

Implementation of data-exchanging system based on Message Oriented Middleware in Agricultural Website

Zhang Xiaoshuan^{1,2}, Wu Qinghua¹, Tian Dong¹, Zhao Ming¹,

¹College of Information and Electrical Engineering

China Agricultural University, Beijing, P. R China, 100083

²Jiangsu Provincial Key Laboratory of Computer Information Processing Technology, Suzhou

Corresponding author: zhaoming@cau.edu.cn

Abstract: - With the startup of the Golden Agriculture Project, the step of being-information of agriculture is becoming rapid. And the transformation and share of data is indispensable to the being-information of agriculture. How to implement data combination, data transformation and data receiving applications are the important means to complete the information share safely and enhance the efficiency.

The paper starts with searching of methods to implement data interchange, and introduce some of the methods, points of the techniques, etc. Basing on this, the paper also introduces the detail requirement analyses, system design and detail implementation of the system.

According to the requirement and trait of the project, a data interchange system is researched and completed. And a data interchange model based on message-oriented middleware (MOM) is presented in this paper, which builds a middleware between the province and the ministry taking part in data interchange. The system has traits as follows: 1. keeping the data safe and credible while it is transformed. 2. having excellent transplantable and applied capability. 3. doesn't need intervention of workman in the process of data interchange. 4. applying the data interchange between databases of different structure. 5. being simple to be developed and applied.

MOM TongLink/Q offers interfaces for application development, and it completes the data transformation through the internet. The integration adapters developed do the data management, which are developed based on the Frame for Applications Integration TongIntegrator. This method offers a new approach to resolve the question of data interchange. Now the system has been successfully applied in the data interchange project of Ministry of Agriculture.

Key-Words: - Message Oriented Middleware, Data Exchange, long-range Database, Website, Data Exchange

1 Introduction

With the startup of the Golden Agriculture Project, the step of being-information of agriculture is becoming rapid. And the transformation and share of data is indispensable to the being-information of agriculture. How to implement data combination, data transformation and data receiving applications are the important means to complete the information share safely and enhance the efficiency.

The paper starts with searching of methods to implement data interchange, and introduce some of the methods, points of the techniques, etc. Basing on this, the paper also introduces the detail requirement analyses, system design and detail implementation of the system.

According to the requirement and trait of the project, a data interchange system is researched and completed. And a data interchange model based on message-oriented middleware (MOM) is presented in

this paper, which builds a middleware between the province and the ministry taking part in data interchange. The system has traits as follows: 1. keeping the data safe and credible while it is transformed. 2. having excellent transplantable and applied capability. 3. doesn't need intervention of workman in the process of data interchange. 4. applying the data interchange between databases of different structure. 5. being simple to be developed and applied.

MOM TongLink/Q offers interfaces for application development, and it completes the data transformation through the internet. The integration adapters developed do the data management, which are developed based on the Frame for Applications Integration TongIntegrator. This method offers a new approach to resolve the question of data interchange. Now the system has been successfully applied in the data interchange project of Ministry of Agriculture.

Message-oriented middleware (MOM) is a client/server infrastructure that increases the interoperability, portability, and flexibility of an application by allowing the application to be distributed over multiple heterogeneous platforms. It reduces the complexity of developing applications that span multiple operating systems and network protocols by insulating the application developer from the details of the various operating system and network interfaces. APIs that extend across diverse platforms and networks are typically provided by the MOM.

MOM is software that resides in both portions of client/server architecture and typically supports asynchronous calls between the client and server applications. Message queues provide temporary storage when the destination program is busy or not connected. MOM reduces the involvement of application developers with the complexity of the master-slave nature of the client/server mechanism.

MOM comprises a category of inter-application communication software that generally relies on asynchronous message-passing, as opposed to a request/response metaphor.

Most message-oriented middleware depend on a message queue system, but there are some implementations that rely on broadcast or multicast messaging systems.

