Sofware Product Line Frameworks For Web-Based Applications

Sirisha Cherala, Second B. Author Jr., and Third C. Author,
Computer Science, Illinois Institute of Technology

Abstract— Web applications have evolved from publishing static information to serving dynamic and complex web content involving business functions. Besides, increasing number of browsers, platforms and devices are driving web application software to accommodate a wide range of client-side specifications without increasing development cost and time.To implement systems capable of handling such diversity as well as complex functionality Software Product Line Engineering (SPLE) is a promising software development strategy. To this end SPL frameworks provide a way to approach various problems concerning the use of SPLE in Web application domain. Some of these concerns include, dealing with cross-cutting behaviours, manage the impact of Non-Functional requirements, mapping the variability at source code with the variability modeled at a higher level of abstraction, allowing average business user to generate ready to run web applications, reducing the complexity of customization process,etc., In this paper, we did a study and analysis of various SPL frameworks that provide a way to deal with the above concerns and discussed the outcomes of the review.

Keywords— Software Product Line, Crosss- cutting behavior, Non-Functional requirements, Customization, Variability

1 Introduction

The unbridled evolution of Web-based presentation techniques, different access platforms, enormous amount of system integration requirements, different client-side specifications has impacted web application domain driving applications to meet all the requirements without increasing development cost and time. Besides, Development and Maintenance of web applications usually takes place under tight schedules. So, for construction of such web applications spanning multiple domains and customers, Software Product Line Engineering is needed to manage the commonalities and variabilities of the product family. To this end, SPL Frameworks help by providing a set of approaches, a group of programs (boiler plate code), a set of rules indicating how SPL needs to be built and how componets of SPL interact with each other so as to have efficient reuse.

The paper is organized as follows: Section 2 covers overview of Software Product Lines, Section 3 provides the reviewed Frameworks for SPL in web application domain, Section 4 gives a comparative analysis on studied frameworks and Section 5 draws the Conclusion.

2 Background

2.1 Overview of Software Product Lines

SPL is a portfolio of similar software-based systems and products produced from a shared set of **software assets** using a common means of production. Software product line Engineering (SPLE) is a strategy to create, evolve and sustain SPL's. SPLE consists of a systematic way to capture the common parts between products of a family of systems, while accommodating the differences. It results in establishing a reusable platform and a set of variable parts. The main benefit is a maintainable and comprehensible software asset base that can be reused. Whenever a new product needs to be developed, the software asset base formed is refined and improvised according to the product's particular software requirements. This might cause errors or might even cost more than developing from scratch instead of reusing.So, for efficient reuse, software engineering process is split into two phases: 1.Domain engineering and 2.Application engineering.

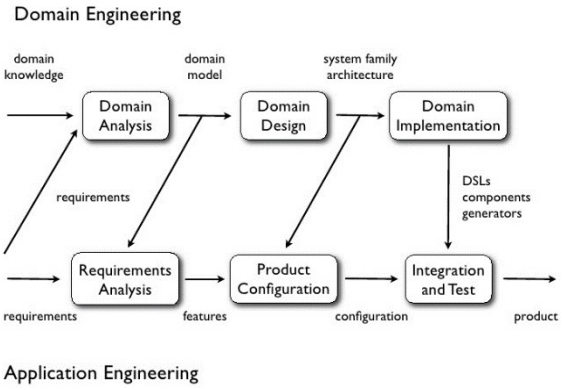

Figure 1: Software Product Line Life Cycle

Domain Engineering establishes the product line whereas the application engineering develops a product from the product line. As such, Figure 1 shows how the two phases of SPLE interact with each other and make reusability possible.

2.2 SPL at the Web Application Domain

Web Systems are an area where the rapid response times of SPLE is beneficial. The dynamic nature of different features in web applications needs to be handled properly. A given functionality can behave differently based on the

various access as well as system requirements and the system is expected to work efficiently. By incorporating SPLE in web-based development, advantages of commonalities and variabilities can be used to build custom web products. As such, Web engineering based on a standardized platform with reusable components is a logical next step in the evolution of Web application development.

2.3 SPL Frameworks in Web Domain

The term framework refers to the essential part of software or a product which helps to keep the product useful. The framework provides a certain structure to the product and defines how it should perform. A well designed SPL framework could reduce Software Building Effort by providing an approach or a set of rules for domain analysis, requirements analysis, or a set of programs for domain implementation, and software generation.

