아키텍처 평가 방법을 통합한 아키텍처 설계 방법

고현희[O] 궁상환 박재년
숙명여자대학교 천안대학교
{hhkoh, jnpark}@sookmyung.ac.kr

Architecture Design Method Intergrating Architecture Evaluation Method

HyonHee Koh [O] SangHwan Kung, JaeNyon Park
Sookmyung Womens' University, CheonAn University

요 약

하나의 시스템의 아키텍처를 설계하기위해서는 여러 아키텍처 스타일들이 복합적으로 결합하여 시스템의 전체적인 아키텍처를 구성하게 된다. 그러나 아키텍처 설계자 또는 개발자들이 아키텍처 설계 과정 중에 현재 설계된 아키텍처가 시스템의 요구사항을 적절히 반영하고 있는지 평가해 볼 수 있는 방법이 없다.
본 논문에서는 아키텍처 평가자와 다양한 이해관계자가 모여 아키텍처를 전반적으로 평가해 보기 전에 아키텍처 설계자가 설계과정 중에 자신이 담당한 부분의 현재의 아키텍처가 시스템의 요구사항을 적절히 반영하고 있는지 여부를 판단하여 설계 과정중에 개선점을 도출하고, 개선점을 재 설계에 반영하는 아키텍처 설계 방법론을 제안하고자 한다

1. 서 론

하나의 시스템의 아키텍처를 설계하기위해서는 여러 아키텍처 스타일들이 복합적으로 결합하여 시스템의 전체적인 아키텍처를 구성하게 된다. 이 때 다양한 아키텍처중 어떤 아키텍처를 선택 할 것인가는 완성될 시스템이 어떤 기능적 또는 비 기능적 품질요구사항을 만족시켜야 하는지에 따라 달라지게 된다. 즉 아키텍처 선정 시는 그 시스템의 요구사항에 따라 아키텍처를 검증하고 품질 요구사항을 포함하는 요구사항을 만족할 수 있는 아키텍처를 선정해야 한다.

이렇게 시스템이 요구하는 품질 속성에 따라 아키텍처스타일을 선정하여 아키텍처를 설계하였다고 하더라도 하나의 시스템이 하나의 스타일로만 구성이 되는 것이 아니라 여러 스타일이 복합적으로 구성이 되어야 하고, 아키텍처 재설계가 필요한 부분도 생기게 된다. 이렇게 설계된 아키텍처는 그 아키텍처의 적합성에 대해 여러 방법으로 평가가 이루어 지게 된다.

현재까지는 아키텍처 설계가 끝난 뒤 다양한 이해 당사자들이 모여 아키텍처 평가자들과 함께 아키텍처를 평가하는 방법이 보편적으로 사용되고 있다. 이는 설계 과정 중에 설계를 도와줄 수 있는 검증과정이 아닐 뿐 아니라, 실행에 따르는 비용과, 설계 평가 이후에 문제점이 발견된 경우 다시 재설계를 해야 하는 부담이 따르게 된다.

본 논문에서는 아키텍처 평가자와 다양한 이해관계자가 모여 아키텍처를 설계과정 중에 자신이 담당한 부분의 현재의 아키텍처가 시스템의 요구사항을 적절히 반영하고 있는지 여부를 판단하여 설계 과정중에 개선점을 도출하고, 개선점을 재 설계에 반영하는 아키텍처 설계 방법론을 제안하고자 한다.

2. 아키텍처 설계 및 평가 방법

2.1 아키텍처 설계

아키텍처 설계 방법은 미국 Carnegie Mellon 대학의 SEI(Software Engineering Institute)에서 연구 개발한 아키텍처 기반 설계 방법(ABD)과 품질 속성에 기반 한 아키텍처 설계 방법(ADD)을 대표적으로 볼 수 있다. ABD는 상위 수준의 트웨어 아키텍처를 설계하기 위한 방법으로 특정한 제품을 만들 때 다양성에 대한 필요를 허용하는 추상적인 수준에서, 기능적 요구사항, 품질 요구사항, 그리고 비즈니스 요구사항을 만족하도록 한다.

