확장된 아키텍처 기술의 개념적 모델

박진욱° 채흥석
부산대학교
{jwpark°, hschae}@pusan.ac.kr

An Extended Conceptual Model of Architectural Description

Jin Wook Park° Heung Seok Chae
Pusan National University

요 약

아키텍처에 대한 기술은 아키텍처 평가와 검증을 위해서 반드시 필요하다. 아키텍처 기술(Architecture Description)을 위해서 아키텍처 기술언어, 아키텍처 모델에 대한 다양한 연구들이 있었다. IEEE-1471의 아키텍처 기술을 위한 개념적 모델이 아키텍처 기술에 대한 일반적인 모델로 제시되었고 이후에 아키텍처 기술에 대한 새로운 요소들이 제시되었다. 그러므로 IEEE-1471의 개념적 모델은 새로운 요소들의 개념 사이에 혼란이 존재한다. 본 논문에서는 IEEE-1471의 개념적 모델을 바탕으로 이후에 제시된 세 가지 요소를 추가함으로써 IEEE-1471의 개념적 모델을 확장하고자 한다.

1)

1. 서 론

소프트웨어 아키텍처의 명확한 기술은 소프트웨어 개발 초기 단계에 매우 중요하다. 요구사항 단계와 상세 설계 단계 사이에 아키텍처에 대한 설계가 진행되며 아키텍처를 평가하고 검증하기 위해서는 아키텍처를 기술할 수 있어야 하기 때문이다. 아키텍처를 평가하고 검증하는 것이 중요한 것은 개발 초기 단계에서 아키텍처를 평가함으로써 향후 발생 가능한 다양한 문제를 사전에 방지할 수 있기 때문이다[1].

그리고 다양한 이해 관계자들에게 시스템에 대한 이해를 돕고 상호 의사소통에 도움을 줄 수 있기 때문에 아키텍처에 대한 기술 정보는 중요하다. 예를 들어 고객과 개발자는 유스케이스와 관련된 아키텍처 기술을 통해서 서로의 정보를 교환한다[1].

아키텍처에 대한 평가와 검증을 가능하게 하기 위한 아키텍처 기술에 대한 연구는 매우 다양하다. 아키텍처를 기술하기 위한 아키텍처 기술 언어(ADL:Architecture Description Language)에 대한 연구가 있으며 아키텍처를 정의하기 위한 모델에 대한 연구도 있다. 아키텍처 기술 언어의 경우 대표적인 것으로 Wright[2], Unicon[3], ACME[4] 등이 있다. 대표적인 아키텍처 모델로는 4+1 뷰 모델[5]과 Siemen의 4가지 뷰[6], Rational의 아키텍처 기술 명세(ADS:Architectural Description Specification)[7], 그리고 SEI의 뷰 모델[1]등이 있다.

IEEE-1471[8]에서 다양한 모델을 포함할 수 있는 아키텍처 기술을 위한 개념적 모델을 제시되었다. 아키텍처 기술을 위한 개념적 모델에는 시스템, 환경, 아키텍처, 이해관계자, 아키텍처 기술, 뷰들의 요소와 그들 간의 관계가 표현되어 있다. 하지만 IEEE-1471 이후에 연구된 내용들이 반영된 통합된 형태의 아키텍처 기술을 위한 개념적 모델이 제시되지 않았기 때문에 이들 개념 간에 혼란이 야기되고 있다.

본 논문에서는 IEEE-1471에서 제시된 개념적 모델을 바탕으로 이후에 제기된 요소들을 반영하여 확장된 아키텍처 기술을 위한 개념적 모델을 제시한다.

2. 연구배경

본 절에서는 연구배경으로서 IEEE-1471에서 제시된 아키텍처 기술의 개념적 모델을 살펴본다. 그리고 이후에 제시된 개념인 SEI의 Viewtype, Style, View에 대한 개념과 Nick Rozanski가 제시한 Perspective에 대한 개념을 살펴 본다.

2.1 IEEE-1471의 기술의 개념적 모델

IEEE-1471에서는 아키텍처를 기술하기 위한 개념적인 모델을 제시한다. IEEE-1471의 의 개념적 모델은 그림 1과 같다[8].

