*IOSR Journal of Computer Engineering (IOSR-JCE)*
*e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 3, Ver. V (May-Jun. 2016), PP 39-45*
*www.iosrjournals.org*

# Tool Support for the Service Oriented Modelling

Mr.N.Ramkumar
*(Department Of Information Technology), Sri Shakthi Institute Of Engineering and Technology, Coimbatore, India.*

***Abstract:*** *The Tool Support for the Service Orientated modelling is one of the technical project and to model the changes from the existing metamodel and to define the grammar for the internal policy and the coordination and to compile in the SRML editor. Some of the major components like the wires, business protocol, business role and the transition have already been implemented in the SRML model. The requirement is the existing metamodel: in the existing metamodel few components have not been implemented. The components are the part of the interaction protocol i.e. Coordination and the external policy. The main task of my project is to implement these two properties in the existing metamodel.*
*The other part of the project is to define the grammar for the coordination and the interaction protocol and to compile in the SRML editor. The grammars are defined in such way that the user can edit and can view the changes in the editor. By using the defined rules we can generate the metamodel using the eclipse tool*
***Keywords:*** *SRML Editor, Metamodel*

## I. Introduction

The SENSORIA (Software Engineering for Service Orientated Overlay Computers) aims to develop novel and comprehensive approach for engineering service orientated computations. The main issues of the SENSORIA are the development of service specification like design and reconfiguration of service based architectures. In this setting, the system consists of the components and the interconnections along with the current state. [1]

SRML is the modelling language that the number of addresse the qualitative and quantitative support which is based on the mathematical foundation.SRML offers for modelling business services and activities and on the methodological approach that SRML supports.SRML addresses Service Orientated Computing (SOC) as a new technologies in which interactions are no longer based on fixed or programmed exchanges of products with particular parties.which is known as clientship in the object orientated programming but on the provisioning of services by external providers that are procured on the fly subject to a negotiation of service level agreements. More precisely, the processes of discovery and selection of services as required by an application are not coded in the design time but performed by the middleware according to functional and non functional requirements. (SLAs).

One of the key concern in defining the SRML was precisely the need to distinguish between these two different modes of composition. In SOC, we decided to follow the basic principles and structures put forward by the Service Component Architecture. However, SCA addresses low level design in the sense that it provices an assembly model and binding mechanisms for service components and clients programmed in specific languages. Instead, we find SRML are primitives that address high level design and support of shift of emphasis from programming to modelling from component interoperability to business integration.

SRML is a technology agnostic in the sense that it does not commit to any specific language or platform for programming and composing services. SRML offers for high level business design by the traditional use case diagrams. The use case diagram can be extended so as to support the specificities of service- oriented software engineering.

In this paper, the requirement is that the existing metamodel, in the existing metamodel, a part of the interaction protocol i.e. Co-ordination and the external policy was not implemented yet. The main task of the project is to extend these two properties in the existing metamodel. The other task of the project is transformation from SRML models to business process languages using the BPEL and also to improve the graphical representation of SRML using the visual graphical editors like Eclipse.

In order to extend the metamodel, the rules are defined as grammar (called XTEXT Grammar) is used to implement the co- ordination and the external policy. In order to transform from SRML models to business process, the language called Business Process Execution Language will be used.

## II. Objective

1. **Extending the** existing Metamodel:
The above step is achieved in the given time duration.

**2. Transformation from SRML models to business process languages:**

Due to the time duration, the given information about the background knowledge about the BPEL and SRML so that it will be useful for the future reference and the implementation of the SRML to BPEL will be considered for the future work.

**3. To improve the graphical representation of SRML Metamodel:**

Due to the time duration, the background knowledge about the graphical representation and concept about GMF so that it will be useful for the future reference and the implementation of the graphical representation will be considered for the future work

The main purpose of the project is to implement the interaction protocol and the external policy in the existing metamodel using the xtext grammar. In the interaction protocol, only the coordination part is not yet implemented. The main objective is to implement the coordination part. The other objective is to make a research on the case study of the existing example (Mortgage).The other objective is transformation from SRML models to business process languages using the BPEL and also to improve the graphical representation of SRML using the visual graphical editors like Eclipse.