ABD 방법론의 수행단계는 다음과 같다.

1. 기능을 분할 한다.
2. 아키텍처 스타일을 선택한다.
3. 기능을 스타일에 할당한다.
4. 템플릿을 정제한다.
5. 기능을 검증한다.
6. 동시성(Concurrency) 뷰를 생성한다.
7. 배치(Deployment) 뷰를 생성한다.
8. 품질 시나리오를 검증한다.
9. 제약사항을 검증한다.

품질 속성 기반의 설계 방법론(Attribute Driven Design) ADD는, " 사용자의 요구사항을 최상으로 만족시키도록, 한 시스템의 아키텍처를 어떻게 설계할 것인가?" 에 중점을 둔 설계 방법론이다. ADD는 기능요구사항과 품질 요구사항 둘 다를 포함한다는 것을 가정하고, 또한 각기 다른 아키텍처 뷰를 사용한다. ADD 방법론의 수행 단계는 다음과 같다

(1) 단계 1 : 분해할 모듈을 선택한다.
설계 대상이 될 모듈을 선택하고 이 모듈의 제약 조건, 품질 요구사항, 기능 요구사항도 정의 한다.
(2) 단계 2 : 선택한 모듈을 분해하고 정제한다..
단계 2a. 아키텍처 동인을 결정한다.
단계 2b. 아키텍처 스타일을 선택한다.
단계 2c. 모듈을 실체화하고 기능을 할당한다.
단계 2d. 하위 모듈의 인터페이스를 정의한다
단계 2e. 유스케이스와 품질 요소를 정제하고 검증해서 하위 모듈의 제약 사항으로 바꾼다.
(3) 단계 3 : 모듈을 분해할 필요가 없을 때까지 반복한다.

2.2 아키텍처 평가

아키텍처 평가의 결과는 아키텍처가 시스템에 적절하게 설계되었는지, 시스템을 위해서 가장 적절한 아키텍처가 무엇인지, 즉 아키텍처의 적합성을 판단하는 것이다. 이러한 아키텍처 평가 방법에는 평가 방법이나, 평가 대상, 평가의 구체성에 따라 여러 방법들이 나와 있다.

아키텍처 트레이드 오프 분석 방법인 ATAM(Architecture Tradeoff Analysis Method)은 품질 속성 요구사항과 비즈니스 목표달성을 위한 아키텍처 결정 사항들을 평가한다. 즉 ATAM은 시나리오를 중심으로 품질 속성요구사항을 찾아내고 아키텍처가 특정 품질 속성들을 만족하는지를 분석하는데, 다양한 이해 관계자들과 함께 위험요소(risk), 민감점(sensitivity point), 절충점(tradeoff point) 등을 분석한다.

이렇게 분석된 품질목표들 간의 발생하는 충돌을 프로젝트 초기에 발견함으로써 경제적인 방법으로 문제를 해결 할 수 있다.

ATAM은 4 Phase 와 9개의 step으로 이루어져 있으며, 평가팀과, 이해관계자들, 프로젝트 의사 결정권자들이 모여 아키텍처 전반에 대해 평가를 하게 된다. 따라서 아키텍처 평가조직과, 개발 조직간의 계약 또는 합의를 통해 이루어지는 조직 규모의 평가 방법이라고 볼 수 있다.

아키텍처에는 달성해야 하는 품질 속성에 대한 기술적인 측면과 시스템을 구축할 때 드는 비용과 시스템이 제공하는 품질에서 얻을 수 있는 이득에 대한 경제적인 측면이 있다.

비용과 이득을 고려한 아키텍처 평가 방법인 CBAM은 아키텍처 접근법을 실현하는 데 필요한 비용과 아키텍처 접근법을 적용했을 때 달성할 수 있는 품질 속성이 가져다 주는 이익을 측정한다. 즉, 비용과 이익으로부터 ' 투자대비효과(ROI)' 를 계산한다. CBAM은 수익이 최대가 될 수 있도록 의사 결정을 지원한다.