- 시스템(System)은 환경(Environment)에 존재하며 시스템은 환경에 영향을 받는다. 환경은 시스템 개발과 관련된 요소를 결정하게 한다. 또한 환경은 시스템 시스템과 상호 동작하는 다른 시스템을 포함할 수 있으며 시스템의 범위를 결정하게 한다.
- 시스템은 하나 이상의 이해관계자(Stakeholder)를 가진다. 각 이해관계자는 일반적으로 시스템에 대한 다양한 관심사(Concern)를 가진다. 시스템에 대한 관심사는 성능이나 신뢰도, 보안과 같은 요소를 포함한다. 시스템은 하나 이상의 임무(Mission)을 이행하기 위해서 존재한다. 여기서 임무는 이해관계자가 사용할 수 있

1) "본 연구는 정보통신부 및 정보통신 연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음" (IITA-2006-C1090-0603-0032)
2) 이 연구에 참여한 연구자는 2단계 BK21 사업 』의 지원비를 받았음.

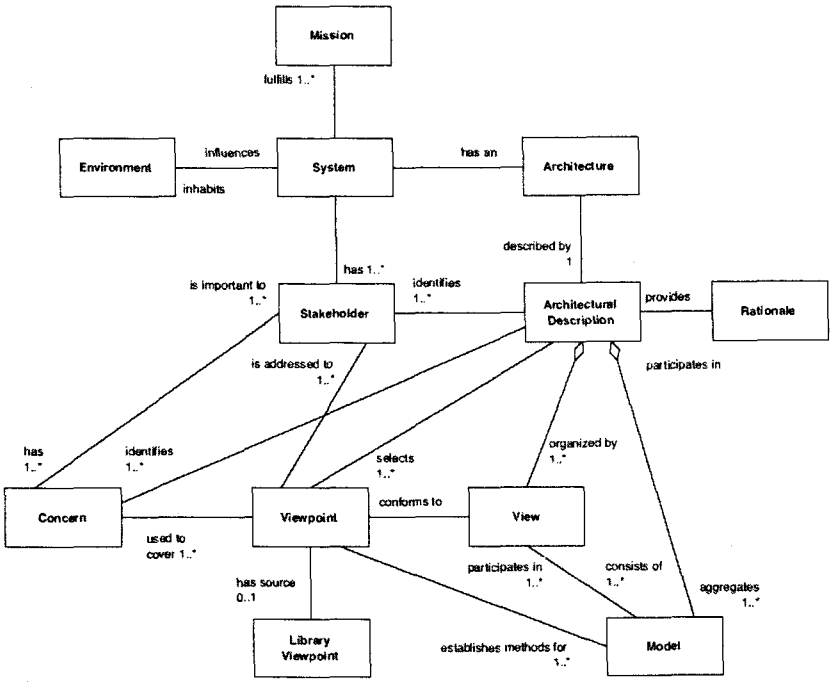

그림 1. 아키텍처 기술을 위한 개념적 모델

는 시스템의 동작을 의미한다. 그리고 모든 시스템은 아키텍처(Architecture)를 가진다.

아키텍처는 아키텍처 기술(Architectural Description)에 의해서 기술되며 아키텍처 기술은 하나 이상의 View로 구성된다. 각 View는 시스템 관계자의 하나 이상의 관심사를 나타낸다. View에서 사용하는 요소의 표현은 Viewpoint에 의해서 결정된다. 아키텍처의 View에 표현되지 않는 다른 정보들은 아키텍처 기술에 포함된다. 시스템의 Overview, 시스템 관계자와 그들의 관심사와 같은 정보가 여기에 해당된다. 아키텍처 기술은 하나의 Viewpoint를 선택하고 Viewpoint는 View에서 표현될 요소를 나타내기 위한 표현 방법을 결정한다. Viewpoint의 정의는 아키텍처 기술에서 하지만 다른 곳에서 정의될 수 있다. 이와 같이 다른 곳에 정의되는 Viewpoint는 Library Viewpoint로서 참조된다. 그리고 View는 다시 하나 이상의 모델로 구성된다.

2.2 SEI의 Viewtype, Style, View, View packet 개념

SEI에서 제안한 요소는 Viewtype과 Style, View, 그리고 View packet이 있다[1].

- Viewtype은 특정 관점에서 바라본 소프트웨어 시스템의 아키텍처를 기술하는 요소와 관계의 타입을 정의한다. IEEE-1471의 Viewpoint와 SEI의 Viewtype은 매우 유사한 개념이다. IEEE-1471의 Viewpoint 정의와 SEI의 Viewtype 정의는 다음과 같다.

