Formal Modeling and Verification of an Information Retrieval System using SMV*

Jong-Hwan Kim, Hea-Sook Park, Doo-Kwon Baik
Dept. of Computer Science & Engineering, Korea University
1, 5-ka, Anam-dong, Sungbuk-gu, Seoul, Korea
Phone : +82-2-925-3706
{angel96, phs, baik}@software.korea.ac.kr

Abstract

An Information Retrieval System offers the integrated view of SCM(Supply Chain Management) information to the enterprise by making it possible to exchange data between regionally distributed heterogeneous computers and also to enable these computers to access various types of databases. The Information Retrieval System is modeled using Data Registry Model based on X3.285. We only verify the MetaData Registry Manager(MDR Manager) among the core parts using SMV(Symbolic Model Verifier) in order to verify whether our model satisfies the requirements under the given assumotions.

1 Introduction

Most of the manufacturing systems are, in nature, regionally distributed in the enterprise environment. And these systems cooperate through the interactions of application programs of distributed computers. Therefore the software, which integrates and monitors distributed application programs to cooperate systematically and efficiently, plays a very important role.

Developing the software for these manufacturing systems takes a lot of time and budget. And to be applicable for the largely extended environments, it should support data exchange between the distributed computers, and access various types cf databases. To support the decision making for SCM(Supply Chain Management) of the enterprise, it can offer the integrated information to the enterprise.

In this paper, we supply the integrated informaton to the enterprise by proposing an Information Retrieval System based on Data Registry Model[1] using X3.285[2]. The Information Retrieval System is modeled using the metadata registry and the schema registry.

And formal methods[3] are used to verify whether the model of the Information Retrieval System satisfies the requirements under the given assumptions. Formal methods, based on mathematics and logics, are used to specify or verify the hardware or software system, and these formal methods are largely made up of formal specifications and formal verifications. Formal

* This study is supported by the Specific Basic Studies of the Korea Science and Engineering Foundation in 2000.

specifications describe the assumptions of the environment in which the system operates, the requirements that the system should satisfy, the system designs that satisfy the requirements, and so on using the proving methods including the formal logic or mathematical logic. In this paper, to verify the model of the Information Retrieval System, we use the SMV among the formal verification tools.

This paper consists of the followings. In section 2, we explain the architecture of an Information Retrieval System based on MDR. In section 3, we explain the modeling of the Information Retrieval System and verify the model using SMV. Finally, in section 4, we draw the conclusion.

2 IRS(Information Retrieval System) based on MDR

To support the decision making of the enterprise, the IRS consisting of four parts, has the architecture as figure 1.

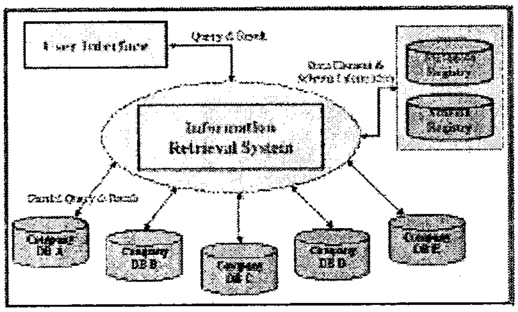


Fig. 1. IRS Architecture

① User Interface : it receives the user queries.

② Information Retrieval System : it offers the information which users want, by analyzing, transforming, processing and integrating queries after joining user query functions spread over the distributed databases in enterprise environment.

③ Metadata Registry and Schema Registry : they store the information of metadata registry and schema registry.

④ The distributed low databases : they store the real data.

The enterprise databases have various types of tables and data elements. Without regard to respective schema information of databases, users can query in order to acquire the information they want. In figure 2, IRS stores the information processed by MDR Manager(MetaData Registry Manager) using metadata in MDR to integrate the real databases including the varying table structures. MDR Manager offers the functions of display, modification, deletion, and insertion of information in MDR by integrating the semantically identical schemas.

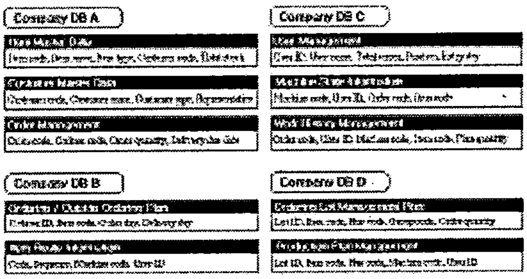


Fig. 2. Distributed Databases

3 IRS Modeling & Verification

The IRS's Modeling and Verification Process is depicted in figure 3 The IRS is modeled based on the IRS Architecture. Because the MDR Manager and the Schema Manager are the core parts of the system, we have only specified the MDR Manager to the statechart using STATEMATE, and then transformed it to SMV model using MOCES, and finally verified it using SMV among the SMV model checkers.