There are many solutions available to implement MOM-based system. TongLink/Q (TLQ for briefly below) is one kind of MOM products developed by TongTech Company. It ensures the reliable message transmitting between different computers and different application software. The message can be transmitted from an application to another through Internet or on local machine. In addition, it supports different Operation System platforms and network environment. Figure.1 illustrates the system of TLQ message transmission.

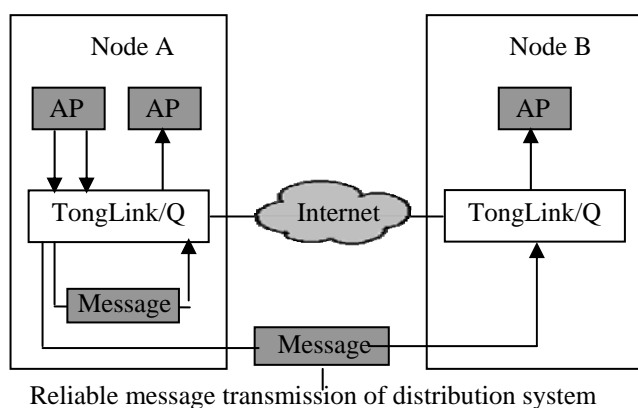


Figure 1: TLQ message transmission.

So the technology of MOM TongLink/Q was adopted as the basic framework to resolve the above problem. This paper is organized as follows. In section 2 gives brief introduction about the MOM TongLink/Q, application Integration Framework TongIntegrator, Section 3 emphasizes on the design and implementation of the system and A sub-system of *China Agricultural website* has selected as the case study.

The paper briefly introduces the MOM TongLink/Q, application Integration Framework TongIntegrator and its second development interfaces simply, and emphasize introduce the design and implementation of the system

2 The brief introduction of the system development technology

2.1 Message Oriented Middleware (MOM) and TongLink/Q

Message-oriented middleware (MOM) is a client/server infrastructure that increases the interoperability, portability, and flexibility of an application by allowing the application to be distributed over multiple heterogeneous platforms.

MOM can be seen as a natural extension of the packet paradigm of communications prevalent in the lower layers of the OSI network model. Unlike RPC and object-orientation it is an asynchronous form of communication, i.e. the sender does not block waiting for the recipient to participate in the exchange. If the message service offers persistence and reliability then the receiver need not be up and running when the message is sent. Unlike OOM, in MOM, messages are generally untyped and the internal structure of messages is the responsibility of the application.

In traditional MOM, messages are addressed to their recipients, although sender and receiver are loosely coupled and need not synchronise to communicate. This may be less suitable for wide-area, large-scale systems and so it may be advantageous to decouple sources and sinks with respect to naming, so they may be mutually anonymous to each other. A typical way of designing this is (so called) publish-subscribe systems, where sources "publish" to the entire network and interested sinks "subscribe" to messages. The network then only forwards them downstream if there is at least on subscriber on that path. This requires the message transport service to understand the message internals, although some systems are "topic-based"

where each message has a subject line which the transport system reads and can ignore the rest of the message.

MOM has a larger share of the market than Object-Oriented Middleware, being used for database access in large business applications.

It reduces the complexity of developing applications that span multiple operating systems and network protocols by insulating the application developer from the details of the various operating system and network interfaces. APIs that extend across diverse platforms and networks are typically provided by the MOM.

TongLINK/Q is a messaging middleware based on message queuing or message passing. Its main function is to provide reliable message transmission between application programs.

TongLINK/Q can transmit these messages between different networks, computer systems and application software.

TongLINK/Q provides a distributed application developing and deploying platform which is simple, convenient for use, highly efficient, and reliable. By TongLINK/Q, users can develop reliable and highly efficient distributed applications in a simple way.

TongLINK/Q provides a distributed application management platform, which monitors and supervises over the distributed applications with name service and application.

TongLink/Q, as one kinds of products developed by TongTech Company, is a messaging middleware based on message queuing or message passing. Its main function is to provide a distributed application developing and deploying platform which is simple, convenient for use, highly efficient, and reliable. TongLINK/Q automatically calls corresponding applications to process requests. At the same time, according to the number of request packet, TongLINK/Q automatically adjusts the number of service processes (or threads) to make sure of efficiency of request queue processing. By TongLINK/Q, users can develop reliable and highly efficient distributed applications in a simple way.