"A specification of the conventions for constructing and using a view."[8] - IEEE-1471의 Viewpoint

"A viewtype defines the element types and relationship types used to describe the architecture of a software system from a particular perspective."[1] - SEI의 Viewtype

즉, View에서 사용할 요소들과 관계의 타입을 정의하는 것이 IEEE-1471의 Viewpoint이며 SEI의 Viewtype이다. SEI에서는 3개의 Viewtype을 제안하고 있으며 그 Viewtype은 Module Viewtype, Component and Connector Viewtype, 그리고 Allocation Viewtype이다.

- Module Viewtype : 시스템의 기능적인 분류를 위한 Module 중심의 Viewtype이다. Style로는 Decomposition Style, Uses Style, Generalization style, Layered Style이 있다.
- Component and Connector Viewtype : 시스템의 실행시간 기능과 컴포넌트의 관계를 표현하기 위한 Viewtype이다. Style로는 Pipe-and-Filter Style, Shared-Data Style, Publish-Subscribe Style, Client-Server Style, Peer-to-Peer Style, Communicating-Processes Style이 있다.
- Allocation Viewtype : 소프트웨어의 요소와 환경요소와의 관계를 표현하기 위한 Viewtype이다. Style로는 Deployment Style, Implementation Style,

Work Assignment Style이 있다.

- Style은 시스템의 요소와 관계를 사용하는 방법을 제약할 수 있는 시스템 요소와 관계의 특수화를 말한다. 다음은 SEI의 Style에 대한 정의이다.

"An architectural style is a specialization of element and relation types, together with a set of constraints on how they can be used."

각 Viewtype은 여러 종류의 Style로 특수화 될 수 있다. 각 Viewtype이 가질 수 있는 Style은 앞에서 설명하였다.

- View는 시스템에 대한 표현을 의미한다. SEI의 View 개념은 IEEE-1471의 View와 의미가 일치 한다. 다음은 IEEE-1471 View와 SEI View의 정의 이다.

"A representation of a whole system form the perspective of a related set of concerns." - IEEE-1471의 View

"A View is a representation of a set of system elements and relationship among them." - SEI의 View

이 두 정의에서 주목해야 하는 점은 View는 시스템을 나타내는 하나의 표현이라는 것이다. 그림 2는 Viewtype과 Style, View 간의 관계를 보여준다.

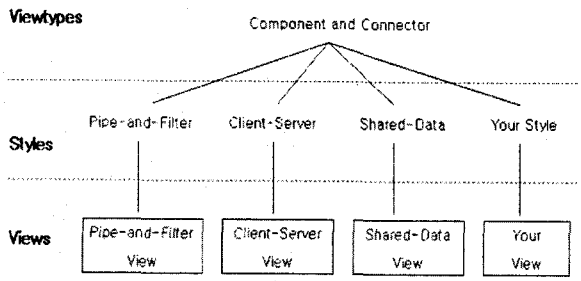

그림 2. Viewtype, Style, View:Component and Connector View의 예

- View packet은 이해관계자들이 원하는 문서화의 응집력 있는 가장 작은 단위이다. 시스템에 대한 몇몇 뷰는 매우 복잡하고 이해하기 어려울 수 있다. 복잡한 뷰를 조금 더 이해하기 쉽게 나눈 것이 View packet이다. 하나의 View는 고수준의 하나의 View packet으로 나타날 수 있으며 고수준의 View packet은 다시 조금 더 작은 View packet으로 표현될 수 있다. 같은 수준의 View packet을 형제(Sibling) View packet이라고 하며 조금 더 자세한 세부 View packet을 자식(Child) View packet이라고 한다.

2.3 Nick Rozanski의 Perspective

IEEE-1471의 아키텍처 기술을 위한 개념적 모델은 아키텍처의 품질 요소적인 부분에 대해서는 아키텍처에 나타나는지를 알 수 없다. 이를 위해서 Perspective란 개념을 추가한 연구가 있다[9]. 이 연구에서는 4+1 View의 다섯 가지 Viewpoint를 확장하여 Functional, Information, Concurrency, Development, Deployment, Operational의 6가지 Viewpoint를 제시하고 각각을 설명한다. 그러나 이런 Viewpoint만으로는 시스템의 품질요소를 나타내지 못함을 지적하고 이를 해결하기 위해서 Perspective의 개념을 제시한다. Perspective는 많은 수의 뷰를 고려하는 품질 특성과 연관된 행동, 체크리스트, 지침, 가이드라인의 집합이다. 연구에서 제시하는 Perspective는 Security, Accessibility, Performance, Location, Availability, Regulation, Maintenance Perspective 등이 있다.

