International Conference on Information Engineering, Management and Security
2015 [ICIEMS 2015]

| | | | |
|---|---|---|---|
| **ISBN** | 978-81-929742-7-9 | **VOL** | 01 |
| **Website** | www.iciems.in | **eMail** | iciems@asdf.res.in |
| **Received** | 10 - July - 2015 | **Accepted** | 31- July - 2015 |
| **Article ID** | ICIEMS031 | **eAID** | ICIEMS.2015.031 |


# Software Application Generator: An ER Model-based Software Product Building Tool

Mr. Souradeep Sarkar[1], Mr. Debasish Hati[2], Mr. Prasun Kumar Mitra[3]
[1,2,3] Lecturer, CST Department, Technique Polytechnic Institute,
Hooghly,WB-712102,India

***Abstract:*** *Software Application Generator (SAG) is a very powerful software product building tool. It helps developer to build an entity relationship model based software product using modern web technology. ER model is one of the most popular methodologies for designing relational database. Asp.net, JSP and PHP are most popular technology for developing web application. Several commercial products have been developed to support ER-model and those web technologies. Inspired by these technologies, we have developed an educational prototype SAG that supports the following task:*

- *Drawing ER diagram visually and translate it to relational database schema automatically.*
- *Develop different type of template easily from ER model.*
- *Develop skeleton code of selective technology automatically.*
- *Develop skeleton test plan automatically.*
- *Deploy generated code to corresponding environment automatically.*

*In this paper, we describe the architecture of SAG and its implementation details to illustrate how such tool can be developed.*

***Keywords:*** SAG, ER modeler, Application Composer, Application Generator, Test plan Generator, Application Deployment.

## I. INTRODUCTION

SAG is very powerful tool for developing a software product based on ER models and web technologies. It contains five non overlapping modules: ER modeler, Application Composer, Application Generator, Test plan Generator and Application Deployment. It is implemented in java, therefore it can be executed in all environment when a java virtual machine available.
**A. ER Modeler:** This module is used to design ER diagram and translate it to a relational database schema. It contains two essential components: entity and relationship. Entity represents object that are involved in enterprise such as student, professor and course in a university. Relationship represents the associations among these objects such as student takes course. ER model supports the following set of attractive features:

- Representing ER-Diagram in Binary Repository: We have defined binary repository to convert an ER diagram in binary repository format and vice versa.



**Cite this article as:** Souradeep Sarkar, Debasish Hati, Prasun Kumar Mitra. "Software Application Generator: An ER Model-based Software Product Building Tool." *International Conference on Information Engineering, Management and Security (2015)*: 189-194. Print.

- Draw ER-Diagram semantically: It recognizes ER diagram components such entity sets and relationship sets semantically. This greatly facilitates the diagram to change the layout of an ER diagrams.
- Validation Verification: It supports the verification of validation of an ER diagram and ensures that only well formed ER diagram can be exported in binary file and translated to relational database schema.
- Automatic translation to relational models: After an ER diagram is created; one can simply translate the ER diagram to a relational database schema.

**B. Application Composer:**

Application Composer module is used to develop template which contain elements of web forms. We consider five types of templates: Login, Role selection, Maintain, Master detail and Associate. Login template is a general purpose template which mainly use for user authentication. Login template contains one textbox (for user name), one hidden textbox (for password), one button (for submit) and one label (for showing error message) elements. Role-Selection template is used for select the role of user. Role-Selection template contains two textboxes (for user ID and user name), one Drop Down List (for role of user), two buttons (for submit and exit) and one label (for showing error message). Maintain template is mainly used for perform basic operations like insert, update, delete and view. Maintain template contain two type of elements: entity element and form element. Entity element contains a textbox and a label for each attribute of entity of ER model which will be selected by developer. Form element contains one datagrid (for display table value), four buttons with form option true (for insert, update, delete and view operation), one button (for exit) element. Master detail template is used for update details of master. The criteria for build master detail template is that there will be two entity and a relationship between them and a common attribute among them. Master detail template contains two type of elements: entity element and form element. Entity element contains a textbox and a label for each attribute of target entity of ER model which will be selected by developer. Form element contains one datagrid (for display table value), five buttons (for submit, add to grid, delete from grid, ok and cancel ) elements. Associate template is used for update the associate table from original table. The criteria for build associate template is that there will be three entities and two relationships between them and two commoun attributes for each of two end entities with target entities. Associate template contain two type of elements: entity element and form element. Entity element contain a textbox and a label for each attribute of target entity of ER model which will be selected by developer. Form element contain two listboxes (for contain the value of original and associate table), four buttons (for add to list,remove from list,ok and cancel ) elements. Application Composer supports the following set of attractive features:

