Providing Assistance for Database Schema Integration A Process Model Approach

Huei-Huang Chen
Department of Computer Science and Engineering
Tatung University
No. 40, Chung-Shan N. Rd, Sec. 3, Taipei, 104, Taiwan, R.O.C.
hhchen@cse.ttu.edu.tw

Ming-Wei Shr
Department of Computer Science and Engineering
Tatung University
No. 40, Chung-Shan N. Rd, Sec. 3, Taipei, 104, Taiwan, R.O.C.
helios@it01.cse.ttu.edu.tw

Abstract

Over the past decades, a considerable number of studies have been devoted to database schema integration. Although completely integrated schema has been an objective of study for a long time, only a few satisfying methodologies have been proposed, especially about schema comparison. The primary cause is that a concept can be modeled in many different ways by different people. In reverse engineering, it would cause several semantic confusions and many critical problems when we were integrating several schemas. Most of the resolutions of these issues in proposed methodologies are dependent on database administrators (DBAs) to provide some domain knowledge to aid the process of integration. That is to say that human plays an important role in integrating phase and saddles responsibility with the success or failure about integrated results. It would, as we said above, be ineffective to develop an automatic methodology for schema integration, but it is possible to find some methods to assist users in disco vering or checking correspondences, and reduce dependence on human. In this paper, we consider that taking process models into account is a practicabl way to provide the integration more semantics and decrease the reliance on the DBAs.

Keywords : database schema integration, revers engineering, process model, Extended Entity-Relationship model(EER),Data Flow Diagram(DFD).

I. Introduction

Over the past decades of information systems development, most enterprises may have several independent information systems as a result of departmental autonomy. For these enterprises, an increasing dependency and cooperation between different departments has created a need to access remote information systems as well as local ones. Thus, it becomes important to be able to interconnect existing, possibly heterogeneous, information systems. An essential part of this issue is database integration, i.e., the process of constructing a global schema from a collection of existin database schemas.

The survey paper of Batini [2] discusses and compares twelve methodologies for schema integration. It divides schema integration activities into five step : preintegration, schema comparison, schema conformation, schema merging, and restructuring. The most difficult part of the schema integration process is schema comparison. The fundamental problem is that a concept can be modeled in many different ways. Thus, schema comparison is an essential difficult problem concerning the semantic interpretation of models created by different developers. It is important to note that comparison of the schema obje cts is primarily guided by their semantics, not by their syntax [1]. It would, therefore, be ineffective to automate the processes of schema comparison completely. However, it should be possible to develop some methods to assists a user in discovering corr espondences between schemas and checking the validity of the proposed correspondences.

For reasons mentioned above, let us review the systems analysis processes in forward engineering. It applies two different views to model an information system. One uses data model to describe the relationships among data items ; the other applies process model to illustrate the relationships among data flows in a functional process. There is a lot of semantics distributed over these two models. In reserve engineering, takes only data models into account in capturing semantics about schema integration is incomplete. It also requires some additional domain knowledge provided by domain experts (such as DBA) to integrate schemas completely. For reducing dependence from this person and getting more semantics about integration, it may be worthwhile to take process models into account.

II. Related Works

A. Schema Translation

Database reengineering [7] starts with the schema, which defines the meaning of data and their relationships in the data models. Schema translations can be accomplished by mapping a logical schema into a conceptual EER model schema in reverse engineering. The translated conceptual schema must have all the logical schema's semantics. The detailed descriptions of translating logical schema in major data models into EER model can be found in [7,8,10].

B. Schema Integration

Database schema integration is the activity of merging the locally developed schemas for a given group of users and applications into an integrated global schema. Over the past decades, a considerable number of studies [1,2,3,4,5,6,9,11,12,15] have been devote to this area. We briefly describe the five steps of integration processes as following.

- Preintegration: involves translation of eac h heterogeneous local schema into a homogeneous one so that we can make further comparison among them and selection of integration processing strategies

such as binary or n-ary strategies.

- Schema Comparison: involves analyzing an comparing schemas in order to determin correspondences, different representations of th same concepts in particular. Methodologies broadl distinguish two types of conflicts
  - ✧ Naming conflicts: include homonyms an synonyms.
  - ✧ Structural conflicts: include type, dependency, key, and behavioral conflicts.

One notes that all conflicts depend on the recognition of the "same" concepts in different schemas. Mos methods only focus on the static structural an terminological information including their names their attributes, ranges of the attributes, and th relationships between object types contained in conceptual schema. This kind of knowledge belongs to syntactic information, not functional semantic about objects. Traditional methodologies provide functional semantics to resolve ambiguity are full depending on DBA. We present a brief technical review of two methodologies about schema comparison below.