3. IEEE-1471 모델의 확장

본 절에서는 2절에서 설명한 IEEE-1471의 개념적 모델을 바탕으로 하여 이후 제시된 Viewtype, Style, View packet, Perspective 개념을 추가하여 IEEE-1471의 모델을 확장한다.

3.1 View packet의 추가

2절에서 설명한 SEI의 View packet의 개념을 추가한다. 그림 3은 Veiw packet 개념이 추가된 모델을 보여준다. 새롭게 추가된 부분은 굵은 선을 이용하여 나타내었다. View packet은 매우 복잡하게 표현될 수 있는 View에 대한 이해를 돕기 위해서 나누어진 정보의 조각이다. 즉 View는 여러 개의 View packet으로 구성될 수 있다. 하나의 View packet만으로도 View를 표현 할 수 있고, 다시 하나의 View packet을 세부적으로 표현하기 위해서 여러 개의 View packet으로 나눌 수도 있다.

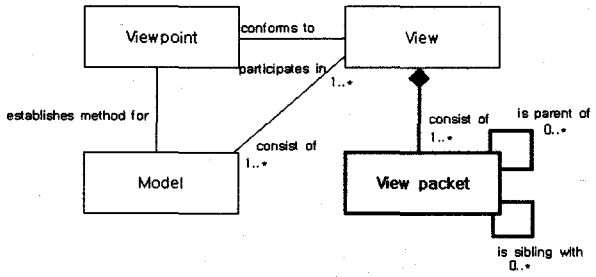

그림 3. View packet이 추가된 모델

하나의 View packet은 다른 View packet과 연관이 있을 수 있다. SEI에서 제시한 View packet 간의 연관은 parent 관계, child 관계, 그리고 sibling(형제) 관계의 세 가지가 있다. 그림 3에서는 parent와 child의 관계를 "is parent of" 관계를 사용하여 표현하고 sibling 관계를 "is sibling with" 관계를 사용하여 표현한다.

3.2 Style의 추가

SEI의 Style은 2.2절에서 본 정의에 따라서 Viewtype의 Specialization이다. 즉, Viewtype이 정의하는 요소와 관

계를 사용하여 그들의 연관이나 구성에 제약을 가한다. 그림 4는 SEI의 Style이 추가된 모델을 보여준다.

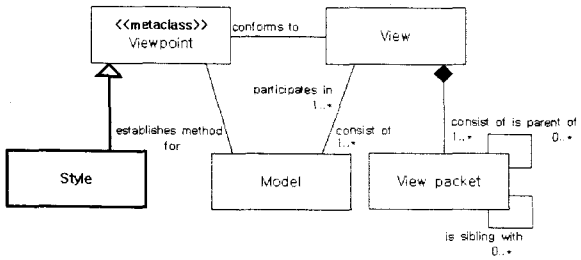

그림 4. Style이 추가된 모델

Viewpoint의 인스턴스가 각각의 Viewpoint가 되고 Style은 각 Viewpoint instance의 Specialization이 되어야 하므로 Viewpoint가 <<metaclass>>가 되었다. <<metaclass>>인 Viewpoint와 View와의 연관은 Viewpoint의 인스턴스와 View와의 관계를 의미한다. 그리고 Style은 Viewpoint 인스턴스의 Specialization이므로 Style은 Viewpoint는 통해서 View와의 연관을 가진다. 예를 들어 Module Viewpoint이면서 Decomposition Style을 사용한다면 Viewpoint의 객체인 Module Viewpoint를 상속 받는 Decomposition Style의 View가 연관되게 되는 것이다.

3.3 Perspective의 추가

3절에서 설명한 Perspective의 개념을 모델에 추가한다. 그림 5는 Perspective 요소를 추가한 IEEE-1471의 개념적 모델을 보여준다.