2.2 Application Integration Framework

Application Integration Framework based on message transmission and processing is responsible for implementing to collect and process data from one system, then provide the processed data to another system. It contains some pre-built component framework, which can be used to integrate application system fast by using configuration files. TI has the following features:

(1) Support multi-data formats and function of data conversion;

(2) Provide data process function, included data compression, encryption, data format translation, message filter and so on;

(3) Provide standard development Framework for system and application integration adapter, and convenient configuration tools;

(4) Implementation with JAVA code so can be deployed on many different platforms.

When developing custom components on the Integration Framework, the system development engineering can follow to the document to implement interfaces of related components, and add needed model for the special function, then put the compiled class files to the given directory. For example, the interface offers a base class for simple-process component Simple Process, then the developer implements different methods according to Pipe component or Sink component.

2.3 TI Adapter Link Mode

The configuration file of TI can be shown with a chain of components, which can be written on the graphics interface by linking needed components. A chain includes a Source component and a Sink component at least, besides it can include Pipe component, Simple-Process component and so on. Inside IT the data format for data transmission is DataObject (DO), which is described by attribute name and value. Figure.2 shows TI Adapter Link Mode.

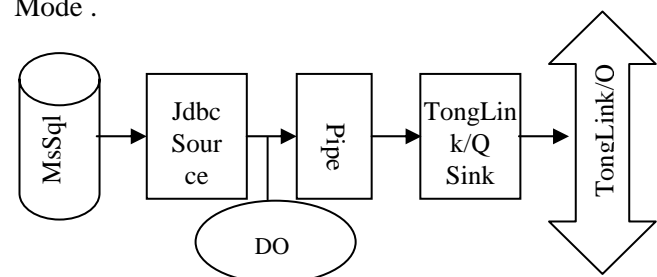


Figure 2: TI Adapter Link Mode.

This Adapter reads data from MSSQL database, processes the data through Pipe component, then transmits the data to TLQ send queue.

2.4 The case subsystem

After our country joins WTO, the further opening to the outside world of market for farm products, domestic, the international market influences each other and strengthens constantly, the market competition is becoming fiercer. How help domestic enterprise open up domestic and international markets, expand agricultural product sell, have already become

a new urgent task. According to the regulation of WTO agricultural agreement, the government of member state can offer agricultural products marketing and promote the service to national enterprises and peasants in the general service plan, including the market information, consultation and promoting related to particular agricultural product. For this reason, since 2002, on the basis of investigations, draw lessons from other methods in WTO member state, supported by financial department, the Ministry of Agriculture begins to explore that sets up agricultural products marketing to promote and serve the platform and service mechanism. Through the efforts of more than two years, have already set up and regarded promoting in world as the focal point tentatively at present, network show until guide, supplemented by the public service advertisement, world until agricultural product marketing combined together to promote promote service system at home, have already launched and made certain actual effect in an all-round way in every promotion and service activity.

Agricultural Exhibition Online System, a subsystem of *China Agricultural website*, has selected as the case study. The data exchanging between central government and province relates to three database tables: Users, Enterprise and Product Tables.

3 The system development and test

3.1 Overall Framework

The development of the system which is related to database tables such as user tables, business forms and products table is based on online exhibition database.

Data processing of the tables is completed by TI adapters. The TI adapters compose software platform of the system and handle directly data processing in tables.

It's necessary to make both province end and ministry end correspond because of the importance of the user IDs and the differences between user IDs in the province end base and user IDs in the ministry end base, when you're registering.

So the idea that new registered IDs of the province end is returned by the user IDs of the ministry end is adopted. Steps are as follows:

- 1.A JavaBean based on TLQ is packaged in the province end which sends requests to ministry end through TLQ and receives queues to get the returned IDs.