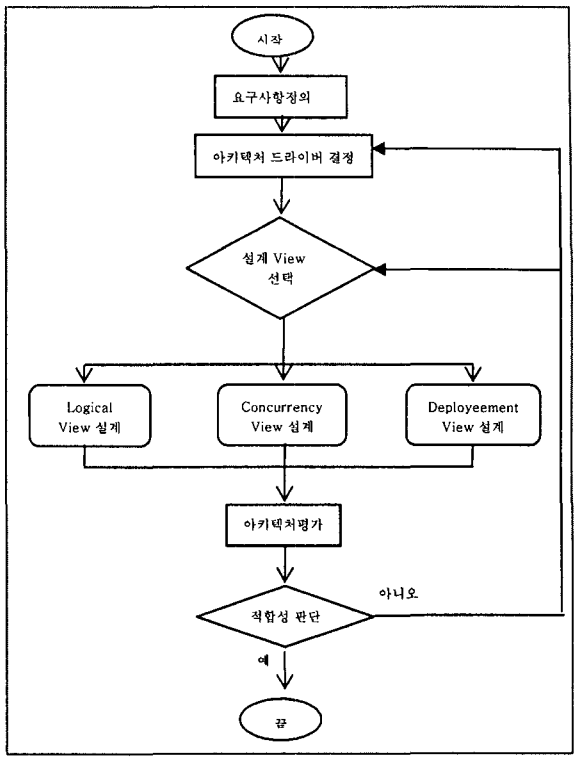
3. 아키텍처 평가 방법을 통합한 아키텍처 설계 방법

위에서 살펴본 ABD나 ADD는 기능요구사항과 품질 요구사항을 기반으로 사용자의 요구사항을 최상으로 실현시키는 아키텍처를 설계하고자 하는 방법이다. 아키텍처 설계 시 선택되는 아키텍처 스타일은 품질 속성 달성방안을 실현 할 수 있는 스타일이나 아키텍처 패턴이 선택된다. 이러한 아키텍처 스타일의 선택을 위해서는 품질 속성을 달성 시키는 아키텍처 스타일들에 대한 분류와 검증이 따라야 한다. 또한 하나의 시스템 또는 분해된 모듈은 하나의 아키텍처 스타일로 구성되는 경우뿐 아니라, 기존 아키텍처 스타일의 변형, 또는 여러 스타일 또는 패턴들의 조합으로 이루어 지는 경우도 있다. 이 경우 달성하고자 하는 품질 속성에 대해 설계 된 아키텍처에 대한 명확한 검증이 필요하다. 따라서 본 절에서는 위에서 살펴 본 아키텍처 평가 방법을 활용하여 아키텍처 설계 과정 중에 설계된 대안 아키텍처의 품질 평가 모델을 만들고, 시험과정을 거쳐 달성하고자 하는 품질 속성 요구사항에 대한 객관적인 검증을 통해 아키텍처를 설계를 발전시켜 나가는 아키텍처 설계 방법론을 제안하고자 한다.

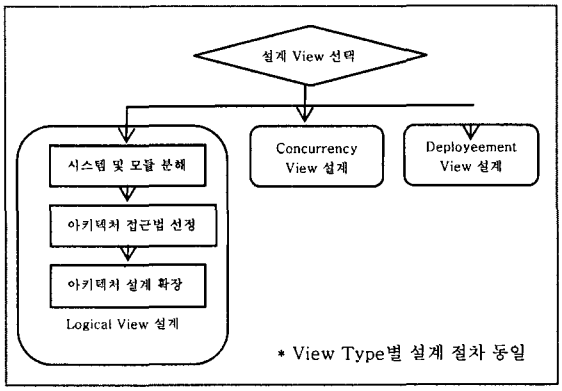
아키텍처 평가 방법을 이용한 기본적인 아키텍처 설계 절차는 [그림 1]과 같다. 먼저 단계 1에서 요구사항 분석 단계에서 시스템의 요구사항 분석과 아키텍처 동인을 식별하여 아키텍처 설계를 위한 준비를 한다.