- Representing Template in XML file: We have defined a XML schema to convert template into xml format and vice versa.
- Design Template Semantically: Application Composer recognize template component such as entity elements and form elements. Entity element will be derived from the entity of ER model which are build by ER modeler. Form elements are general purpose web form elements, which will be selected by developer. This greatly facilitates the design to change the layout of a template.
- Validity Verification: Application Composer supports the verification of validity of a template and ensures that only well form template can be exported in xml file.

**C. Application Generator:**

Application Generator is used to generate code. Befor generate code, first select xml file which contains all templates. This xml file is constructed by Application Composer. Then select the web technology in which envornment the code will be generated. In ASP technology, we generate one .aspx and one .aspx.vb file for each login, role selection, master detail and associate template. For maintain template five .aspx and five .aspx.vb for main, insert, update, delete, view page. For menu page one .aspx and one .aspx.vb file will be generated. One .vb file will be generated for database connection class in App_Code folder. One configaration file will be generated by default.

**D. Testplan Generator:**

Testplan Generator is used to generate testplan. Before generate testplan, select xml file which contain all templates. This xml file is constructed by Application Composer. It generate one each document that contains one table for template or menu page. One row of table contain a test case for each element of template. It maintain the standard format of test plan.

**E. Application Deployer:**

Applicatin Deployer is used to deploy the code which is generated by Application Generator. It helps the user to deploy the code automatically. First user select the code which will be deployed. Next they select the server in which the deployment will be done. Then the deployment process will be done and the browser will be opened with default url.

**Cite this article as:** Souradeep Sarkar, Debasish Hati, Prasun Kumar Mitra. "Software Application Generator: An ER Model-based Software Product Building Tool." *International Conference on Information Engineering, Management and Security (2015)*: 189-194. Print.

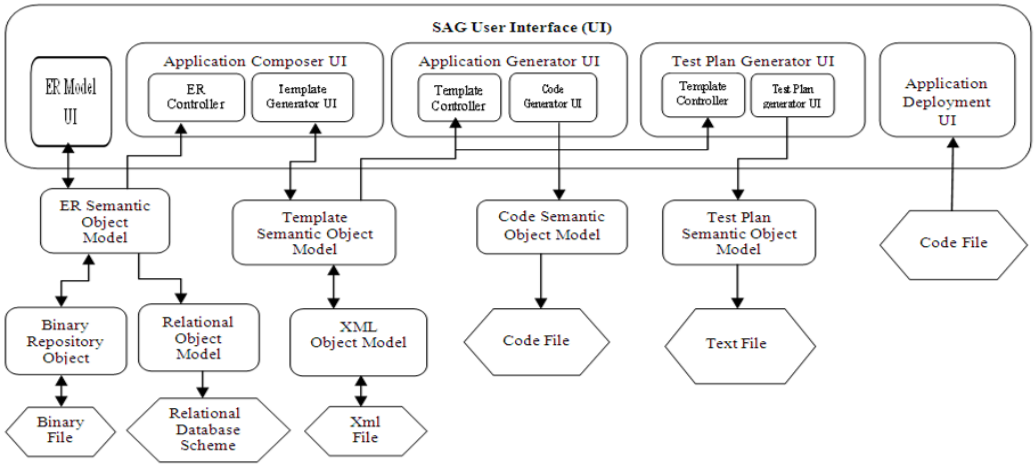


Figure 1. System Architecture of SAG Tool

## II. SYSTEM ARCHITECTURE

The overal system architecture of SAG Tool is illustrated in figure 1. Basically, it consists of the following modules.

- **ER Model User Interface:** It provides a user friendly graphical user interface to support the interaction between ER diagram designers and ER Modeler.
- **ER Semantic Object Model:** It is the key internal data structure that represents the complete semantic information of an ER diagram. An ER sementic object model is created either for ER diagram or from a binary file.

**Application Composer User Interface**: It provides a user friendly user interface to support the interaction between developer and application composer. It is divided into two sub modules : ER Controller and Template Generator User Interface. ER Controller is used to manipulate the existing ER diagram.Template Generator User Interface is used to manipulate the templates of application.