As such in web application domain, frameworks help to deal with cross cutting behaviors, support variability management at diverse abstract levels that could accommodate particularities in web-based languages, reduce complexity of service customization process, allow an average business user to generate ready to run web applications and many others.

3 SPL Frameworks For Web Applications

3.1 Aspect Oriented Framework

Aspect oriented Framework has been developed to deal with Cross cutting behaviours and also to manage the impact of Non-Functional Requirements which are common in web application domain. In software development crosscutting concerns are those broadly-scoped features or properties that often crosscut several other modules in software system. Non-Functional requirements are those which specify how a system needs to behave.

The framework focuses on two key reusable artifacts of domain engineering phase which are basis for SPL development and products derivation: feature model and reference architecture. Feature Model is used to configure member products and represents all possible member products of SPL interms of features and their relationships. Reference architecture is derived based on feature model. It provides common structures, components and their relation ships and also describes planned variabilities. Based on the requirements on the product, the reference architecture provides templates for developing concrete architectures for the members in the product line.

Problems addressed by Aspect Oriented Framework:

- Modelling complex feature relationships.
- Modelling relationship between quality attributes and features.
- Establishing systematic feature mappings to components.
- Establishing an understandable way to describe desired features for products and the possible variations to reference architecture.
- Representing impact of NFRs on feature selections and architecture.

To better model NFRs and cross cutting features, this framework consists of two parts:

1) Aspect Oriented (AO) Feature Model
2) Aspect Oriented (AO) architecture development

3.1.1 AO Feature Model

Goal: To convert conventional feature model into four elements, i.e. NFRs, concrete features, aspectual features and aspectual concerns. It also aims to define impact of aspectual concerns on both concrete and aspectual features.

Steps in AO Feature Modelling:

1. Model Concrete features (user visible functionalities) and relation ships between them. This can be done using use case modeling and scenario based approaches.
2. Examine and modularize aspectual concerns i.e., contribution of each concrete feature to satisfy NFR's.
3. Identify aspectual features i.e., the features that cross cut or impact concrete features and map them to the impacted concrete features.

3.1.2 AO Reference Architecture

Goal: To specify how features and aspectual concerns from feature model are mapped onto reference architecture. Identify the crosscutting concerns at architecture level and present the impact on the reference architecture.

Steps in AO Reference architecture:

1. Develop components and subsystems to implement concrete features.
2. Develop set of architecuture scenarios describing internal behaviours of systems.
3. Identify variabilities of architecture components, model interactions between components and possible ways components cross cut each other.
4. Address aspectual concerns by identifying scenario interactions which refines possible bigger scope crosscutting concerns.

3.1.3 Case Study – Crisis Management Web System Product Line

This Case study illustrates how this framework can help in identifying aspects from requirement level and map them to architecture level.

Table 1 below shows features with their responsibilities. By relating these features to use case mappings, some features exist in multiple business flows. These features should be considered as candidates of aspectual features. In crisis management system, capturing a crisis report is one of the business flows. In this flow, we need to rerank crisis priority (one of the functional behavior) whenever a crisis has been resolved, whenever there is an update in crisis information, whenever a new crisis comes in. This shows that reranking crisis priority would be a potential candidate of aspectual feature. Likewise first two steps of AO feature model could be implemented.

Features	Responsibility
Crisis receive	when system receives a new crisis report, this information will be sent to center management
Position detection	to confirm a crisis report, the location has to be identified by various approaches
Communication	which provides support for communication and information transmission among all resources
Task assignment	system assigns rescue tasks to mobile units according to current resources and crisis priorities
rerankingCrisisPriority	once a crisis been received or a rescue task been completed, system will automatically re-rank the rest crisis based on their critical levels

Table1: Features and responsibilies

Aspectual Concerns	Performance	Security	Mobility
Constant cache management	X	X	
Instant message exchange	X		X
Fast logging verification	X	X	

Table 2: Aspectual concerns & related Quality Attributes

Next step is to identify aspectual concerns.In this case study, constant cache-management is one of the aspectual concern as it has an impact on Security and Performance, quality attributes of the System. Table 2 lists the aspectual concerns.

Once concrete, aspectual features and concerns are identified, according to the framework AO architecuture needs to be built. For this set of architecture scenarios need to be developed to describe system architecture, which includes branching lines and alternative flows to address architectural variabilities.