단계 2에서는 아키텍처 동인에 따라 아키텍처 설계가 이루어지는데 이 때 구체적인 아키텍처 설계를 위해서는 설계 뷰 타입을 선택 하고 각 뷰 타입별로 아키텍처 접근법을 결정 하여야 한다.

아키텍처 설계 절차는 [그림 2]와 같이 세부 실행 내용으로 정제해 볼 수 있다.



[그림 1] 아키텍처 설계 절차



[그림 2] 뷰타입별 아키텍처 설계 세부 절차

3.1 단계 별 세부 실행 내용

아키텍처 설계 절차는 다음과 같이 세부 단계로 나타낼 수 있다.

1. 요구사항 정의
2. 아키텍처 동인 결정
3. 설계 뷰 선택
4. 시스템 및 모듈 분해 (반복)
5. 아키텍처 접근법 식별, 평가 및 선정
6. 아키텍처 설계
7. 아키텍처 설계 평가

각 단계별 세부 실행 내용은 다음과 같다

3.2.1 요구사항정의

요구사항은 시스템의 기능 요구사항과 비기능 요구사항으로 나누어 볼 수 있다. 요구사항을 시나리오 형태로 표현하고 정제하는 과정을 거친다.

1. 요구사항 정의
   1-1 시나리오를 수집한다. (기능요구사항, 비기능 요구사항)
   1-2 시나리오를 정제한다.

3.2.2 아키텍처 동인 결정

아키텍처 동인을 선정할 때는 가장 높은 우선순위를 가진 비즈니스 목표를 찾고, 찾은 비즈니스 목표를 품질 시나리오나 유스케이스로 바꾼다. 이 과정은 품질 속성 유틸리티 트리를 작성하는 과정을 통해 실행한다.

2. 아키텍처 동인 결정
   2-1 시나리오 우선순위를 정한다.
   2-2 유스케이스 다이어그램과 유스케이스 시나리오를 작성한다.
   2-3 품질 속성 유틸리티 트리를 작성한다.

3.2.3 설계 뷰 선택

설계 뷰는 Logical 뷰, Concurrency 뷰, Deployment 뷰로 구분된다.

3. 설계 뷰 선택
   3-1 모듈 별 설계 뷰 선택

3.2.4 시스템 및 모듈 분해

분해된 모듈은 아키텍처 설계의 단위가 되며, 각 팀 또는 조직으로 할당되는 단위가 된다.

4. 시스템 및 모듈 분해
   4-1 적절한 수준의 서브시스템으로 분할과정 반복
   4-2 서브시스템을 적절한 수준의 모듈로 기능 분해

3.2.5 아키텍처 접근법 식별, 평가 및 결정

접근법이 선택된 경우 선택된 아키텍처 스타일이 요구사항을 충족하는지 분석해 보아야 한다.

5 아키텍처 접근법 식별, 평가 및 결정
   5-1 아키텍처 동인에 따른 아키텍처 스타일 선택
   5-2 아키텍처 접근법 대안 파악
   5-3 아키텍처 접근법 분석(평가)
      (1) 평가-1 (대안이 하나일 경우)
         - 품질 속성에 대한 아키텍처 접근법 평가
      (2) 평가-2 (대안이 2개 이상일 경우)
         - 품질 속성에 대한 아키텍처 접근법 비교평가
         - 아키텍처 접근법 선택
   5-4 선택된 접근법의 품질 속성 만족여부 판단
      - 품질 시나리오 재조정, 또는 아키텍처 접근법 선정 여부 판단
   5-5 아키텍처 접근법 선정

3.2.6 아키텍처 설계

각 모듈별 선택된 아키텍처 스타일에 기능을 할당하여 실체화 함으로써 아키텍처를 설계를 확장한다.