- 2.Add demon to the ministry end. Demon can receive queues from TLQ to get request information

from which demon can get a new ID from user tables and put it into sending queues which will return to the province end TLQ.

When there are new users registering on the province end, the province user IDs are returned by JavaBean and demon from the ministry user tables. When operations such as deletion in user tables, insertion, update, deletion in enterprise tables and product tables occur, they can reach the adaptor end in real-time.

3.2 Developing Data exchanging mode based on MOM

Through analysis on the system, an data exchange model can be concluded. And the transmission of the data is shown in picture 5-2. The principal of it is: when the province end is operating the database (activated trigger inserts changed information to the middle tables), TI adapter will take control of the middle tables. When the TI adapter finds there new records, it will read information of the middle tables and get data according to the information. The figured data will be sent to the ministry end through TLQ.

A JavaBean based on TLQ is packaged in the province end. It can send request information to the ministry end through TLQ and receive returned information from the receive queues in TLQ. When the ministry end add demon, it can read request from receive queues in TLQ and compare the information in the corresponding database tables. Then after some operations it will return the information and put it on the TLQ sending queue to send it back to the province end.

While the application at the Province End begin to run the database, the database the triggers are activated to insert needed data into middle tables, then TI adaptor running background reads the table and take out data, which is transmitted to the Ministry End through TLQ.

The application at the Province End needs to encapsulate JavaBean based on TLQ, which is used to send request message to Ministry End through TLQ and receive response message from receive queue of TLQ. On the End of Ministry a daemon is added, which is used to read request message from receive queue of TLQ, and query in related database tables to get a flag of duplicate data according the message, and then return a message to send queue of TLQ to the Province End.

When users are entering new information to the province end system (Province Exhibition online system), the province end will send the request (name of the enterprise) to the receive queues of the ministry

end. The demon will figure out the information (the overall user ID) and put it into sending queue of the TLQ. The province end applies the returned information from the receive queue. After some treatment, new data will be inserted into the province end database. The operation will activate the three tables' triggers, and insert information into middle tables. The province adapter will get data from database according to the recorded information of the middle tables. When the data is figured out by TI adapter, it will be send to the ministry end TLQ receive queues through TLQ in the form of messages. The ministry end TI adapter will get the information from receive queue and figure it out again. At last, the data will be written into the ministry end database. Then the data exchange process is completed. Figure 3 illustrates the process of data transmission.

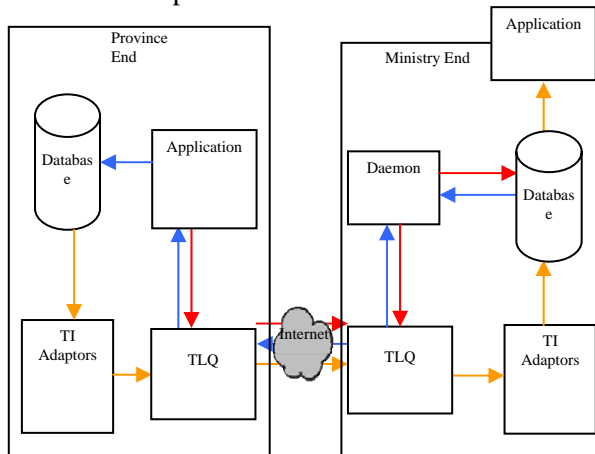


Figure 3: Data Exchanging Mode.

When users fill new information on the application at Province End (*Agricultural Exhibition Online System website at the provincial bureau*), the Province End sends a request message (Enterprise name) to the receive queue of TLQ on Ministry End, then after processed corresponding data (Global UserID) by the daemon the request is put on the send queue of TLQ. Application at Province End gets the returned message and inserts the worked data into tables. The triggers of the three tables are activated and insert data into the middle tables.

The TI adaptors on Province End take out data from database according to the information of the middle tables and after processed the data is sent from send queue to receive queue of TLQ in the form of message. The TI adaptors get and process the message, finally write the data to the database on Ministry End. The process of data synchronization is completed.