◆ **Attribute Equivalence**

This approach [9] focuses on the problem of determining equivalencies on attributes an exploiting these in schema integration. Attribut equivalence is classified into three subtypes based on the basic equivalence properties. **STRONG** attribute equivalence indicates that updates are possible against the integrated schema. It is classified into two categories **STRONG** α equivalence (holds for some point in time) and **STRONG** β equivalence (holds for every point in time). **WEAK** attribute equivalence is applied when only retrieval will be permitted against the integrated schema. **DISJOINT** attribute equivalence is applied when neithe **STRON** nor **WEAK** equivalence holds but attributes may still be combined. This holds when the roles of the attributes are the same, yet there exists n mapping function on the basic attribute equivalence properties. So attribute a and b are disjoint equivalence, we denote a α **DOMAIN-DISJOINT-ROLE-EQUAL** b.

Object and relationship set equivalencies are defined in terms of the above equivalencies o attributes. Two object classes are compared in th same manner as the identifier attributes of those objects. And accordingly, to determine th equivalence of a relationship set is to determine th equivalencies of the identifier attributes of th participating object classes. Here, we have to define somewhat formal description of object equivalence. Let object class A have un que identifier k_1 and attributes $a_1, a_2, \ldots, a_m$. Let object class B have unique identifier k_2 and attributes $b_1, b_2, \ldots, b_n$. Let RWS(A) and RWS(B) (called real-world state) refe to the sets of real-world instances of object class A and B at a given moment i n time. There are five cases as shown below.

Case 1 : If k_1 STRONG β EQUAL k_2, then RWS(A) EQUAL RWS(B).

Case 2 : If k_1 STRONG β CONTAINS k_2, then RWS(A) CONTAINS RWS(B).

Case 3 : If k_1 STRONG β CONTAINED IN k_2, then RWS(A) CONTAINED-IN RWS(B).

Case 4 : If k_1 α DOMAIN-DISJOINT-ROLE-EQUAL k_2, then RWS(A) DISJOINT RWS(B).

Case 5 : If k_1 STRONG β OVERLAP k_2, then RWS(A) OVERLAP RWS(B).

◆ **System-Guided Integration**

In this approach [15], the main stress falls on searching all possible suspect similaritie automatically, and then confirming these predicates by integrators according to their domain knowledge. The fundamental idea is to organize the integration steps into the two interleaving and iterative phases of schema comparison and schema conforming, and succeeding phase of schema merging. There are several guidelines derived from syntax of schema, such as two attributes may be similar because they have identical names or their attribute domains are identical. Then the integrators can distinguish two classes of factual predicates (*equal*-predicate, *different*-predicate) for each concept.

- Schema Conforming: involves modifying one or both of the schemas to be integrated until each phenomenon in the UoD is represented in the same way in both schemas.
- Merging and Restructuring: involves superimposin the schemas in order to obtain one integrated schem and maintaining the integrated schema on completeness, correctness, minimality, an understandability.

III. Applying Semantics within Process Model

A. Characteristics of Process Model

Viewing an information system as a collection of processes, we are interested in analyzing processes whose objective is to provide services to either internal processes or external customers. In this paper, we adopt data flow diagram (DFD) for our process model. Some features about DFD: (1) presents explicitly the essential functions in the systems, and relationships among them; (2) understands the data flows among processes; and (3) is tightly linked t data models. Here, we need an integrated proce ss model t show the relationships between each object in a global view. We can get each local process model through source code analysis or existing development documents. And then we have to integrate each local process model into a global one We apply three kinds of operations in process model [14], such as merging processes which own the same functions, coupling processes which have a casual relationship, and regrouping processes which are related. In this paper, we assume that the integrated process m odel had been derived, and a constraint about data stores in process model is that each data store has to map into an entity or a relationship in EER model.

B. Semantic Patterns within Process Model

There are five relationships between two entity types, E_1 and E_2 [11]; and similar relationships exist among relationship types :

- Identical.
- Superclass/Subclass.
- Generalization with Overlap.
- Generalization with Disjoint.
- No Relationships.

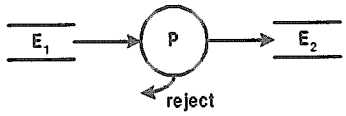
Besides these five concepts, there are two

higher-level concepts to represent more abstract semantics, categorization and aggregation. Based on past studies, we can determine the former three relationships easily without controversy, but we have to apply the domain knowledg from DBA to check the two cases of disjoint gene ralization, and no relationship. About categorization and aggregation, almost no existing methodologies take them into account. We hope we can get more relationships, as we said above, about integration through process models. We appl attribute equivalence to conform the suspect relationships between two objects derived from process model.

Generalization--Superclass/Subclass Relationship

A subclass is an entity type that has a distinct role and is also a member of a superclass.