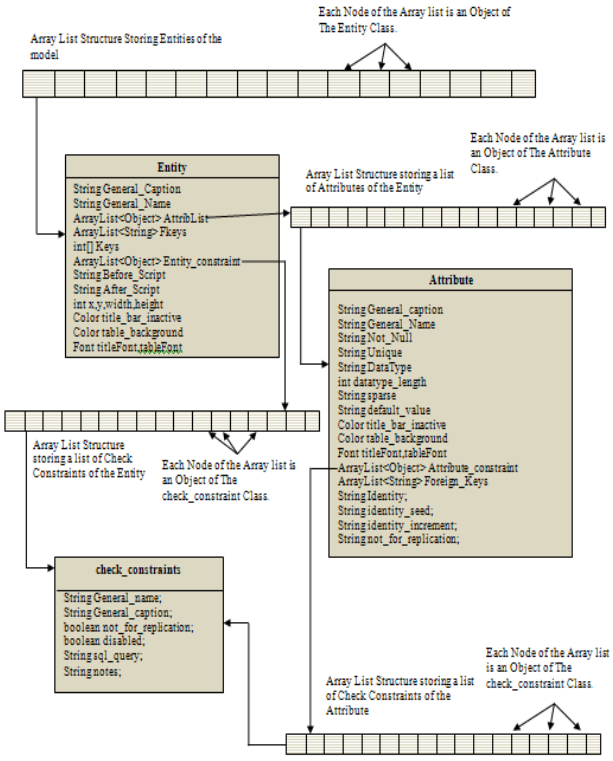


Figure 2. Local Data Structure for storing entity collection.

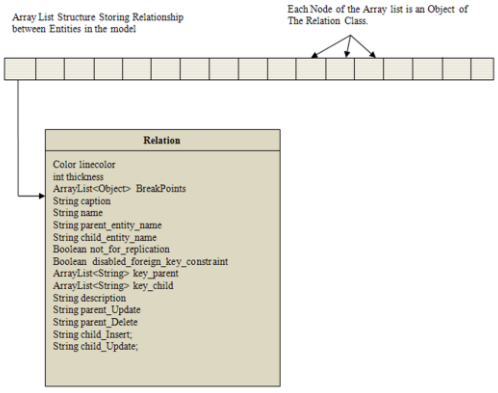


**Figure 3. Local Data Structure for storing relationship collection.**

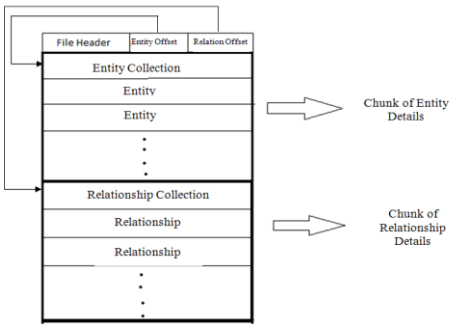


Figure 4. Data structure of storing the ER Model

**Cite this article as:** Souradeep Sarkar, Debasish Hati, Prasun Kumar Mitra. "Software Application Generator: An ER Model-based Software Product Building Tool." *International Conference on Information Engineering, Management and Security (2015)*: 189-194. Print.

- **Template Semantic Object Model:** It is the key internal data structure that represents the complete sementic information of templates of application. Template semantic object model is created either from application composer or from an XML file.
- **XML Object Model:** It is the intermediate data structure that map sql and Template semantic object model to XML file. It is used to generate a XML file or is constructed from a XML file.
- **Application Generator User Interface:** It provides a user friendly user interface to support the interaction between developer and application generator. It is divided into two sub modules : Template Controller and Code Generator User Interface. Template Controller is used to manipulate the existing templates of application. Code Generator User Interface is used to manipulate the codes of application.
- **Code Semantic Object Model:** It is the key internal data structure that represents the complete semantic information of codes of application. Code Semantic Object Model is created from application generator. It is used to generate a code file.
- **Testplan Generator User Interface:** It provides a user friendly user interface to support the interaction between developer and testplan generator. It is divided into two sub modules : Template Controller and Testplan User Interface. Template Controller is user to manipulate the existing templates of application. Testplan Generator User Interface is used to manipulate the testplans of application.
- **Testplan Semantic Object Model:** It is the key internal data structure that represents the complete semantic information of testplans of application. Testplan Semantic Object Model is created from testplan generator. It is used to generate a text file.
- **Application Deployment User Interface:** It provies a user friendly user interface to support the interaction between deployer and Application Deployment.