Metamodelling is the construction of collection of things, terms etc within a certain domain. A metamodel is yet another abstraction, highlighting the properties of the model itself. A model conforms to its metamodel in the way that a computer program conforms to the grammar of the programming language. [2]

Common uses for metamodel are as a schema for semantic data that needs to be exchanged or stored, language that supports a particular method or process, as a language to express additional semantics of existing information. Metadata modelling is a type of metamodelling used in software engineering and systems engineering for the analysis and construction of models applicable and useful to some predefined class of problems.

In Software engineering, several types of models can be distinguished they are:
Metadata Modeling, Meta-process modelling, Executable metamodeling and Model Transformation Language.

Why extending the metamodel?

The status of the SRML metamodel could be defined already as a prefinal, the development of the SRML editor is ongoing. Until, now only the SRML modules can be represented graphically while the specification of Business Roles, Business Protocols and interaction protocols still have to be done in tree-based view. Generating the extended properties views in GMF is still under development, so that the generated editor code may be extended by hand to provide user-friendly editing support of certain features. In order to overcome some of the above said features, the metamodel is extended. [8]

In this paper, already created metamodel is taken into consideration and going to extend the existing metamodel i.e.Ecore Metamodel. The Ecore metamodel is a powerful tool for Model driven architecture. Typically if we take an application we would define the objects, their attributes and their relationship. We can also define the specific operation that belongs to those objects using the Eoperation model element. By default, EMF will generate the method signatures, for those operations but we have to go back and implement those operation often changing the code in similar logic time.[2]

To add some sort of arbitrary implementation behaviour in the model, one approach is to add the text based annotations of type Eannotation to model the objects and to interpret those changes in templates in the code generation. However our aim is not to validate model elements but rather to model implementation itself, in order to reuse those metamodel elements with any concrete model need to extend the metamodel.[2]

There are several methods to extend the existing metamodel, adding the grammar in the existing metamodel gives the better result. So by following this methodology so as to get the better outcome.

In the recommended part is to develop the transformation from SRML models to some business process execution language. So far there are only transformation of BPEL to SRML is there but in this project the transformation from SRML model to BPEL is to be implemented

**To improve the graphical representation of the SRML in visual editors like Eclipse:**

Earlier the graphical representation does not show better result, there were no specific tool to implement the models will be messed without any order and will not give the clear picture, so in order to improve the graphical representation to use the Eclipse visual editor project is to advance the creation, evolution, promotion of the eclipse visual editor platform, and to cultivate both an open source community and an ecosystem of products, capabilities and services. The most obvious tool that the visual editors draws upon is the Graphical Editing Framework (GEF) As part of the visual editor it uses the Eclipse Modelling framework behind the scenes to map among the model, a java class, and the graphical representation.

## III. Literature Survey

Xtext is use to create a small domain specific Language or it is used to create a general purpose programming language. With Xtext create our own languages. Xtext is use to create an Eclipse-based development environment providing editing experience from modern Java IDEs in a short amount of time. So Xtext grammar rules are used in the eclipse to generate the metamodel [3].

The Eclipse is an open source software development project to develop a wide range of exemplary extensible development tools and the tools specific components for the Eclipse Platform. The Eclipse is used to generate the class diagram. It is one of the efficient tools.

## IV. Technologies

***SRML:***

SRML is a modelling language for service oriented systems. It operates the higher level of abstraction of business modelling. It provides the semantic modelling that are independent of languages and platforms in which services and program are executed.SRML is a prototype modelling language, methodological approach to service orientated systems. SRML relies on the mathematical domains of service composition and reconfiguration SRML abstracts from the typical mechanisms made available by service oriented middleware such as sessions and event or message correlation, as well as the brokers that are responsible for the discovery and binding of services. A formal computation and coordination model was developed for SRML over which qualitative and quantitative analysis techniques were defined using the UMC model checker and the PEPA stochastic analyser. [1]