Pattern 1

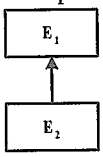


Here, we expect that there is a superclass/subclass relationship between E_1 and E_2. When the following conditions hold, we can get an integration primitive, subset (E_2, E_1).

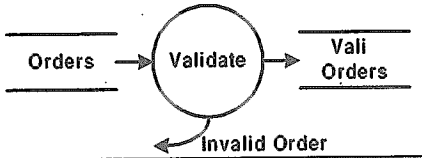
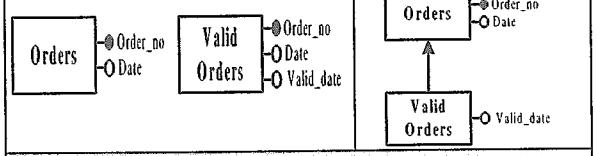
Conditions :
RWS $(E_1) \cap$ RWS $(E_2) \neq \varnothing$ and RWS $(E_1) \supseteq$ RWS (E_2).

Algorithm :
For each process *For each entity E_1 is an input of P* *For each entity E_2 is an output of* *IF RWS $(E_1) \cap$ RWS $(E_2) \neq \varnothing$ and RWS $(E_1) \supseteq$ RWS (E_2)* *THEN Subset (E_2, E_1) is hold* *ENDIF* *IF subset (E_2, E_1) is hold THEN* *Remove E_2 attributes which are E_1 attributes $\cap$ E_2 attributes* *ENDIF*

Finally, we map this semantic pattern into EER data model.



For example, we consider the situation below.

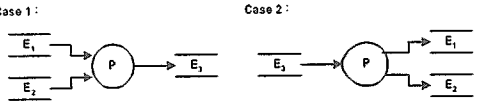
Process Model

Original Schema / New Schema

Instance Set

Orders.Order_no	Valid Orders.Order_no
A101	A101
A102	A104
A103	A105
A104	
A105	
Analysis Results	
RWS (Orders) $\cap$ RWS (Valid Orders) $\neq \varnothing$	
RWS (Orders) $\supseteq$ RWS (Valid Orders)	

Generalization—Overlap/Disjoint Relationship

Overlap/Disjoint relationships mainly describe the relationship between those subclasses. The disjoint constraint requires the subclasses to be mutually exclusive. Subclasses that are not disjoint are overlap.

Pattern 2

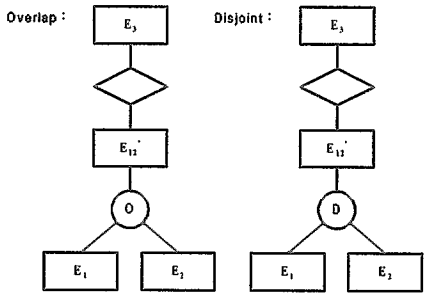


Here, we expect that there is an overlap or disjoint relationship between E_1 and E_2. When the following conditions hold, we can get an int egration primitive, overlap(E_1, E_2) or disjoint(E_1, E_2). In these two situations, we have to create a new entity involved the common attributes within those subclasses.

Conditions :
Case 1 : RWS $(E_1) \cap$ RWS $(E_2) \neq \varnothing$ Then E_1 overlap E_2.
Case 2 : RWS $(E_1) \cap$ RWS $(E_2) = \varnothing$ Then E_1 disjoint E_2.

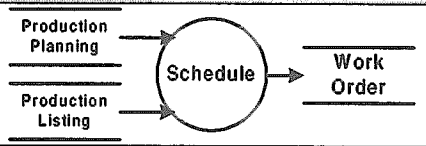
Algorithm :
For each process *For each entity E_1 and E_2 are inputs of* *IF RWS $(E_1) \cap$ RWS $(E_2) \neq \varnothing$ THEN* *Overlap (E_1, E_2) is hold* *ELSE* *Disjoint (E_1, E_2) is hold* *ENDIF* *For each entity E_1 and E_2 are outputs of* *IF RWS $(E_1) \cap$ RWS $(E_2) \neq \varnothing$ THEN* *Overlap (E_1, E_2) is hold* *ELSE* *Disjoint (E_1, E_2) is hold* *ENDIF* *IF Overlap (E_1, E_2) or Disjoint (E_1, E_2) is hold THEN* *Generate a super entity E_{12} for E_1 and E_2* *Attributes of E_{12} are E_1 attributes $\cap$ E_2 attributes* *Create a new relationship between E_{12} and E_3* *ENDIF*

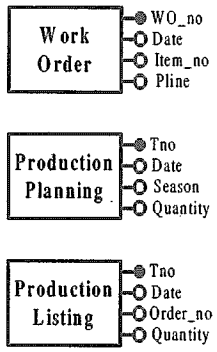
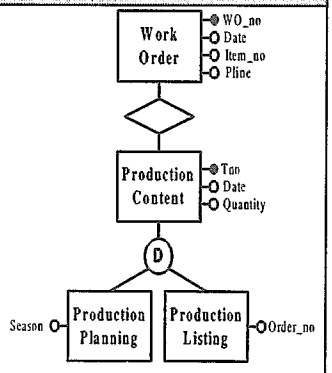
Finally, we map these semantic patterns into EER data models.