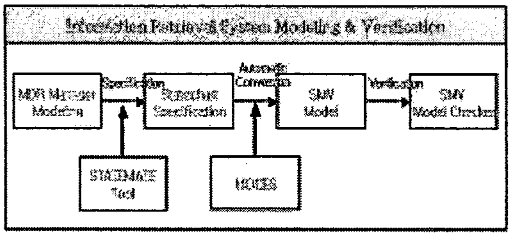


Fig. 3. IRS's Modeling & Verification Proesss

3.1 IRS Modeling

Based on the IRS Architecture, The IRS is modeled using the metadata registry and schema registry as figure 4. The users transform the query into the IRS by the user interface. And then IRS processes the query and returns the results.

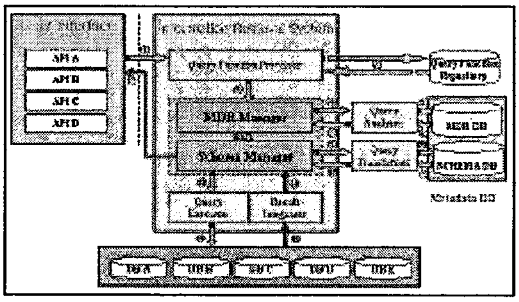


Fig. 4. IRS Model

The IRS has the functions as follows.

① The user interface : it receives the user queries.
② The query function processor : it calls query functions and transfers the invoked functions to the MDR Manger.
③ The MDR Manager : it offers the functions of display, modification, deletion, and insertion of information in MDR by integrating semantically identical schemas.
④ The Schema Manager : it manages the schema information of distributed databases.
⑤ The query analyzer : it analyzes the query functions using MDR.
⑥ The query transformer : it transforms a query function into another query statement to access the distributed databases by referencing the schema registry.
⑦ The query executor : it executes transformed queries and stores the results of queries.
⑧ The result integrator : it estimates the results of queries, and integrates results.

The execution procedure of the IRS is as follows.

① The User interface receives the user queries, and then sends them to the query function processor.
② The query function processor calls the query functions using the query function repository, and then sends invoked functions to the MDR Manager.
③ The MDR Manager offers the functions of display, modification, deletion, and insertion of information in MDR through the query analyzer by integrating semantically identical schemas, and sends the data element information to the Schema Manager.
④ The Schema Manager transforms the queries into the schema information which is applicable for the distributed databases using the query transformer by referencing the schema registry. And it gets the results using the query executor by perfomming partial queries in the respective database. And it estimates these results and integrates them using the result integrator. Finally, it shows the results to the users in the form that the users want.

The MDR Manager of the IRS is based on the Data Registry Model using the data registry. The Data Registry is a repository that stores the features of data needed to describe, retrieve, analyze, and classify the data. And with these features it can support interoperability, reusability, and standardization as its core functions[4]. The Data Registry Model integrates the Data Element Concept Entity by integrating Object

Class Entity and Property Entity, and then implements Data Element Entity, which is the core of DR by integrating Value Domain Entity and Data Element Concept Entity.

3.2 Statechart Specification

When the display buttons are clicked, if MDR Manager doesn't make the connection to MDR, it immediately displays the error message, "Connect Failure or SQL Error!!!". So we assume that the connection to MDR is assured. The MDR Manager's statechart designed by STATEMATE is shown as figure 5. Because Statechart[5] specifies system behaviors in the figure form called state machine, beginners who aren't used to formal methods can understand it, and statechart has the merits that it can definitely and visibly simulate the system behaviors. But it doesn't offer the function of the formal verification to prove the system properties. Therefore, we formally verify the statechart specification using SMV.