***XTEXT:***

XTEXT can be use to create our own languages. It is a decent tool support to use XTEXT to create a sophisticated Eclipse based development environment. It provides editing experience from java IDE in a short period of time. TEXT is otherwise called as Language development framework. XTEXT is a professional open source project. XTEXT provides with the set of domain specific language and modern API to describe the different aspect of the programming language. XTEXT also gives the full implementation of that language running on the JVM. The grammar language is the cornerstone of the Xtext. It is a domain specific language, carefully defined for the description for the textual languages. The main idea is to describe the concrete syntax and how it is mapped to an in memory model created during parsing [3]

Xtext uses the lightweight dependency injection framework for the googe guice to wire up the whole language as well as the IDE infrastructure.Xtext comes with default implementations and DSLs and APIs for the aspect that are common spots for customization. Google Guice gives you the power to exchange every littles class.[3]

Xtext is a professional Open Source Project. Xtext is an Eclipse.org.project. Besides many other advantages this means that not to worry anything about the IP issues. Because the Eclipse Foundation had their own layers who take care that no intellectual property is violated.[3]

***ECLIPSE MODELLING FRAMEWORK:***

EMF is an integral part of the Eclipse platform, as well as a cornerstone of related technologies and frameworks, such as the eclipse visual editors.SOD and UML amny of which are integrated into IBM platforms like Rational Application Developer and websphere Business Modeler. In recent years, EMF has grown to encompass java technology features such as enumerated types, annotations and generics.[2]

EMF is a modelling framework and code generation facility for building tools and other applications based on a structured data model. It provides the run time support to produce the java classes for the model, a set of adapter classes that enable viewing and command based editing of the model. Models can be specified using Java, XML documents or modelling tools like Rational Rose, then import into EMF. The key point about EMF is it provides the foundation for interoperability with other EMF based tools and applications.[4]

***XPAND:***

XPAND is also part of Eclipse and ships with its own documentation.The second workflow component is an instance of org.eclipse.xpand2.Generator, which is the MWE2 facade to the Xpand template language. The Xpand generator needs to know which template to invoke for which models. Qualified names in Xpand are separated by a double colon. In Xpand there is a file statement which refers to outlets.

XPAND is statically typed template language featuring. It works very similar to the incremental java compiler in the eclipse. XPAND is used for functional extensions, model transformation, model validation and much more. It has editor features like syntax colouring, error highlighting, and navigation and the code completion. XPAND was originally developed as a part of the open architecture ware project

before it became a component under eclipse.[3]

***BPEL:***

Business Process Execution Language (BPEL) is an XML based language for the formal specification of business processes and business interaction protocols. It defines the notation of business process behaviour based on the web services. Business process can be divided into two categories one is executable business processes and the other type is Business protocol. BPEL is used to model the behaviour of both executable and abstract processes. The scope includes sequencing of process activities, correlation of messages.[5] BPEL is often associated with Business Process Management Notation (BPMN), which also seeks to streamline the BPM modelling process. Unlike BPEL, BPMN is not executable and so is mostly used for planning and design. BPMN, though, has a visual component that makes it easier to understand their own visual notation for BPEL to further simplify the language.[14] BPEL and BPMN have grown in popularity together over the last few years as each seeks to simplify business process management and encourage collaboration between business people and developers. But translating from one to the other remains a challenge. [14]

## V. Methodology

### *5.1 ECORE METAMODEL :*

The Ecore model or the meta model of a textual representation describes the structure of its abstract syntax tree. Ecore models are declared to be either inferred from the grammar or imported.[7] The Ecore metamodel is the powerful tool for designing model driven Architecture which can be used as a starting point for the software development. Typically would define the objects in our applications, their attributes and the relationships. It defines the specific operations that belong to those objects. Those operations are added using the Object EOperations. By default the EMF will generate skeletons , or method signatures for those operations, but have to go back and implement those operations often recoding similar logic and time. [7] The term Ecore is a modelling framework for Eclipse. According to the Eclipse foundation, the core EMF framework includes a metamodel (ecore) for describing models and run time support for the models. In other words, ecore defines the structure of the models Developers use to maintain application data. [7]

**Creation of the ECORE Diagram for the existing metamodel:**

Considering the Existing model (TravelBooking) to run the file SRML Ecore in the Eclipse tool in order to generate the Ecore Diagram (metamodel). The Ecore diagram will look like tree like structure which is used to check the properties defined in the class diagram. Whereas it can generate the class diagram in the eclipse but the best way is to generate the ecore diagram. The Ecore metamodel can be generated using eclipse.Metamodel is generated by using the Eclipse Modelling Framework and by loading the file to get the tree like structure.