In the figure 2, identified aspectual feature "rerankingCrisisPriority" is mapped as a component called "reranking"on architecture. To illustrate, "reranking" crosscuts components "Report" and "Task management" through interface "data transferring" to transmit data for systems report generation and live task management.It also crosscuts "Communication" and "Execute mission" through "information collecting" to provide secure/instant communication and information update among available resources.

In this way, AO framework shows how it can be used to describe feature modelling and develop architecuture models for web applications.

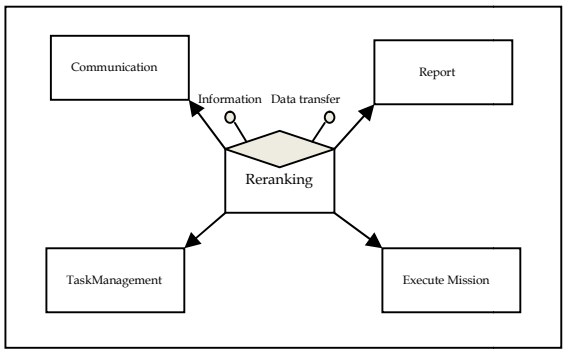

Figure 2: Re-reranking cross cuts other components

3.1.4 Advantages of Aspect Oriented Framework:

- Enhances modularity of System
- The transformation between requirement and architecture is more efficient.
- The software architecture developments will be-more accurate because NFRs and qualities are treated as part of architectural elements.

3.2 Framework Based on Process Models

This framework is based on the fact that the use of comprehensive process models would enable an average business user to generate ready to run web applications. The goal of this framework is two fold: 1) enable domain developers to define domain specific product line artifacts 2) enable product managers to configure concrete software products using process models.

3.2.1 Framework Development Process

Step -1: Determine and scope a specific domain (Example: hotel booking, book shopping, shop rental etc.,).

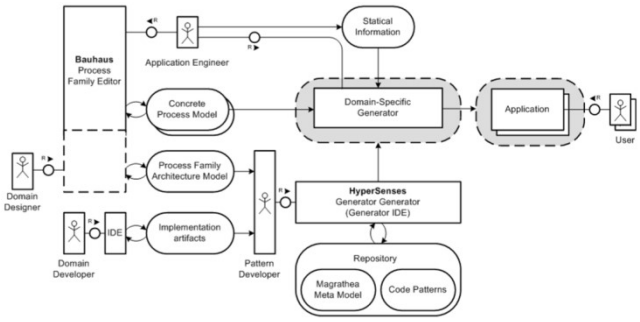

Figure 3: Development process

Step -2: Define process family architecture model comprising of all possible process variants. This is done by domain designers using "Bauhaus Process Family Editor". The modeling notation for process family architecture models is BPMN extended by variation points The modeling notation for architecture models is BPMN extended by variation points.

Step -3: Once family architecture has been designed, the phase of domain implementation starts. Here, domain developers can either implement pluggable code artifacts which are to be mapped to processs artifacts or implement product variants that are decomposed to code artifacts by pattern developers.

Step -4: This steps involves the configuration of Hypersenses generator which is done by pattern developers using code artifacts and process family architecture model.

Step –5: After the above step, the configured generator builds every possible process variant in the process family.This steps finishes Domain Engineering phase.

Step -6: Then comes Application Engineering phase. The first process is to analyse concrete business requirements of required application.

Step -7: Application design needs to done by using Bauhaus to derive a concrete process model either by resolving variation points in the domain engineered process family architecture models or by freely combining the building blocks of the process.
Step -8: The outcome of Step-7 would be a concrete variant free business process model. This is fed to generator configured in Step-4 which outputs a ready to run web application based on the specific application requirements.

Apart from providing the above development methodology, this framework illustrates variability mechanisms to derive process model variants from process models (Step-2 of development process) and also emphasizes on how elements comprising process models, how these elements are mapped to code in the generator(Step-5).

At the last for evaluation purpose, this framework has been used for designing Hotel Booking product line.

3.2.2 Variability Mechanisms

This framework uses Encapsulated Subprocesses, Abstract Subprocesses, Null Subprocesses and Optional Stereotypes as mechanisms to model variability.

Encapsulated Subprocesses are referenced using a plus sign at the bottom of an activity node. This allows for information hiding since they are modeled in detail in a separate pool. Thus the subprocesses are interchangeable.