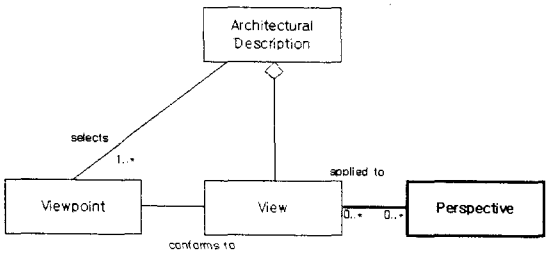

그림 5. Perspective가 추가된 모델

Perspective는 기존의 Viewpoint 모델로는 시스템의 품질 속성을 명시적으로 나타낼 수 없었던 문제를 해결하기 위해서 추가된 개념으로 여러 개의 뷰에 대해서 적용된다. 예를 들어, 특정 시스템의 아키텍처를 기술하기 위해서 4+1 뷰 모델을 사용하고 만약 성능적인 요소에 관심이 있다면 4+1 뷰의 모든 뷰에 대해서 성능적인 요소를 살펴 볼 수 있다는 것을 의미한다.

Viewpoint는 View에 사용되는 요소를 정의하고 Perspective는 다양한 View에 적용된다. 이 관계에서 중요한 것은 다중성(Multiplicity)이다. 양방향 모두 0..* 다중성을 가진다. Perspective가 여러 개의 View와 연관이 있는 이유는 위에서 설명하였다. 반대로 View가 여러 Perspective와 연관이 있는 이유는 하나의 View에 다양한 Perspective가 영향을 미칠 수 있기 때문이다. 예를 들어, 특정 시스템의 Logical View로 나타내어진 뷰에 대해서 성능 Perspective가 영향을 끼치고, 가용성에 대한 Perspective도 같은 View에 대해 영향을 미칠 수 있다.

3.4 확장된 전체 모델

View packet, Style, 그리고 Perspective 개념이 추가된 아키텍처 기술을 위한 개념적 모델은 그림 6과 같다.

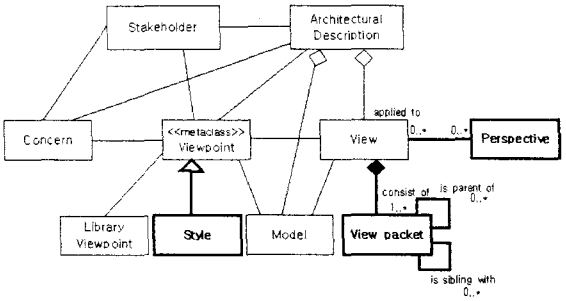

그림 6 확장된 아키텍처 기술을 위한 개념적 모델

IEEE-1471의 개념적 모델에 추가된 요소는 굵은 선으로 표시되었다. Style은 Viewpoint의 Specialization이며 Perspective는 View와 서로 연관된다. View packet은 View를 구성하며 View packet 간에도 연관이 있다.

4. 적용 예

본 절에서는 확장된 개념의 모델의 실제 적용 사례를 SEI에서 예로 들었던 ECS(EOSDIS Core System) Software Architecture View[1]에 적용하여 보여준다.

ECS Software Architecture View는 모두 10개의 View로 구성된다. 각 View의 내용은 표 1과 같다.

표 1. ECS Software Architecture의 View

View	View packet 개수
Module Decomposition	14
Module Uses	3
Module Generalization	1
Module Layered	1
C&C Pipe-and-Filter	1
C&C Shared-Data	1
C&C Communicating-Processes	10
Allocation Deployment	1
Allocation Implementation	1
Allocation Work Assignment	1

표 1의 View의 가장 앞의 단어는 해당 View의 Viewpoint를 나타내고 그 뒤의 문장은 해당 View의 Style을 나타낸다. 예를 들어 Module Decomposition

View의 경우는 Module Viewtype을 사용하고 Decomposition Style의 View이다. 그림 7은 확장된 모델에서의 Module Decomposition View를 나타낸다.

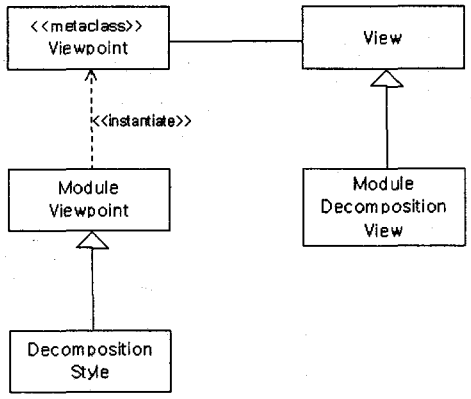

그림 7. Module Decomposition View의 예

먼저 Module Viewpoint는 <<meataclass>>인 Viewpoint의 인스턴스이다. Decomposition Style은 Module Viewpoint를 상속받으므로 Viewpoint가 가지는 View와의 연관을 그대로 가진다. 즉, Decomposition Style의 객체와 Module Decomposition View의 객체는 Viewpoint와 View와의 연관에 의해서 연결지어 질 수 있다.