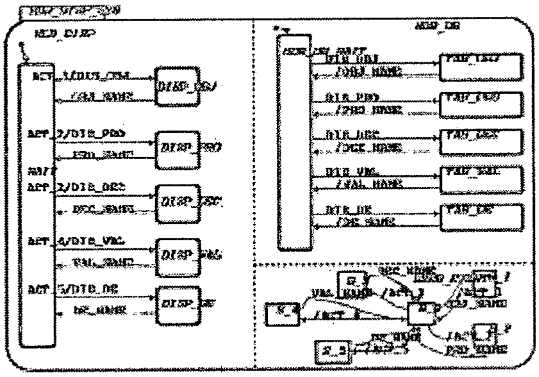


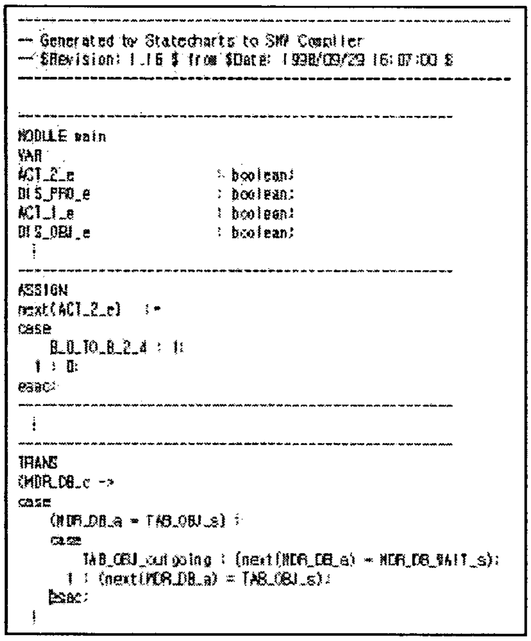
Fig. 5. Statechart Specification of the MDR Manager

3.3 Verification using SMV

There are many formal verification tools like VERSA[6], SPIN[7] and SMV[8]. In this paper, we verified using SMV. SMV system is a formal verification tool which can verify whether the finite state system satisfies the requirements specification represented in CTL[9]. The SMV's input language is created to specify the finite state system, and the SMV system can be easily specified as a synchronous mealy machine or an asynchronous network through the SMV's input language. SMV uses the symbolic model checking algorithm based on OBDD(Ordered Binary Decision Diagram)[10] in order to efficiently insepect whether the model represented SMV's input language satisfies the requirements specification represented in CTL.

To prove whether the model converted to SMV does the right operations, it is necessary to represent the properties to be satisfied using CTL. CTL can represent the paths that can occur according to time flow in tree forms. The computational tree is derived from the state transition graph. All paths appeared in the tree represent all the possible computations of the modeled system. Because CTL has the operator to describe the branching structure as a tree, it is classified as branching time logic. The statement represented in CTL consists of the atomic proposition, the boolean connectives of proposition logic, and the temporal operator. Every temporal operator consists of two parts. Those are the path quantifier and the temporal modality. Path quantifier A represents "about all path"; and E represents "about some path". Temporal modality F represents "sometimes"; X represents "in next state"; G represents "always"; and U represents "until".

By using SMV, when MDR Manager operates the display button, we can verify the correctness of the MDR Manager. To do this, the statechart specification should be transformed to SMV model using MOCES. The converted SMV model is as figure 6.



```
----------------------------------------------------------------
-- Generated by Statecharts to SMV Compiler
-- $Revision: 1.16 $ from $Date: 1998/09/29 16:07:00 $
----------------------------------------------------------------

------------------------------------------------------
MODULE main
VAR
ACT_2_e                    : boolean;
DIS_PRO_e                  : boolean;
ACT_1_e                    : boolean;
DIS_OBJ_e                  : boolean;
  ⋮
------------------------------------------------------
ASSIGN
next(ACT_2_e)   :=
case
    B_0_TO_B_2_4 : 1;
  1 : 0;
esac;
------------------------------------------------------
  ⋮
------------------------------------------------------
TRANS
(MDR_DB_c ->
case
    (MDR_DB_a = TAB_OBJ_s) :
    case
        TAB_OBJ_outgoing : (next(MDR_DB_a) = MDR_DB_WAIT_s);
      1 : (next(MDR_DB_a) = TAB_OBJ_s);
    esac;
  ⋮
```

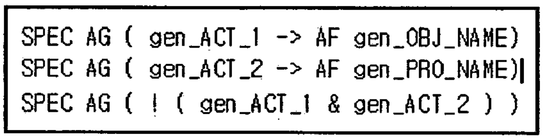
Fig. 6. SMV Model of the MDR Manager

To verify the operation of the SMV model represented as above, CTL specification as figure 7 has the meanings as follows.

① AG(gen_ACT_1 -> AF gen_OBJ_NAME) : When the button 1 is clicked, that is when there is a request to search the OBJ_NAME, it always should return the OBJ_NAME. Because the request and the response are represented as events in SMV model, we can prove whether the relevant events occurred.

② AG(gen_ACT_2 -> AF gen_PRO_NAME) : Like ①, when the button 2 is clicked, that is, when the request to search the PRO_NAME is occurred, it always should returnt the PRO_NAME.