## II. IMPLEMENTATION

### A. ER Modeler:

To support interoperability between different ER diagram we have defined data structure to convert an ER diagram into memory object. The data structure of entity and relationship is shown in Figure 2 and Figure 3 respectively. The data structure of storing the ER Model is shown in Figure 4.
To translate an ER diagram to a relational database schema, ER Modeler follow the following steps:

a) For each entity set A implement a table using CREATE table SQL command.
b) For each relationship implement a ALTER table SQL command to set foreign key concept.

### B. Application Composer:

To support interoperability between different Template generating tools, we have defined a XML schema defination to templates of application into XML file and vice versa. The following features is maintained by the application composer.

- Login and role selection template is general pupose template. It is fixed for all applications.
- Each Maintain template will be generated for each Entity.
- Each Master detail template will be generated for two entities and a relationship between them.
- Each Associate template will be generated for three entities and two relationships between them.

### C. Application Generator:

To support interoperability between different Code generation tools, we have defined a data structure to convert templates of application into corresponding code. The data structure is shown in Figure 5.To generate code the following points will be considerd

- Code will be generated according to selected technology.
- Code will be generated for each template.
- Code for menu template will be generated by default.
- Code will maintained standard skeleton. Programmer can run this code in any environment by minimum change.
- Some additional code file will be generated by default depending on web technology.

**Cite this article as:** Souradeep Sarkar, Debasish Hati, Prasun Kumar Mitra. "Software Application Generator: An ER Model-based Software Product Building Tool." *International Conference on Information Engineering, Management and Security (2015)*: 189-194. Print.

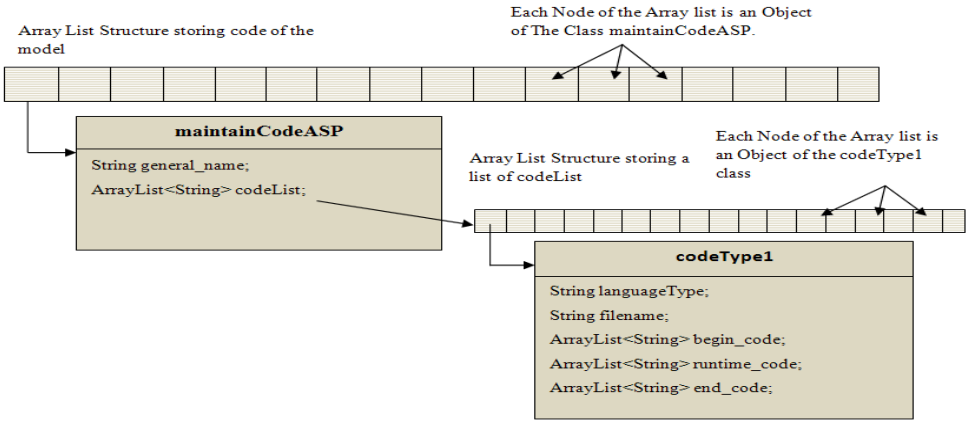


Figure 5. Local Data Structure for storing code of application .

**D. Testplan Generator:**

To support interoperability between different Testplan Generator tools, we have defined a data structure to convert templates of application into corresponding testplan. The data structure is shown in Figure 6.To generate testplan the following points will be considerd

- Testplan will be generated for each template.
- Testplan for menu template will be generated by default.
- Testplan will maintained standard skeleton. Developer can generate final testplan by minimum change.

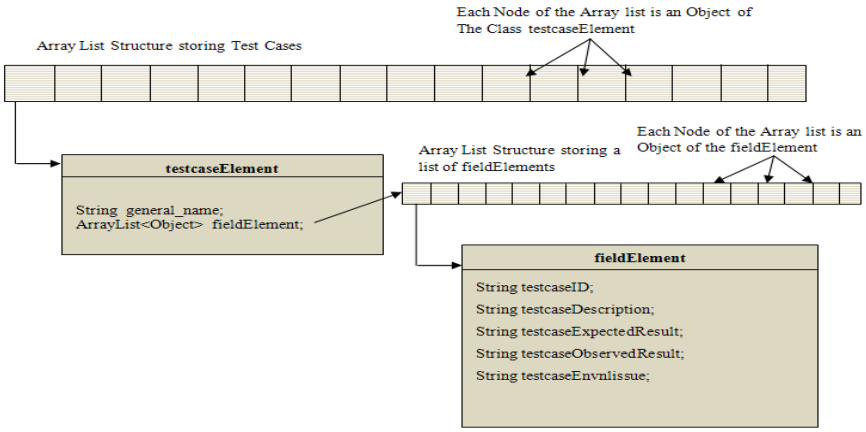


Figure 6. Local Data Structure for storing Testcase of Application .