In the existing grammar, the changes are made i.e. adding the external policy of the above grammar and the ecore diagram is generated. The external policy and their properties in the ecore diagram is viewable. The properties which are name, SlaVariables and the constraints.

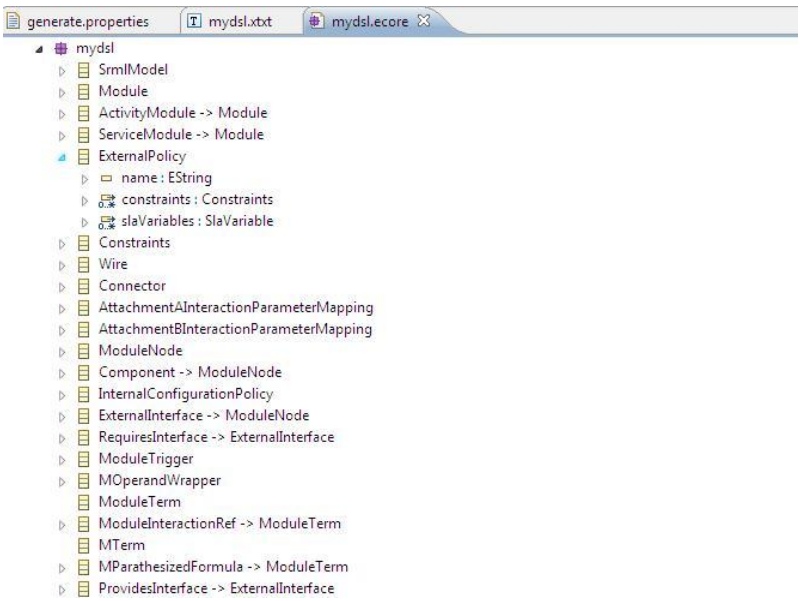


**Fig: 1** *SRML Model*

After implementing the rules of the grammar, the external policy in the above ecore diagram can be viewed, in the diagram the properties and their name, SlaVariables and the constraints. The name is defined as a string.

After the evaluation of the metamodel the below two diagrams are viewed. One for the Interaction protocol and external policy.

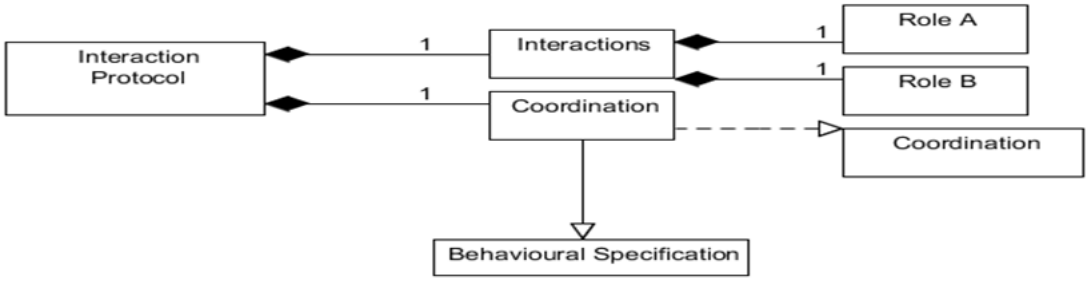


**Fig :2** MetaModel

The above metamodel shows the representation of the interaction protocol where it contains the interaction and the coordination. The interactions consists of Role A ,Role B and the Coordination. After defining the set of rules of the grammar used to generate the metamodel using the eclipse tool, but the metmodel generated by the tool will be very large so metamodel is drawn using the UMLet tool

The above fig shows that the each instance of type Interaction protocol seems to contain an instance of type Interactions and the coordination. The black diamond represents the composition. It is placed in the interaction protocol class because it is the interaction protocol that is composed of the interaction and the coordination. From the metamodel it is understood that the interaction and the coordination knew about the interaction protocol. Composition relationship is the strong form of containment or aggregation. Aggregation is the whole part relationship.