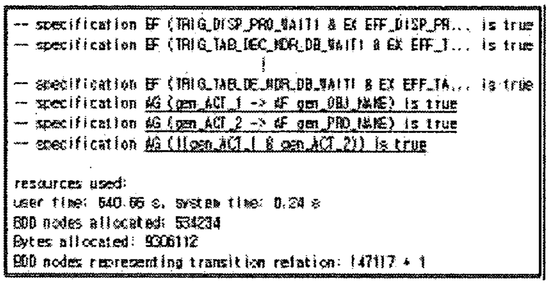
③ AG(!(gen_ACT_1 & gen_ACT_2)) : This expression is to specify mutually exclusive operations, that is, no more than 1 request should occur at the same time.



```
SPEC AG ( gen_ACT_1 -> AF gen_OBJ_NAME)
SPEC AG ( gen_ACT_2 -> AF gen_PRO_NAME)|
SPEC AG ( ! ( gen_ACT_1 & gen_ACT_2 ) )
```

Fig. 7. Requirements Specification of the MDR Manager represented CTL

The CTL expressions specified as above are added to the SMV specification. And using these expressions, the verified results are as figure 8.



```
-- specification EF (TRIG_DISP_PRO_WAIT1 & EX EFF_DISP_PR... is true
-- specification EF (TRIG_TAB_DEC_MDR_DB_WAIT1 & EX EFF_T... is true
                              ⋮
-- specification EF (TRIG_TAB_DE_MDR_DB_WAIT1 & EX EFF_TA... is true
-- specification AG (gen_ACT_1 -> AF gen_OBJ_NAME) is true
-- specification AG (gen_ACT_2 -> AF gen_PRO_NAME) is true
-- specification AG (!(gen_ACT_1 & gen_ACT_2)) is true

resources used:
user time: 540.66 s, system time: 0.24 s
BDD nodes allocated: 534234
Bytes allocated: 9306112
BDD nodes representing transition relation: 147117 + 1
```

Fig. 8. SMV Model's Verification Results of the MDR Manager

As above, we can know that every requirement is satisfied. This means that the MDR Manager of the IRS gets the data from MDR correctly. That is, this means that the correctness of the MDR Manager is verified.

4 Conclusion

The Information Retrieval System offers the integrated information to the enterprise by making it possible to exchange data between the regionally distributed heterogeneous computers and also to access various types of databases. The Information Retrieval System is modeled using Data Registry Model based on X3.285. We have verified the MDR Manager(MetaData Registry Manager) among the core parts in order to verify whether our model satisfies the requirements under the given assumptions. The verification results show that the MDR Manager satisfies every requirement. And this means that MDR Manager gets the data from MDR correctly.

The expected effects of the IRS's modeling and verification based on the metadata registry and the schema registry are as follows. First, data elements registered in the metadata registry can offer the integrated view about the total databases to users or application program developers. Application program developers don't need the thorough knowledge of databases schema structures, and can use only a small number of functions based on the conceptually established metadata registry for the application programs. Second, when schema structures of databases change over time in the distributed environment, the independence and flexibility of the total system can be maintained only by modifying the information registered in the schema registry. Third, it is possible to integrate and manage the enterprise information which exists regionally distributed and dependently.

In this study, we didn't verify and specify all components of the Information Retrieval System. Actually for formal verification of the Information Retrieval System, the components, such as Schema Manager, Query Analyzer, Query Transformer, Query Executer, Result Integrator and so on, should be specified and verified.

References

[1] Hea-Sook Park, Hong-Seok Na, Doo-Kwon Baik, "A Data Registry- based Environment for Sharing CALS/EC Metadata", Proceedings of The 27th KISS Spring Conference(B), VOL. 27-1, pp.60-62, 2000. 4.

[2] ANSI, "metamodel for the Management of Sharable Data", ANSI X3.285, 1999.2.

[3] Edmund M. Clarke and Jeannette M. Wing, "Formal Method : State of the Art and Future Directions", ACM Computing Surveys, pp.626-643.

[4] ISO, "Metadata Registry", ISO-11179, 2000.

[5] David Harel, "Statecharts : A Visual Formalism for Complex Systems", Science of Computer Programming 8, 1987.

[6] Duncan Clarke, "VERSA : Verification, Execution and Rewrite System for ACSR", University of Pensylvania, 1994.

[7] Gerald J. Holzmann, "The Model Checker SPIN", IEEE Transactions on Software Engineering, VOL. 23, NO. 5, pp.279-295, 1997. 5.

[8] Kenneth L. McMillan, "SYMBOLIC MODEL CHECKING", Kluwer Academic Publisher, 1993.

[9] Zohar Manna and Amir Pnueli, "The Temporal Logic of Reactive and Concurrent systems - Specification", Springer-Verlag, 1996.

[10] R. E. Bryant, "Graph-Based Algorithms for Boolean Function Manipulation", IEEE Transaction Computers, VOL. 35, NO. 6, pp.677-691. 1986. 8.