3.3 Implementing the databases synchronization

While the application at the Province End begin to run the database, the triggers are activated to insert needed data into middle tables, then TI adaptor running background reads the table and take out data, which is transmitted to the Ministry End through TLQ.

The below table 1 includes the key Fields, showing the process of databases synchronization

Table1 the key fields of databases synchronization

| Source Table: | Sink Table: | Comments |
|--------------------|-------------|--|
| i_product | PROD_CPXX | |
| USER_ID | USER_ID | New Field of source table, its value gotten from sink table |
| PRODSEQ | PRODSEQ | New Field of sink table, matching source table |
| i_prod_name | CP_MC | Product name doesn't need processing, correspond directly |
| i_prod_sort | CP_LB | Serial number of product sort is different, need FilterPipe component |
| i_harmless_flag | CP_YZLX | There Fields of source table correspond to one Field of sink table, need type conversion component |
| i_harmless_pic_url | CP_WGHTP | Related picture files to products, need files read&write components. |

There needs some deployment of database on Province End:

Step 1: create middle tables using to record information of Insert, Update and Delete operation, which is inserted by Triggers of these tables.

Step 2: create Triggers of tables, which are activated while insert, update or delete a record of a table and save these changes in middle tables, including id, status and operation type (insert, update, delete) of the record etc.

3.4 Developing TI Adaptor

While the application of the Province End works on the database, the triggers are activated to insert needed data into middle tables, then TI adaptor runs background and is responsible for reading the table and take out data, which is transmitted to the Ministry End through TLQ.

3.4.1 Sub-subsection

Some customized components are required to executive the special functions:

Customized component 1: When TI Adaptors on Province End take out data from database an import question is the processing of files, such as picture files, video files. So File Processing component is needed inside a TI Adaptor, writing the files on disk to DO and then publishing on TLQ.

Customized component 2: Accordingly File Processing component inside TI Adaptor on Ministry End is needed to read DO and write the date to files on disk.

Customized component 3: A type conversion component is needed to resolve the question that Multi-Fields correspond to one Field.

3.4.2 Component Link

Take the synchronization of product tables for example.

(1) Province End

TI Adaptor on Province End reads data from SQL Server through reading Attribute Values of the source component C1. The data is processed in Customized component C2 and published on send queue of TLQ. JdbcSqlSource source component was adopted given that MS SQL Server is the database system(Fig 4).

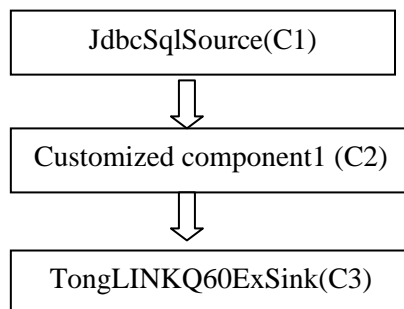


Figure 4: The transaction flow at province end

(2)Ministry End

TI Adaptor embedded in the system at Ministry End receives data from receive queue of TLQ through C1 and writes the data to DO. The component C2 judges the type of operations on database: If the operation is Delete, then the TI adaptor links to SQLSink component, executes delete operation according to Attribute values of DO, end; If the operation is Insert and Update, then the TI adaptor links to Customized component 2 to process files, Customized component 3 to converse Field types, C3 component to filter some

different Field values, C4 component to judge to different SQLSink component to do different operation according to Insert and Update(Fig 5).