Similarly, an instance of the type Interaction seems to contain an instance of the type Role A and Role B. There is an association represented between the coordination and the Behavioural Specification. In the diagram the association has an arrowhead to denote the Behavioural Specification does not know anything about the coordination. The dashed arrow between the Coordination and the coordination are seen in the diagram. This is the dependency relationship. The coordination somehow depends on the coordination.

The cardinalities are the interaction Protocol represents the one interaction and the one coordination. The interaction represents the one role a and one Role B.In other words it is one to one relationship.

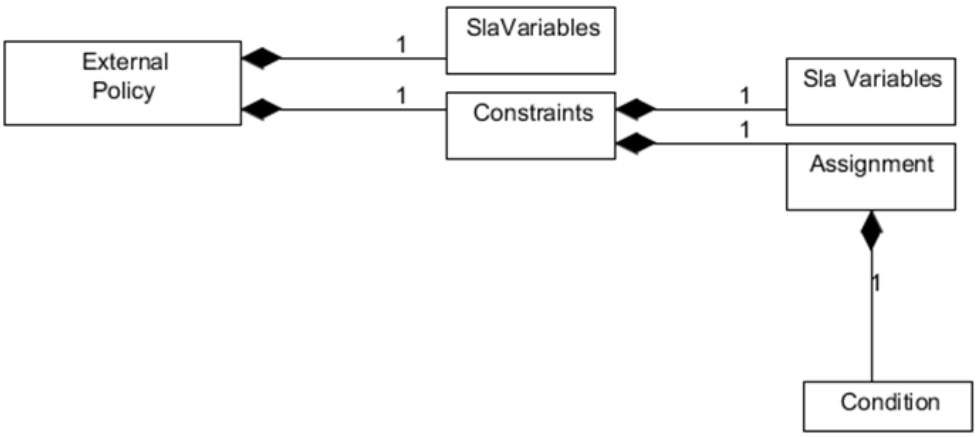


**Fig :3** Extended Metamodel

The above metamodel shows the representation of the external policy. The external policy consists of slavariables and the constraints. The constraints consist of the Sla variables and the assignment. The assignment consists of conditions. The conditions are defined in the grammar.

The above fig shows each instance of the type External Policy seems to contain the instance of the type Sla Variable and the constraints. This relationship is known as Composition. The black diamond represents the composition and the other side of the relationship is blank just the line so it is understood that the constraints and the sla variable knew about the external policy. Similarly, the instance constraint seems to contain the

instance of the type Sla Variable and the Assignment. And these instances are represented as a compositional relationship. The instance Assignment has an instance of the condition and also denoted as a composition relationship. Where the condition knew about the assignment since it is represented as normal line without any arrowhead.

In the above metamodel, the External policy class instance will always have at least one SlaVariables and one constraints. Because the relationship is called the composition relationship, when the External policy is removed or destroyed, the slavariables and constraints class will automatically removed or destroyed as well. Another important concept about the composition aggregation is that the part of the class can only be related to one instance of the parent class. Likewise, the constraints class instance will always have atleast one SlaVariables and one Assignment. This relationship is a composition relationship. Hence the cardinality is also mentioned as one. When the Constraints is removed or destroyed from the model then the class SlaVaribles and the Assignment will automatically will be removed.

The assignment is further associated with the conditions. The assignment class will always have one class which is condition. If the Assignment class is removed then obviously the Condition class will also be removed. There are several conditions which will be specified in the condition class.