For example, we consider the situation below.

Process Model



Original Schema	New Schema
	

Instance Set

Production Planning.Tno	Production Listing.Tno
F2001	S1234
F2003	S2346
F2004	S2564
F2010	S3390
	S5641
	S7363

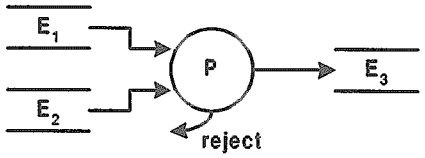
Analysis Results

RWS (Production Planning) ∩
RWS (Production Listing) =∅

Categorization

A category has two or more superclasses that ma represent distinct entity types, and it is a subset of the union of its superclasses.

Pattern 3



Here, we expect that there is a categorization relationship between E_1, E_2, and E_3. When the following conditions hold, we can get an integration primitive, category (E_1, E_2, E_3) .

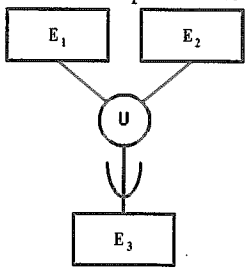
Conditions :

RWS (E_1) ∩ RWS (E_3) ≠∅ and
RWS (E_2) ∩ RWS (E_3) ≠∅ and
(RWS (E_1) ∪ RWS (E_2)) ⊇ RWS (E_3) .

Algorithm :

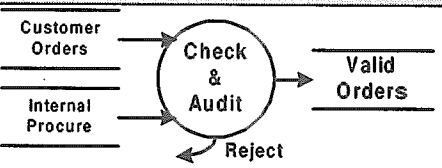
For each process
For each entity E_1 and E_2 are inputs of
For each entity E_3 is an output of
IF RWS (E_1) ∩ RWS (E_3) ≠∅ and
RWS (E_2) ∩ RWS (E_3) ≠∅ and
(RWS (E_1) ∪ RWS (E_2)) ⊇ RWS (E_3)
THEN Category (E_1, E_2, E_3) is hold
ENDIF

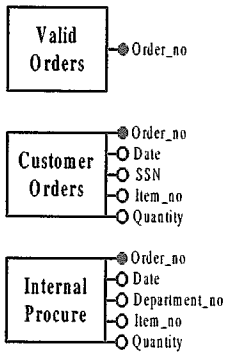
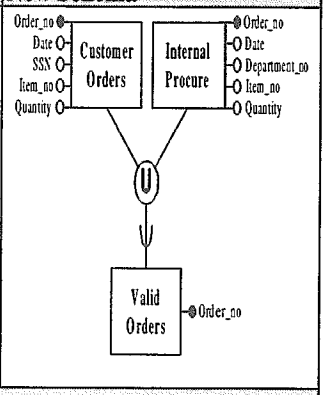
Finally, we map this semantic pattern into EER data model.



For example, we consider the situation below.

Process Model



Original Schema	New Schema
	

Instance Set

Valid Orders .Order_no	Customer Orders .Order_no	Internal Procure .Order_no
C1020	C1011	I3472
C1043	C1020	I3587
I3472	C1043	I4790
I7733	C1067	I7733

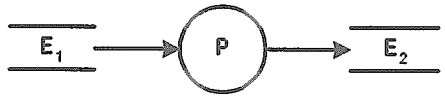
Analysis Results

RWS (Customer Orders) ∩ RWS (Valid Orders) ≠∅
RWS (Internal Procure) ∩ RWS (Valid Orders) ≠∅
(RWS (Customer Orders) ∪ RWS (Internal Procure))
⊇ RWS (Valid Orders)

Aggregation/Association

Aggregation is an abstraction concept for buildin composite objects from their comp onent objects. It is sometimes useful when the higher-level aggregate object is itself to be related to another object.

Pattern 4



Here, we can consider that there is an association relationship between E_1 and E_2 when E_1 is a relationship relation and forming an aggregation object with two participant entities in EER model. In this situations, we can get an integration primitive, associate (E_1, E_2).