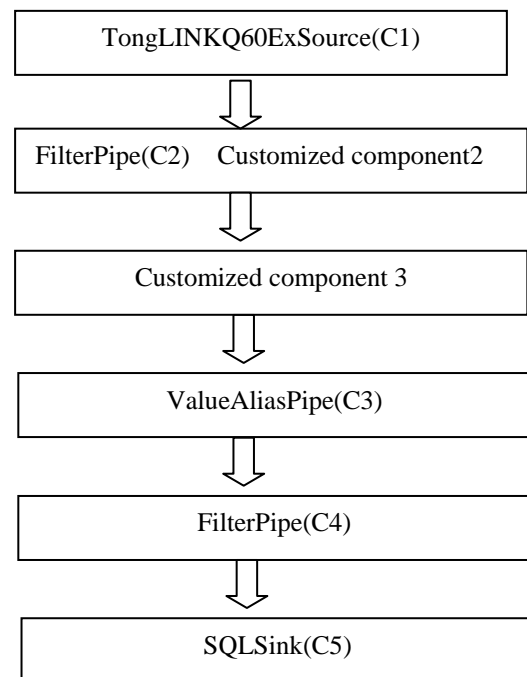


Figure 5: The transaction flow at Ministry end

3.4 the system test

3.4.1 Security and reliability of data transmission I Adaptor

The transmission of data on the internet is actually on the virtual network built by MOM TLQ, so the system has the same security and reliability features with TLQ for data or message transmission.

3.4.2 Excellent availability and applicability

There need only to modify some Attribute Values of TI Adaptors and no extra works to change the link mode of components when deployed on Agricultural Exhibition Online System of different Province Ends.

Otherwise, the JaveBean based on TLQ is also the same and no need to supply different JavaBean for different Province Ends. The virtual network established by TLQ is the transport layer, what we need to do is just to configure a node name for it.

3.4.3 No human intervention in the process of databases synchronization

TI Adaptors and TLQ can run automatically after startup, and in the whole process of databases synchronization human intervention is unnecessary.

3.4.4 Simply for development

TI offers a set of ready-made standard components. The system develop engineering can link the components to Adapters through configuration files following the establishment methods of different kinds of link modes.

3.4.5 Implement data exchanging of different databases system

TI offers many ready-made standard source components and sink components, such as JDBC, which supplies read and write operations on RDBMS, including Sybase, Oracle, MS SQL Server etc. So in order to implement data exchanging of different databases system developers just use different JDBC source components and sink components.

4 Database Analysis

Create three middle tables in the province database: BCG Msg table, BCG MsgService table, BCG MsgType table. These three tables are used to record operations of the database, so that TI adapters can get the required information according to the insertion record.

When Source component which is to get information in database spot a issue (in the BCG Msg table) , it will perform an reference which returns results collections to form a sql sentence. Then it'll turn the results collection into DO. The returned DO type will be named after set message type and its attribute name will be named after each field's capitalization.

The detailed steps are as follows:

1. insert into BCG MsgType table one record. Put message type and the process of storage into the table. Here the process of storage is the process in which the TI will operate to get data source. You can get the required record collection by going through this process.

2. insert into BCG MsgService table one record. Insert into each linked component of BCG MsgService table one corresponding record.

3. insert into the main table one record. It must include BCG MsgType table, master keys of BCG MsgService table and the storage process parameters of the operated BCG MsgType table. These can be 10 parameters at most. When inserting new records into BCG Msg ,the TI system will spot the new record, and operate the storage process which is stored in the BCG MsgType table according to the recorded information to figure out results of the record collection of the storage process.

Main BCG Msg table is used to monitor Source database regularly to check information. Structure of the table is as follows:

```
BCG_Msg
(
  MsgServiceId int,
  MsgTypeId int,
  MsgQueuedAt datetime,
  MsgStatus varchar(10),
  MsgDeliveredAt datetime,
  MsgFailedText varchar(255),
  OperationType varchar(10) NULL,
  MsgParam1 varchar(60) NULL,
  MsgParam2 varchar(60) NULL,
  MsgParam3 varchar(60) NULL,
  MsgParam4 varchar(60) NULL,
  MsgParam5 varchar(60) NULL,
  MsgParam6 varchar(60) NULL,
  MsgParam7 varchar(60) NULL,
  MsgParam8 varchar(60) NULL,
  MsgParam9 varchar(60) NULL,
  MsgParam10 varchar(60) NULL,
  MsgId numeric(32) Identity,
  MsgSpid int NULL
)
```

When we insert data into BCG Msg table according to the storage process, it transmits information service name, information type, operation name and 10 parameters. Information service name and type are those which you insert into MsgTypeName in BCG MsgService table and MsgServiceName in the BCG MsgType table. The operation name is highlighted in the user table(insert, update or delete) . 10 parameters are values of the operation storage process(at most 10 parameters, less than 10 or none is permitted). Then you can take these values as parameters to operate the storage process and insert data into the BCG Msg table.