표 1에서 Module Decomposition View는 모두 14개의 View packet을 가진다. SEI에서 문서화한 Module Decomposition View의 View packet의 구성은 표 2와 같다.

표 2. Module Decomposition View의 View packet 구조

1. ECS	2. SDPS	3. Client
		4. Interoperability
		5. Ingest
		6. Data Management
		7. Data Processing
		8. Data Server
		9. Planning
	10. CSMS	11. System Management
		12. Communications
		13. Internetworking
	14. FOS	

표 2에서 각 셀의 첫 숫자는 View packet 번호를 나타내고 뒤는 이름을 나타낸다. 표 2에 따르면 ECS 시스템은 1번 View packet을 통해서 전체를 세 가지 View packet으로 나누며 세 가지 View packet은 다시 각각 여러 개의 View packet으로 나누어 진다. 자신의 왼쪽 칸에 있는 View packet이 parent View packet이며 자신과 같은 열에 있는 View packet이 sibling View packet이다. 그림 8은 표 2의 View packet 구조 중 일부를 확장된 개념 모델에 적용한 것을 보여준다.

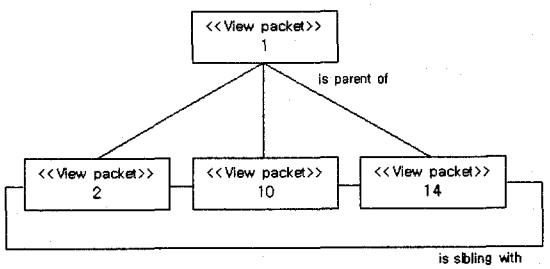

그림 8. View packet 구조의 예

스테레오타입 View packet은 해당 클래스가 View packet임을 나타낸다. 클래스 이름은 표 2에서의 View packet 번호를 의미한다. 1번 View packet은 2, 10, 14번 View packet의 parent이며 2, 10, 14번은 서로 sibling 관계가 된다.

위의 예는 같은 View안에서의 View packet 간의 관계만을 보여준다. 그러나 다른 View에 있는 View packet 간에도 sibling 연관이 가능하다. 표 3은 Module Decomposition View의 1번 View packet과 연관된 다른 View의 View packet을 보여준다.

표 3 Module Decomposition View의 View packet 1번과 연관된 다른 View의 View packet

Viewtype	Style	View packet 번호
Module	Layered	1
C&C	Pipe-and-Filter	1
Allocation	Deployment	1
Allocation	Implementation	1
Allocation	Work Assignment	1

Module Decomposition View의 View packet 1번은 5개의 다른 View의 View packet과 sibling 관계를 가지는 것을 알 수 있다. 그림 9는 이들 간의 관계를 보여준다.

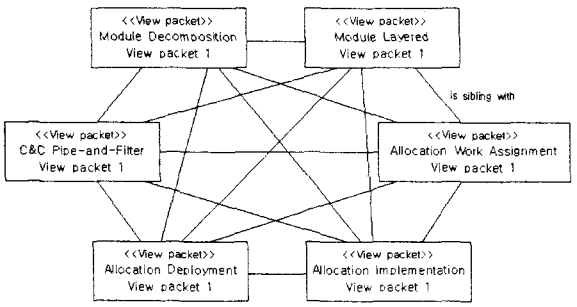

그림 9. 다른 View의 View packet 간 관계의 예

그림 9의 View packet 간 관계는 모두 "is sibling with" 이다. 즉, View packet은 같은 View 안에서 parent, child 관계, 그리고 sibling 관계를 가질 수 있으며 다른 View에 있는 View packet과는 sibling 관계만을 가질 수 있다.

5. 관련연구

본 절에서는 IEEE-1471의 아키텍처 기술을 위한 개념적 모델을 확장하는 다른 연구들에 대해서 알아본다.