Conditions :

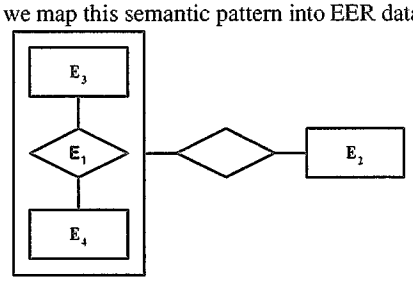
Data store E_1 is a relationship relation

Algorithm :

For each process
  For each E_1 is an input of P
    For each E_2 is an output of
      IF E_1 is a relationship relation THEN
        Associate (E_1, E_2) and E_1 is an aggregation object
      ENDIF

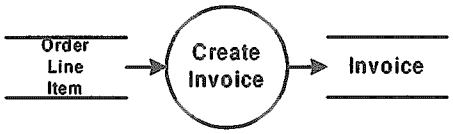
IF associate (E_1, E_2) is hold THEN
  Create a relationship between aggregation object E_1 and E_2
ENDIF

Finally, we map this semantic pattern into EER data model.

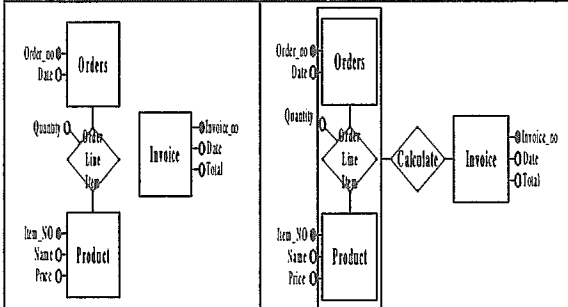


For example, we consider the situation below.

Process Model



Original Schema | **New Schema**



Analysis Results

Aggregate (Orders ∪ Order Line Item ∪ Product)
Associate (Order Line Item, Invoice)

C. Combining Process Model with Schema Integration

Now, we want to combine these semantics derived from process models with existing schema integration methodologies. In Fig. 1, we show what role the process model plays in the w hole architecture. The process model facilitates schema comparison by making different kinds of "hidden" information explicit concerning the semantics of the model objects. The starting point is that two conceptual schemas, Schema 1 and Schema 2, describing two systems are compared to each other and the problem is to establish correspondences between concepts between the two schemas. Then, we introduce process model to which all or a subset of objects in both schemas are connected. Thus, a semantic relation between the compared models is established and can be used in determinin correspondences among the concepts. The primar advantage of applying process model is that we can get functional semantics about each object within the system. It would be more understandable to say that we will know what roles the objects play in the whole system. The "role" is functional-oriented semantics which tells us the correct positions the objects stand. Through process model, we can avoid the wrong integrating schemas and can get more semantics.

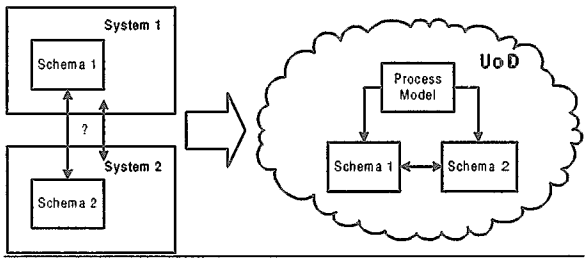


Figure 1 Process Model within Schema Comparison

Now, we illustrate a complete methodology for integrating schemas with assistance of process model. We can represent the total processes in a simple diagram as : Fig. 2. There are six steps in our schema integration methodology.