Abstract Subprocesses are used to express an extension point. This is done by using an implements relation to denote possible implementations of a certain activity.

Null Subprocesses are used to model the case where a subprocess would not exectute any activity apart from handing over input data to subsequent activities.

Optional Stereotype is applied to a subprocess to express that it may but need not be present in a concrete process variant.

3.2.3 Process Models and Model-based Code Generation

Modelling a System implies depicting System aspects that are relevant to certain concern and leaving out others. The model proposed by this framework comprises of:

- Pools(Processes)
- Tasks
- Subprocesses
- Start and end events
- Gateways
- Sequence Flow
- Data Objects
- Message Flow

Using the model, based on above elements, the code generator maps each of the element to model,view or controller as MVC is used as a programming paradigm.

3.2.4 Case Study – Hotel Booking Web Application Domain

As per the framework, the domain is scoped as Hotel Booking platform. Domain analysis is performed based on the requirements by the users. The following are the list of requirements which are modeled as processes.

- Basic Requirements
  - Search for Hotels
  - Display Search Results
  - Display Hotel Details
  - Display different pricing categories
  - Enable user to enter personal details
  - Enable user to enter payment information
  - Enable user to commit a booking request
- Advanced Requirements
  - Allow user to register and log in
  - Provide single-sign-on, disable registration
  - Enable user to store and change a personal profile
  - Prefill personal and payment detail fields for logged in user
  - Display past bookings
  - Enable user to cancel a booking
  - Enable user to book a visited hotel again
  - Deny booking for non-authorized user
  - Skip personal and payment information forms for logged in user
  - Display text messages in different languages

Based on these requirements, variant rich process models are developed during domain analysis phase. During Domain Implementation, development of:

1) Controller code blocks that collect data from models and prepare for views.
2) Views to display data to user and collect information in web-forms.
3) Controller code blocks that validate, manipulate data and interact with models to store the data.
4) Models that handle persistence of data.

This Implementation along with process model is configured in the generator and is used for variants of Hotel Booking platforms.

3.2.5 Advantages

- Large number of Similar but different web applications can be development in a short time.
- Addresses Dynamic Aspects of software modelling (processes) that underlie software products.
- Incorporates mechanisms to account for variabilities in existing process models
- Lack of possibility for free recombination of processes constituents is solved by employing process building blocks.

3.3 Web Service Customozation Framework

This framework provides a feature oriented approach for Web Services Customization. Service customization refers to operations of adapting a service to a particular application scenario. The main goal of this framework is to reduce the complexity in the customization process and enable automatic validation of the resulting customized services. This is done by exploiting feature modeling techniques of SPL where all possible customization options are captured into a feature model. Also weaving models are used to describe links between feature model and service model. These enable service consumers to customize through selection of features in the feature model.

This framework is described as follows.In 3.3.1, feature modeling technique and its benefits will be discussed followed by overview of framework in 3.3.2. The advantages of this framework are given in 3.3.3.

3.3.1 Feature Modeling Technique and Benefits

A feature is a visible characteristic of a product family.A feature model represents the information of all possible products of a product line as features and relationships among them.

Feature model shows a hirerachy composed by: Relationships between parent feature and its child feature which can be mandatory, optional and alternative. Cross tree constraints which are dependencies among features.

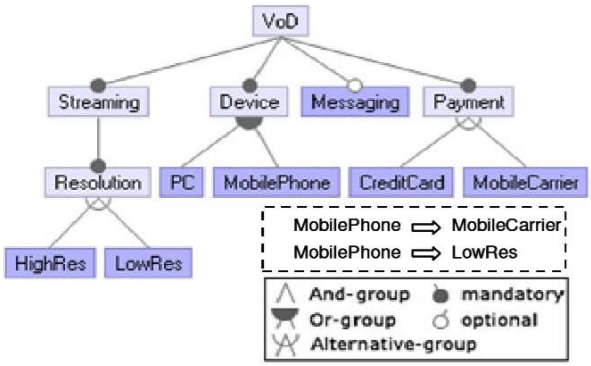

Figure 4: Sample Feature Model

Figure 3 depicts a feature model representing a family of video on demand services. As seen from the feature model, all the customized web services should have streaming, device and payment as mandatory features. Messaging is an optional feature.

As seen from the above figure 4, it is easy as well as intuitive to represent features through feature models.