**E.Application Deployer:**

Application Deployer is a user friendly interface. It guides user to deploy the application in selected web server environment. All the server setup is previously completed. Here our application code file will be posted in the default location of selected environment and then the application willl be run with home page. The following sets will be consider in application deployer:

- Select the Server Environment.
- Select the location where the application file will be exists. All the files of the application will be listed in the application file list box.
- Select the files which will be deployed and bring into deploy file list box.
- Press the deployment button.

## III. CONCLUSION

We have developed an ER-model and web technology based software application product generating tool for educational purposes. This tool incorporates object oriented technolgoy, XML. The verification process guarantee the semantic correctness for ER diagrams. The automatic translation from ER- diagrams to relational data schemas is practically useful. The automatic generation of template give relief from the time consuming form design task. The automatic generation of code give relief from the complex and time consuming task. Using this tools developer generate a software product in five minutes using different technology. The automatic testplan generator helps the tester to test this product according to all testcase. It helps the developer to design the testplan and guarantee that all test cases will be considerd. The automatic deployment helps the tester to test the product is run correctly in server environment in short time.

**Cite this article as:** Souradeep Sarkar, Debasish Hati, Prasun Kumar Mitra. "Software Application Generator: An ER Model-based Software Product Building Tool." *International Conference on Information Engineering, Management and Security (2015)*: 189-194. Print.

## V. OUR FUTURE WORK

We are extending ER modeler to support Complex type of ER diagram. We are currently extending Application Composer to support other complex type of templates such as tree template. We are exteding Application Generator to support more web technology language and different database server environment. We are extending Testplan Generator to support black box testing and white box testing test plan. We are extending Application Deployer to support the application in Cloud computing environment.

## REFERENCES

[1] Florescu, D. & Kossmann, D. (1999), Storing and Querying XML Data Using an RDBMS, IEEE Data Engineering Bulletin (22), 1999, 27-34

[2] Fernandez, F., Tan, C. & Suciu, D. (2000), SilkRoute, Trading between Relations and XML, Proc. 9th International World Wide Web Conference, Netherlands, 2000, 723-745

[3] Kleiner, C. & Lipeck, U. (2001), Automatic Generation of XML DTDs from Conceptual Database Schemas, Proc.Workshop of the Annual Conference of the German and Austrian Computer Societies, Austria, 2001, 396-405

[4] Shanmugasundaram, J.,Tufte, K., He, G., Zhang, C., DeWitt, D. & Naughton, J. (1999), Relational Databases for Querying XML Documents: Limitations and Opportunities, Proc. International Conference on Very Large Data Bases (VLDB), Scotland, 1999, 302-314

[5] Lee, D., Mani, M. & Chu, W. (2003), Schema Conversion Methods Between XML and Relational Models, Knowledge Transformation for the Semantic Web, IOS Press, Netherlands, 2003, 1-17

[6] A Practical Approach for Automated Test Case Generation. Published in: Computer Software Applications Conference, 2006. COMPSAC '06. 30th Annual International, Page(s): 183 – 188. IEEE Computer Society.

[7] Tallman, Owen H. Project Gabriel: Automated Software Deployment in a Large Commercial Network. Digital Technical Journal Vol. 7 No. 2, 1995, pp56-70.

[8] Paul E. Ammann, Paul E. Black, and WilliamMajurski. Using Model Checking to Generate Tests from Specifications. In Proceedings of the Second IEEE International Conference on Formal Engineering Methods (ICFEM'98), pages 46–54.IEEE Computer Society, 1998

[9] Kim B. Bruce. Foundations of Object-Oriented Languages: Types and Semantics. MIT Press, 2002.

[10] Luciano Baresi and Mauro Pezz`e. An introduction to software testing. Electr. Notes Theor. Comput. Sci.,148(1): 89–111, 2006.

**Cite this article as:** Souradeep Sarkar, Debasish Hati, Prasun Kumar Mitra. "Software Application Generator: An ER Model-based Software Product Building Tool." *International Conference on Information Engineering, Management and Security (2015)*: 189-194. Print.