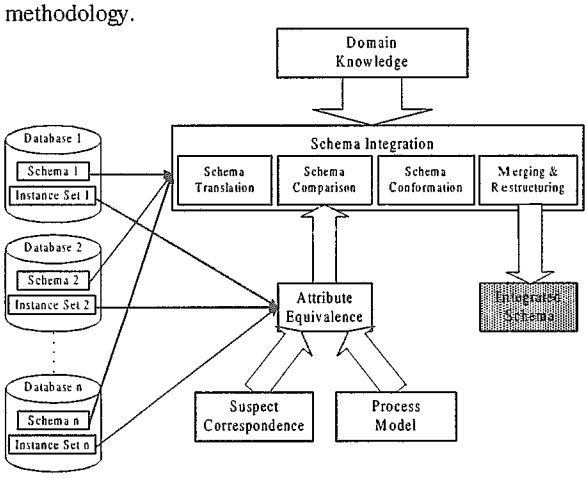


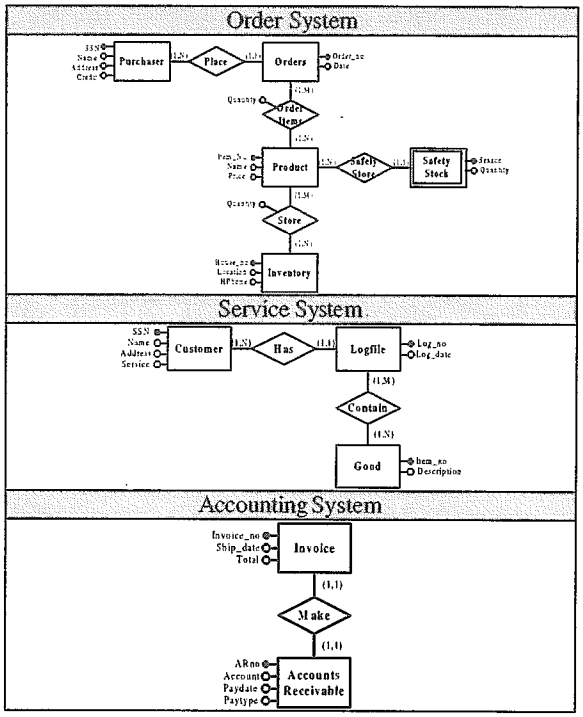
Figure 2 Total Processes of Schema Integration by Using Process Model

Step 1: First, we must translate heterogeneous database schemas into homogeneous ones such as EER model we use.
Step 2: We apply system -guided approach to find all possible suspect correspondences.
Step 3: We use process model to find further suspect correspondences aiming at those semantic patterns.
Step 4: After Step 2 and Step 3, w e can collect some suspect correspondences. Now, we depend on attribute equivalence to confirm these correspondences.
Step 5: In this step, we resolve all conflicts to make the compatible for integration.
Step 6: According to those confirmed predicates, w e merge the schemas by means of a superimposition of common concepts.

IV. Case Study

We provide an example to illustrate the integratin processes and the feasibility of this method. This example includes three database schemas, order system, service system, and accounting system.

Step 1: First step, we have to translate each existing database schema into an EER model. In our paper, we give a premise that we already got these three EER models.



Step 2: we compare the similarities and differences between two database schemas to find the integration primitives. These primitives can aid us to integrate these database schemas especially positive clauses. We list total suspect correspondences we found below.

$ass\_similar_{att}$ ($Purchaser.Name_{Order}$, $Customer.Name_{Service}$).
$ass\_similar_{att}$ ($Purchaser.Address_{Order}$, $Customer.Address_{Service}$).
$ass\_similar_{att}$ ($Purchaser.SSN_{Order}$, $Customer.SSN_{Service}$).
$ass\_similar_{att}$ ($Product.Item\_No_{Order}$, $Good.Item\_No_{Service}$).
$ass\_similar_{att}$ ($Product.Name_{Order}$, $Customer.Name_{Service}$).

$ass\_similar_{obj}$ ($Purchaser_{Order}$, $Customer_{Service}$).
$ass\_similar_{obj}$ ($Product_{Order}$, $Good_{Service}$).
$ass\_similar_{obj}$ ($Product_{Order}$, $Customer_{Service}$).

Step 3: Here, we present a complete data flow diagram in Fig. 3. Then we discover the semantic patterns to get more semantics about integrating database schemas. We list the suspect predicates below.

$ass\_generalization$ ($Purchaser_{Order}$, $Customer_{Service}$).
$ass\_generalization$ ($Product_{Order}$, $Customer_{Service}$).
$ass\_generalization$ ($Order_{Order}$, $Order\ Items_{Order}$).
$ass\_generalization$ ($Order_{Order}$, $Inventory_{Order}$).
$ass\_generalization$ ($Inventory_{Order}$, $Safety\ Stock_{Order}$).
$ass\_categorizati\ o$ ($Customer_{Service}$, $Product_{Order}$, $Logfile_{Service}$).
$ass\_associate$ ($Order\ Items_{Order}$, $Invoice_{Accoun}$).
$ass\_associate$ ($Invoice_{Account}$, $Accounts\ Receivable_{Account}$).

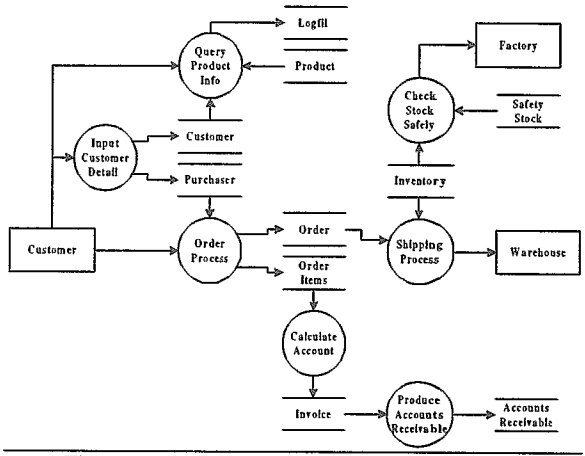


Figure 3 Data Flow Diagram of Whole System

Step 4: We check these predicates by using attribute equivalence technique for aiding us to get conformation predicates. For example, $ass\_similar_{obj}$ ($Purchaser_{Order}$, $Customer_{Service}$).