Benefits:

- Business analysts and engineers communicate interms of which features the service has or which it delivers; feature models are a more efficient way for service customization.
- Enables variability modeling at feature level
- Feature model reduces the number of variation points available for customization.
- Can analyzed and validated automatically.

3.3.2 Overview

Step -1: Service provider analyses the commonalities and variabilities of service family and captures all of them into a feature model.

Step -2: Feature model is published to enable service consumers so as to enable them to select the features they need in the service

Step -3: Service consumers discover the feature models they are interested in.

Step -4: Feature configuration is done in this step by service consumers.This is done by enabling or disabling optional features in the feature model and also selecting a particular feature from a different feature group.

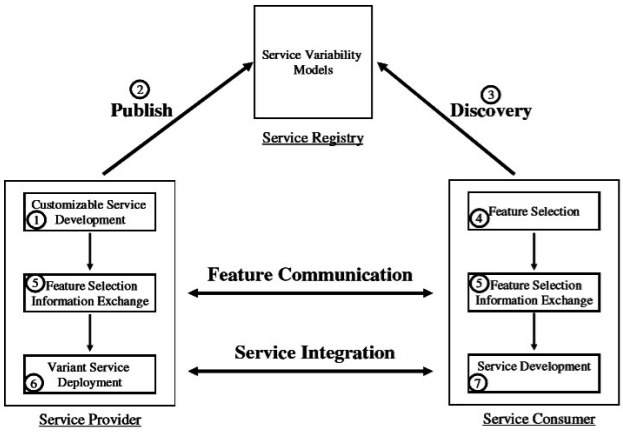

Figure 5: Overview of Framework

Step- 5: Information on selected features is exchanged between service consumer and provider.

Step -6: Based on information provided by consumer in the previous step, the service provider generates the particular service interface and deploys it

Step -7: As the customized service interface is ready, the consumer implements it and developer the service.

3.3.3 Advanatages

- Complexity of web service customization is reduced by the use of feature models to represent variability
- Automated validation is made possible as the customized service can be compared with the selected feature model to check if the desired features have been incorporated.

3.4 FeatureJS - Feature Models and Pre-processors based Framework

To support the increasing number of devices, platforms and browsers in web system domain most of the Client-side code is written in Java Script. As using JavaScript could not adequately support the variabilities as it has its own particularities, this framework uses a combination of

Feature-oriented programming and preprocessors as a strategy to solve this problem. This allows for mapping of variability at source code with the variability at a higher level of abstraction.

3.4.1 Overview of Framework

This framework is intended to support for SPL development in JavaScript and HTML, the prominent client side languages in Web Domain.

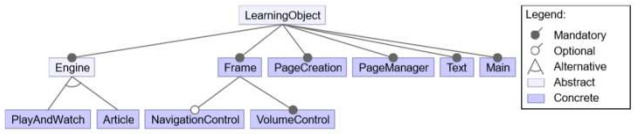

Figure 6: Example of a feature Model

To illustrate on how frameworks helps to capture variabilities at implementation level is described by considering a feature model shown in Figure 6.

From the figure, we can see that Learning Object is the root feature representing the domain taken for analysis. It has some concrete features like Frame, Page Creation, etc., as well as abstract features like Engine. Engine has two sub features PlayAndWatch as well as article. So, engine can be set to any of the both.

Let us consider a scenario wherein the implementation of the feature Frame depends on the subfeature of Engine.In this case, the way in which FeatureJS deals is shown in Figure 6.

The preprocessor directives ifdef, elif in the Frame.js source code of Figure 7 specify which part of the code needs to compiled based on the feature selected.So, the implementation for ref.clickForwardPage in the example considered in Figure 7 would contain a different code based on the sub features of Engine i.e., PlayAndWatch and Article. As a result the product variant would contain the code only for the configuration selected and would not contain multiple if else statements having code for the different configurations.

This strategy would also reduce maintanence effort as business rules corresponding to a single configuration would be present in a single file. Besides it also allows for program level customization at an early period in development cycle.

3.4.2 Architecture

The Architecture of FeatureJs and its components are depicted in Figure8.

