference on System Sciences, January 2017

[6] https://www.thepsi.com/what-it-takes-to-build-a-cross-platform-web-application/

[7] https://www.goodcore.co.uk/blog/web-technologies/

[8] https://angular.io/guide/architecture

[9] https://www.pixelcrayons.com/blog/best-web-development-frameworks-comparison/

[10] https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-ionic-mobile-development/