	Purchaser.SSN	Customer.SSN
1	F123491370	F124789910
2	B221023348	A223313345
3	A124587612	F123491370
4	S123456789	J228912471
5	C228773625	K123321311
6		B221023348

Through analysis, we find the values in Purchaser.SSN and Customer.SSN are overlapping with each other, $RWS(Purchaser) \cap RWS(Customer) \neq \varnothing$, so we get a positive predicate, $similar_{obj}$ ($Purchaser_{Order}$, $Customer_{Service}$). Then, according to the another predicate, $ass\_generalization(Purchaser_{Order}, Customer_{Service})$, we ca get a more positive predicate, $overlap_{obj}$ ($Purchase_{Order}$, $Customer_{Service}$).

Step 5: Through step 2 to step 4, we get four positive predicates.

$overlap_{obj}$ ($Purchase_{Order}$, $Customer_{Service}$).
$equal_{obj}$ ($Product_{Order}$, $Good_{Service}$).
$aggregate$ ($Orders_{Order} \cup Order\ Line\ Item_{Order} \cup Product_{Order}$).
$associate$ ($Order\ Items_{Order}$, $Invoice_{Accoun}$).

Here, we discover a naming conflict in $equal_{obj}$ ($Product_{Order}$, $Good_{Service}$), so we have to renam "Good " to "Product". The new predicate is $equal_{obj}$ ($Product_{Order}$, $Product_{Service}$). Then we can merge schemas according t these predicates, and we show the integrated schema in Fig. 4.

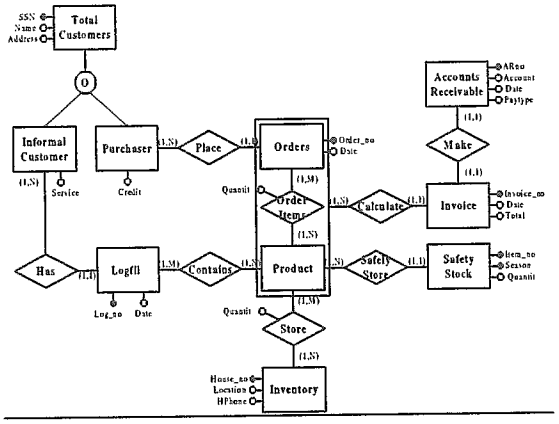


Figure 4 The Integrated Schema of Overall System

V. Implementation

In this paper, we propose a prototype system to verify our hypothesis. This prototype system is based on a PC platform and runs in Microsoft Windows NT 4.0. We use Borland Delphi as our development tools to program our system. We implement a prototype system focusing on process model semantic patterns analysis and attribute equivalence to verify the practicability of this approach. We can represent the relationships between our implementation and external objects diagrammatically as follows :

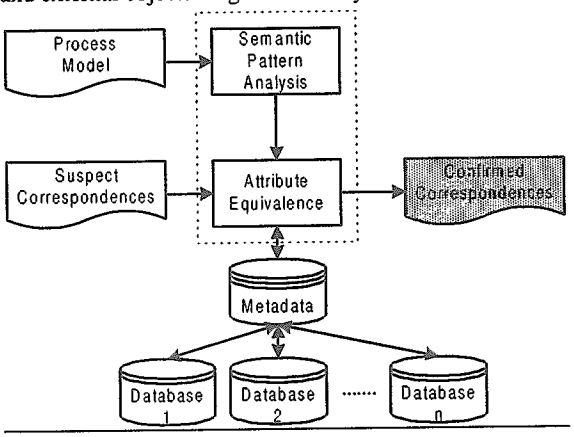


Figure 5 The Architecture about System

The circled with a dotted line is the part of our implementation. The suspect correspondences include two parts : attribute equivalence and object equivalence. Whe we got suspect correspondences, we have to confirm the separately. Fig. 6 shows suspect attribute correspondences analysis window, and Fig. 7 shows suspect entity correspondences analysis window in our prototype system. Fig. 8 shows the window of process model semantic pattern analysis in our prototype system.