IEEE-1471의 아키텍처 기술을 위한 개념적 모델은 비즈니스 프로세스 모델을 표현하기 위한 요소를 포함하고 있지 않다. 이를 위해서 IEEE-1471의 아키텍처 기술 모델에 비즈니스 프로세스 모델을 표현하기 위해 sematic model과 symbolic model, 그리고 signature를 추가하여 확장하는 연구가 있다[10]. 그림 10은 비즈니스 프로세스 모델을 표현하기 위해서 확장된 아키텍처 기술 모델을 보여준다.

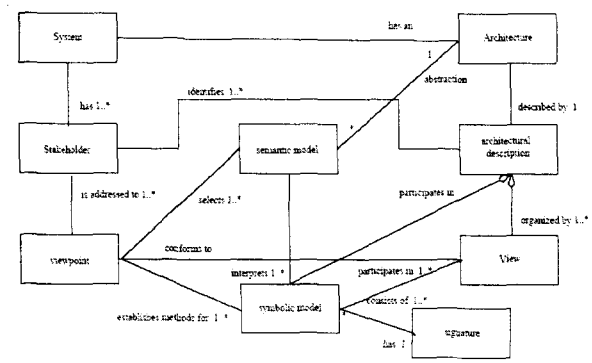

그림 10. 비즈니스 프로세스 모델을 위해 확장된 모델

그러나 비즈니스 프로세스 모델을 위해 추가된 요소는 일반적인 아키텍처 기술의 요소가 아니므로 IEEE-1471의 모델을 일반적으로 확장하지는 못한다.

6. 결론 및 향후 연구

본 논문에서는 IEEE-1471에서 제시한 아키텍처 기술의 개념적 모델을 바탕으로 하여 이후에 새롭게 제시된 View packet, Style, Perspective의 개념을 추가하여 확장된 아키텍처 기술의 개념적 모델을 제시하였다. 그리고 확장된 모델을 ECS 시스템에 적용하였다.

그러나 SEI의 Viewtype 모델을 사용하지 않는 다른 시스템의 아키텍처에 대해서도 그대로 적용할 수 있는지에 대한 연구는 진행하지 못하였다. 향후에는 4+1 뷰 모델이나 Siemen의 네 가지 뷰 모델을 사용하여 표현된 아키텍처에도 확장된 모델을 적용해 보고, 모든 아키텍처 모델이 표현 가능한 모델로 확장할 계획이다.

참고문헌

[1] P. Clements, F. Bachmann, L. Bass, D. Garlan, J. Ivers, R. Little, R. Nord, and J. Staffor, Documenting Software Architectures: Views and Beyond, Boston, MA, Addison-Wesley, 2002.

[2] R. Allen and D. Garlan, Beyond Definition/Use : Architectural Interconnection, Proceedings of Workshop on Interface Definition Languages, Portland, Oregon, USA, January 1994.

[3] J. Magee and J. Kramer, Dynamic Structure in Software Architectures, Proceedings of the 4th ACM SIGSOFT Symposium on Foundations of Software Engineering, San Francisco, CA, USA, pp. 3-14, Oct 1996.

[4] D. Garlan, Robert T. Monroe, and D. Wile, Acme: Architectural Description of Component Based Systems, Foundations of Component-Based Systems, pp. 47-68, Cambridge University Press, 2000

[5] P. Kruchten, The 4+1 View Model of Architecture, IEEE Software, 12(6), pp. 42-50, Nov 1995

[6] D. Soni, R. L. Nord, and C. Hofmeister, Software Architecture in Industrial Applications, Proceedings of the 17th International Conference on Software Engineering, Seattle, WA, pp. 196-207, 1995.

[7] E. Norris, Communicating Complex Architectures with UML and the Rational ADS, Proceedings of the IBM Rational Software Development User Conference, 2004.

[8] IEEE, Recommended Practice for Architectural Description of Software-Intensive Systems(IEEE Std 1471-2000), New York, NY, 2000

[9] Nick Rozanski and Eoin Woods, Software Systems Architecture: Working with Stakeholders Using Viewpoints and Perspectives, Addison-Wesely, 2005

[10] F. S. de Boer, M. M. Bonsangue, J. Jacob, A. Stam, and L. van der Torre, A Logical Viewpoint on Architectures, Proceeding of the 8th IEEE Enterprise Distributed Object Computing Conference(EDOC), 2004.