The top level components of FeatureJS framework are used to link the FeatureJS with FeatureIDE which is an Eclipse based framework for feature oriented programming. The current framework extends it by incorporating strategies to provide automated support for JavaScript.
The following are the components exclusive in FeatureJS

```
(function (window){
                    ⋮
// #ifdef PlayAndWatch
ref.clickForwardPage = function(event){
   if(_mode == _pageTypeEnum.PLAY || _mode ==
 _pageTypeEnum.PLAY_ANDROID || _mode ==
 _pageTypeEnum.PLAY_ANDROID_CHROME) { _page.
     animate();
   }
   else{
      _pageManager.forwardPage();
   }
}
// #elif Article
ref.clickForwardPage = function(event){
   _pageManager.forwardPage();
}
// #endif
                    ⋮
}(window))
```

Figure 7: Frame.js developed using FeatureJS

architecture:

- File Manager: Business Rule Class.
  - Manages Files that have been copied to keep only the selected features and remove the rest.
  - Defines precedence logic for each preprocessor directive, and the replacement policy.
- File Feature : Retrieves all feature interactions
- File Interpreter: Abstraction of code fragment framed by pre-processor directive.
- Feature Treatment: Encapsulates information on feature and generates regular expression to recognize a directive.
- Preprocessor Directives: All directives used by Feature Treatment are placed in this class. The directives used by framework are #ifdef, #ifndef, #else, #elif, and #endif.
- Feature Metrics: Identifies files which contain preprocessor directive so as to track files when a modification is needed.

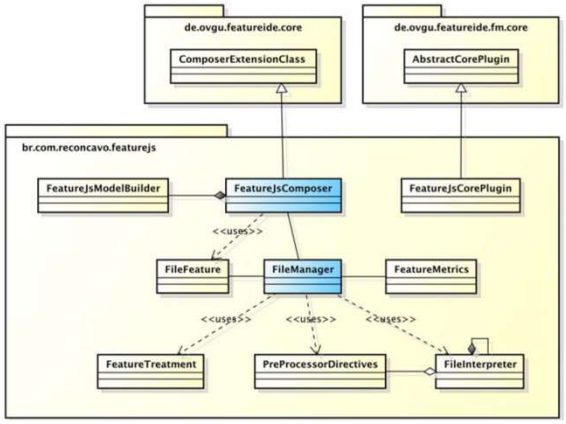

Figure 8: FeatureJS Class diagram

3.4.3 Advantages

- ✓ Capable of managing variability at implementation level.

- ✓ Supports coding in JavaScript and HTML, popular client side coding languages in web domain.
- ✓ Has a positive impact on development cost and time as well as maintainability of SPL in web system domain.

4 COMPARITIVE ANALYSIS

A Comparitive Analysis of the studied Frameworks is

	Aspect Oriented	Process Model	Service Customization	Feature Js
Requirements	Yes	Yes	Yes	Yes
Variability	Yes	Yes	Yes	Yes
Design and Architecting	Yes	Yes	–	–
Cross Cutting Concerns	Yes	–	–	Yes
Implementation	–	Yes	Yes	Yes
Validation	–	–	Yes	–

Shown in Figure 8.

Figure 8: Table comparing studied Frameworks

5 CONCLUSION

It is seen that much similarity across Web applications creates opportunities for cutting development cost and easing evolution via reuse.As web applications have evolved steadily and ubiquitously to serve dynamic and complex Web content, the need for an extensible and flexible framework is increasing. The existing frameworks studied in the try to solve problems pertaining to a specific area like requirement to architecture mapping, feature to implementation mapping, etc., there is still a need for framework which can accommodate all the phases in web engineering starting from domain engineering to building an actual customized web application

REFERENCES

[1] J.S. Bridle, "Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition," *Neurocomputing – Algorithms, Architectures and Applications,* F. Fogelman-Soulie and J. Herault, eds., NATO ASI Series F68, Berlin: Springer-Verlag, pp. 227-236, 1989. (Book style with paper title and editor)

[2] W.-K. Chen, *Linear Networks and Systems.* Belmont, Calif.: Wadsworth, pp. 123-135, 1993. (Book style)

[3] H. Poor, "A Hypertext History of Multiuser Dimensions," *MUD History,* http://www.ccs.neu.edu/home/pb/mud-history.html. 1986. (URL link *include year)

[4] K. Elissa, "An Overview of Decision Theory," unpublished. (Unpublished manuscript)

[5] R. Nicole, "The Last Word on Decision Theory," *J. Computer Vision,* submitted for publication. (Pending publication)