Figure 6

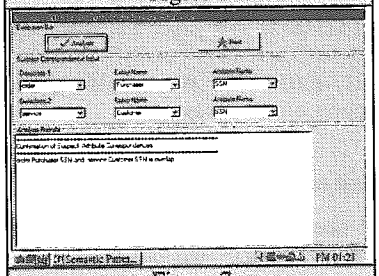


Figure 7

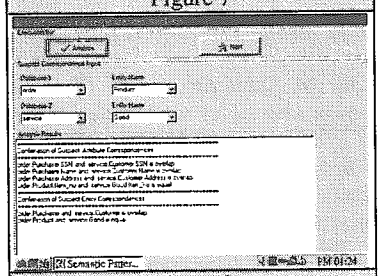
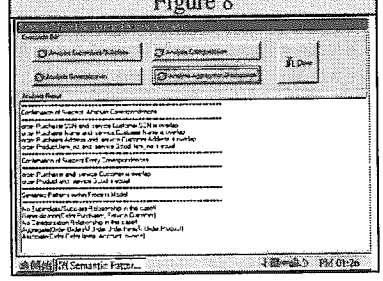


Figure 8



VI. Summary

In this paper, we propose a process model approach to aid integrators in integrating database schemas. Through finding semantic patte rns, we can get several semantics mainly describing the relationship between objects that are in the same functional process. In traditional integration methodologies, this kind of semantics are provided b DBAs or domain experts. Moreover, we also propose the architecture for schema integration with the assistance of process model.

The urgent future work for this paper is to implement a real application for our process model approach. It will be a very useful semi-automatic tool to help people findin more semantics. Another piece of future work is included object-oriented techniques into this research area, such as UML (Unified Modeling Language) or OO (Object-Oriented Analysis) . These new techniques own richer expressions to represent a concept. In other words, it can be hold more semantics in their expressions. It is a worthy direction of future research in schema integration.

Another direction of research is knowledge extraction from another sources for aiding schema integration. It is also a har d task because of all knowledge is memorize in the domain experts ' brain, or distributed over several tools. For example, providing a global vie for helping our integrating schemas is a worthwhile method to study.

Finally, we hope that this result of this research can provide a foundation for further researches in the area of schema integration.

VII. Acknowledgement

The work presented in this paper has been supported by National Science Council, Taiwan, R.O.C., under Grant No. NSC 88-2213-E-036-002. We de eply appreciate their financial support and encouragement.

VIII. Reference

[1] Amit P. Sheth, and James A. Larson, "Federated database systems for managing distributed, heterogeneous, and autonomous databases, " ACM Computing Surveys, Vol. 22, No. 3, pp.183 -236, Sep. 1990.

[2] C. Batini, M. Lenzerini, and S. B. Navathe, "A comparative analysis of methodologies for database schema integration," ACM Computing Surveys, Vol. 18, No. 4, pp. 323-364, Dec. 1986.

[3] Carlo Batini and Maurizio Lenzerini, "A methodology for data schem a integration in the entity relationship model," IEEE Transactions on Software Engineering, Vol. SE-10, No. 6, pp.650-664, Nov. 1984.

[4] Christine Parent and Stefano Spaccapietra, "Issues and approaches of database integration, " Communication of the ACM, Vol. 41, No. 5es, pp.166-177, May. 1998.

[5] Carlo Batini, Stefano Ceri, and Shamkant B. Navathe, *Conceptual Database Design. An Entity-Relationship Approach.* Benjamin/Cummings. 1992.

[6] Isabelle Mirbel, "Semantic integration of conceptual schemas," Data & Knowledge Engineering, Vol. 21, pp.183-195, 1997.

[7] J. S. P. Fong and S. M. Huang, Information Systems Reengineering, Springer-Verlag, 1997.

[8] J. S. P. Fong, "Methodology for Schema Translation from Hierarchical or Network into Relational," Information and Software Technology, Vol. 34, No. 3, pp.159-174, 1992.

[9] James A. Larson, Shamkant B. Navathe, and Ramez Elmasri, "A theory of attribute equivalence in databases with application to schema integration, " IEEE Transactions on Software Engineering, Vol. 15, No. 4, pp.449-463, Apr. 1989.

[10] Navathe, S. and Awong, A., "Abstracting relational and hierarchical data with a semantic data model," Entit -Relationship Approach, pp.305-333, 1988.

[11] Navathe, S., Elmasri, R., and Larson, J., "Integrating user views in database design," IEEE Computing, Vol. 19, No. 1, pp.50-62, Jan. 1986.

[12] Stefano Spaccapietra, Christine Parent, and Yann Dupont, "Model independent assertions for integration of heterogeneous schemas, " VLDB Journal, Vol. 1, pp.81-126, 1992.

[13] S. Castano, V. De Antonellis, M. G. Fugini, and B. Pernici, "Conceptual schema analysis: Techniques and applications," ACM transactions on Database Systems, Vol. 23, No. 3, pp.286-333, Sep. 1998.

[14] Silvana Castano, and Valeria De Antonellis, "A framework for expressing semantic relationships between multiple informatio n systems for cooperation," Information Systems, Vol. 23, No. 3/4, pp.253-277, 1998.

[15] Willi Gotthard, Peter C. Lockemann, and Andrea Neufeld, "System-guided view integration for object-oriented databases," IEEE Transactions on Knowledge and Data Engineering, Vol. 4, No. 1, pp.1-22, Feb. 1992.