Insert into BCG Msg table a "NEW" state information. As you insert into it, it will perform lots of investigation and form a correct number. The storage process will perform whenever there's a valuable piece of information. Also the performance is generally operated by the trigger.

BCG MsgService table listed all the service name which is monitored by the linked components from all the database. Each service name has an unique ID logo table. Structure of the table is as follows:

```
BCG_MsgService
(
  MsgServiceId int,
  MsgServiceName varchar(30),
```

```
MsgServiceDesc varchar(255)
)
```

BCG MsgType table defines information type ID logo, information type and the storage process name. The information type's name is "Name". Information is described as "description". Its storage name is MsgSprocName. Information type name is the corresponding created by Source DO type. Table structure is as follows:

```
BCG_MsgType
(
  MsgTypeId int,
  MsgTypeName varchar(30),
  MsgTypeDesc varchar(255),
  MsgSprocName varchar(92)
)
```

Secondly create a trigger in a database which is related to data exchange. The function is that when you do some operations such as insertion, update and deletion, the activated trigger can insert information into the middle table.

We need 7 triggers. They are triggers for operations such as insertion, update and deletion of "I applicant", "I enterprise" and "I product".

The insertion code for trigger of the i_product table :

```
CREATE TRIGGER i_productinsert ON i_product
AFTER INSERT AS
BEGIN
  DECLARE @product_id bigint , @userid
numeric(13) , @product_seq int
  DECLARE cursor_insert cursor for select
i_id , USER_ID from inserted
  open cursor_insert
  fetch NEXT FROM cursor_insert into
@product_id , @userid
  IF @@fetch_status=0
  BEGIN
    DECLARE @theseq int
    DECLARE @strid varchar(30)
    DECLARE cursor_select CURSOR for
select
ISNULL(max(PRODSEQ) , 1) from
i_product where USER_ID=@userid
    OPEN cursor_select
    FETCH NEXT FROM cursor_select
into @theseq
    SET @theseq=@theseq+1
```

```
PRINT
```

```
'theseq='+convert(varchar(20) , @theseq)
CLOSE cursor_select
DEALLOCATE cursor_select
update i_product SET
PRODSEQ=@theseq where i_id=@product_id
select @strid=convert(varchar(30)
, @product_id) from inserted
exec BCG_QueueMsg
@MsgTypeName = 'i_product' , @MsgServiceName
='i_product' , @OperationType ='insert' ,
@MsgParam1 = @strid
END
close cursor_insert
deallocate cursor_insert
END
```

The trigger code for deletion operation of the i_product table :

```
CREATE TRIGGER i_productdelete ON i_product
AFTER INSERT AS
BEGIN
  DECLARE @product_id bigint , @userid
numeric(13) , @product_seq int
  DECLARE cursor_insert cursor for select i_id ,
USER_ID from deleted
  open cursor_insert
  fetch NEXT FROM cursor_insert into @product_id ,
@userid
  IF @@fetch_status=0
  BEGIN
    DECLARE @theseq int
    DECLARE @strid varchar(30)
    DECLARE cursor_select CURSOR for select
ISNULL(max(PRODSEQ) , 1) from i_product where
USER_ID=@userid
    OPEN cursor_select
    FETCH NEXT FROM cursor_select into
@theseq
    SET @theseq=@theseq+1
    PRINT 'theseq='+convert(varchar(20) , @theseq)
    CLOSE cursor_select
    DEALLOCATE cursor_select
    update i_product SET PRODSEQ=@theseq where
i_id=@product_id
    select @strid=convert(varchar(30) , @product_id)
from inserted
```



```

exec    BCG_QueueMsg    @MsgTypeName    =
'i_product' ,    @MsgServiceName = 'i_product' ,
@OperationType = 'delete' ,    @MsgParam1 =
@strid
END
close cursor_insert
deallocate cursor_insert
END

```

5 Conclusion and prospects

At present, the system has been successful in the application of the data exchange projects in Ministry of Agriculture. This shows that the operation of the system is stable and the implementation is feasible. Now, the Data exchange has become a hot research and difficult problems. The data exchange model in this paper provides a new way of thinking for resolving this issue.