The associations represent relationships between instances of types. The interpretation varies with the perspective. Conceptually they represent the conceptual relationships between the types involved. In specification these are responsibilities for knowing, and will be made explicit by access and update operations. A more implementation interpretation implies the presence of a pointer. Thus it is essential to know what perspective is used to built a model in order to interpret it correctly.[5]

Associations may be bi-directional can be navigated in either direction or uni-direction can be navigated in one direction only. Conceptually all associations can be thought of as bi directional associations are important for the specification and implementation models. For specification modules the bi directional associations gives more flexibility in navigation but incur greater coupling. In implementation models a bi-directional association implies coupled set of pointers, which many designers find difficult to dealt with.Often those who use bi-directional associations have notation to indicate a unidirectional when needed. With a bi-directional association the word role represents a single direction. Thus, a bi-directional association has two roles but a uni-directional association would have only one.[5]

One of the key aspects of association is the cardinality of an association sometimes called the multiplicity. This corresponds to the notation of mandatory, optional, 1-many, many to many relationships in the Entity- relationship approach. Each method uses a particular notation to indicate the cardinality. The cardinality is specified for each role in the association.Aggregation relationships are introduced by many methods. These are represented as a part or whole relationships. It is difficult to define the difference between an aggregation and an association or to indicate whether the distinction is useful.[5]

In class diagram, a generalization relationship is a relationship in which one model element is based on the other model element. The child is based on the parent. Generalization relationships are used in class, component, deployment and the use case diagrams. In order to comply with the UML semantics, the model elements in a generalization relationship must be the same type. For example the generalization relationship can be used between actors or between use cases, it cannot be used between an actor and a use case. We can add generalization relationship in order to capture the attributes, operations and relationships in a parent model element and then reuse them in one or more child model elements. Because the child model elements in generalizations inherit attributes, operations and relationships of the parent. The parent model element can have one or more children, and any child model element can have one or more parents. It is common to have a single parent model element and multiple child model elements. Normally generalization relationships do not have names.[5]

The realization is the one of the relationship between two models elements, in which one model element realize the behaviour that the other model element specified. A realization is indicated by a dashed line with an unfilled arrowhead towards the supplier. Realizations can only be shown on class or component diagrams.

A realization is a relationship between classes, interfaces, components and packages that connects a client element with a supplier element.

The Dependency is a weaker form of relationship which indicates that one class on another because it uses it at some point of time. Dependency exists if a class is a parameter variable or local variable of a method of another class.

## VI. Conclusion

In the way to conclude, the implementation of the metamodel satisfies the primary objective of the project like extending the existing metamodel and defining the grammar for the interaction protocol and the external policy. The primary objective is to implement the external policy and the interaction protocol in the

existing metamodel. After defining rules of the grammar, run the grammar in the editor in order to generate the ecore diagram and the corresponding metamodel. The grammar is defined in a such a way that the user can edit and change the grammar effectively. Hence the user can edit the grammar using the eclipse tool. The information about the background technologies of the BPEL and SRML are provided for the future reference

The remaining work like transformation from SRML models to Business Process Execution Language (BPEL) and to improve the graphical representation of SRML metamodel are added to the future work.

## References

[1] Victoriya Dessler, , Service-Oriented Architecture for Smart Environments, IEEE ,6th International conference on service oriented computing and Applications , 2013,99-104.
[2] **[**W.Dai et al , Service oriented Architecture in Industrial Automation Systems, IEEE, VOL-11, 2015, 771-781
[3] **[**Abdelfattah El-Sharkawi, Ahmed Shouman, Sayed Lasheen, Service Oriented Architecture for Remote Sensing Satellite Telemetry Data Implemented on Cloud Computing, IJITCS Vol. 5, No. 7, June 2013, 12-26
[4] **[***Said Nabi, Saif Ur Rehman, Simon Fong, Kamran Aziz, A Model for Implementing Security at Application Level in Service Oriented Architecture, Journal of Emerging Technologies in Web Intelligence*, Vol 6, No 1 ,(2014), 157-163,.
[5] Feng Wang, Liang Hu, Jin Zhou, and Kuo Zhao, A Data Processing Middleware Based on SOA for the Internet of Things, *Journal of Sensors*,*Volume 2015 (2015), Article ID 827045, 8 pages*

**BOOKS**

[1] Laura Bocchi, Yi Hong, Antónia Lopes, and José Luiz Fiadeiro From BPEL to SRML: A Formal Transformational Approach Department of Computer Science, University of Leicester University Road, Leicester LE1 7RH, UK .
[2] BPEL Tutorial, http://searchsoa.techtarget.com/tutorial/BPEL-tutorial
[3] Richard C.Gronback Eclipse Modeling Project U.S.A Addison Wesley December 8, 1997