6. 아키텍처 설계
   6-1 아키텍처 스타일을 기능을 할당하여 모듈 별 아키텍처 설계

3.2.7 아키텍처 설계 평가

선택된 아키텍처 접근법에 따라 설계된 아키텍처를 평가한다. 만족하지 못하는 경우 다시 설계 단계를 반복하게 된다.

7. 아키텍처 설계 평가
   - 모듈이 충분히 분해되었는가?
   - View간 동기화가 되는가?

4. 아키텍처 설계 방법에 대한 분석

위에서 제안하고 있는 아키텍처 설계 방법의 특징 및 장점은 다음과 같다.

첫째, 기존의 아키텍처 설계 방법과 평가 방법을 수정, 보완하여 활용하고 있다. 아키텍처 동인을 결정에는 ATAM에서 사용하고 있는 품질 속성 유틸리티 트리를 활용하여 요구사항의 분석을 체계적이고 명확하게 아키텍처 동인을 찾을 수 있도록 하고 있다. 아키텍처 설계 방법은 ABD나 ADD에서처럼 시스템을 분할하고, 아키텍처 동인에 따라 뷰 타입별 아키텍처 설계를 해 나가는 방법을 활용하고 있다.

둘째, 보다 객관적인 평가 방법을 활용하고 있다. 시험을 통해 결과를 비교 평가 함으로써 기존의 검토로 이루어 지던 평가 과정보다 객관적인 평가함으로써 신뢰성을 높여주고 있다. 또한 아키텍처 접근법 평가 뿐만 아니라 설계된 아키텍처의 완정성에 대해서도 다시 한번 평가 과정을 거침으로써 아키텍처 설계의 완전성을 높이고 있다.

셋째, 설계 이후 이루어 지던 아키텍처 평가를 설계 단계로 통합하고 있다. 설계 후 모든 이해 관계자들이 모여 이루어지던 기존의 아키텍처 평가방법을 아키텍처 설계 단계로 통합함으로써 아키텍처 설계 단계의 산출물에 대한 품질을 높이고, 평가 이후 재설계에 대한 비용도 줄일 수 있는 효과가 있다.

5. 결론 및 향후 연구과제

본 논문에서는 아키텍처 설계 및 평가방법을 활용한 새로운 아키텍처 설계 방법에 대해 제안 하고 있고, 제안된 방법에 대한 기존 방법과의 차이 및 장점에 대해서도 분석하여 보았다.

향후 제안된 설계 방법을 통해 시스템의 설계를 해 봄으로써 제안 된 방법의 보완점을 찾고 설계 사례를 통해 본 방법의 유효성에 대한 연구를 계속하여야 할 것이다.

6. 참고문헌

[1] Jayatirtha Asundi, Rick Kazman, Mark Klein, "Using Economic Considerations to Choose Among Architecture Design Alternatives" Technical Report CMU/SEI. 2001

[2] Felix Bachmann, Len Bass, Gary Chastek, Patrick Donohoe, Fabio Peruzzi, "The Architecture Based Design Method" Technical Report CMU/SEI, 2000

[3] Mario R. Barbacci, Mark H. Klein, Charles B. Weinstock "Principles for Evaluating the Quality Atributes of a Software Architecture" Technical Report CMU/SEI, 1997

[4] Mario R. Barbacci, S. Jeromy Carriere, Peter H. Feiler, Rick Kazman, Mark H. Klein, Howard F. Lipson, Thomas A. Longstaff, Charles B. Weinstock, "Steps in an Architecture Tradeoff Analysis Method : Quality Attribute Models and Analysis", CMU/SEI, 1998

[5] Len Bass, Paul Clements, Rick Kazman, "Software Architecture in Practice Second Edition", Addison Wesley, 2003

[6] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, "Pattern-Oriented Software Architecture : A System of Patterns volume1",Wiley, 1996

[7] Paul Clements, Rick Kazman, Klein, "Evaluating Software Architectures : Methods and Case Studies", Addison Wesley, 2002