The main role of MOM TongLINK/Q's is to complete the various computer information transmissions. It provides a safe and reliable end-to-end real-time communication services, a data security guarantees for customers between the messaging and Easy-to-use, highly efficient and reliable system of distributed application development platform and application programming interface. Meanwhile, it support for multiple operating environment and the underlying network platform. This makes the development of simple and do not have to concern the data transmission network. You can only concentrate on the preparation of the upper logic.

The base on the middleware data exchange model, that this paper presents, greatly simplifies the system development. And it makes the system to have good safety and reuse. As the Department of the provincial-and the TI-adapter completed data-processing system in this critical part and to amend the property value of TI adapter is very simple, the systems in various provinces also promote the use of very simple. In addition, the system is operating in the background and need not manual-intervention after it run, so it greatly reducing the maintenance workload.

In the course of the study, on the following issues are further discussed:

1. As the system requirements for the restrictions, the implementation of the system of data exchange is only from the province –database to the department-database one-way data transmission. However, as the demand changes, Two-way data transmission is the future trend of development. So considering the data exchange continue to improve the model. In this model

for provincial-use database triggers changes in the way trigger. If doing two-way exchange, the Department will not be able to end the same way, otherwise it will cause data transmission cycle. We should considered replace the use of trigger functions in the end of the procedure.

2. A key step of implementation of the system is to be done at both province –database and department-database simultaneously. But due to the development of client application does not take into account the needs of the latter part of data sharing in the province –database, there are large differences in the province–database design and the department-database structure. That leading to the database in sync more complicated, we need to make the development of the TI components compared multifunctional and complex, So it increases the workload of system development. Data sharing is the trend of development of network technology, so, in the future, application of construction should be took fully into account whether the future demand for data sharing, in the database design of this into account.

6 Acknowledgements

This research is supported by the Hi-tech development plan (863 projects under Grand No. 2006AA10Z239), and open fund from Jiangsu Provincial Key Laboratory of Computer Information Processing Technology, Suzhou University. The authors are grateful for the anonymous reviewers who made constructive comments.

References:

- [1] *TongLinkQ6.0 programmer reference*. TongTech Company, 2004.3
- [2] Java2 : The Complete Reference (Fifth Edition) , Electronics Industry Publishing House, 2003
- [3] *TongIntegrator using book*, TongTech Company, 2004.9
- [4] Zhao Ming, Ma Yongliang, Zhang Xiaoshuan, Design and development of an automated information gathering and issue system based on a small and medium-sized website, *WSEAS Transactions on Computers*, Vol.6, No.12, 2007, pp.1174-1176.
- [5] Shim, Young-Chul; Kang, Ho-Seok; No, Sehun Source, Design and application of a middleware for context-aware ubiquitous computing, *WSEAS Transactions on Computers*, Vol.6, No.5, 2007, pp.785-792.
- [6] Doblander, Andreas ; Rinner, Bernhard; Trenkwalder, A middleware framework for

- dynamic reconfiguration and component composition in embedded smart cameras, *WSEAS Transactions on Computers*, Vol.5, No.3, 2006, pp.574-581.
- [7] German Shegalov, Michael Gillmann, Gerllard Weikum, XML-enabled workflow management for e-services across heterogeneous platforms, *VLDB Journal*, Vol.10, No.1, 2001, pp.91-103.
- [8] Tran P, Greenfield P, orton I, Behavior and Performance of Message -oriented Middleware Systems, *Distributed Computing Systems Workshops*, 2002, pp.645-650.