Database Management System

Exception handling

Content

- **Function**
- **Design**
- **Implementation**

Copyright Declaration

Function

Function: design the exception handling method in the program, and define the program unified exception handling class.

1. Program exception

There will be some exceptions more or less in the program running. Such as, operation files do not exist, computer disk destroyed, database server does not start, network cannot access, user's input are not standardized, and so on. These cases are not predicted and unavoidable.

2. Exception handling

To enhance software usability and program robustness, we need to add exception handling in the program.

When the program encounters a different type of exception, it will throw various exceptions, For example, operating MFC file class abnormally will case exception of MFC files class, so do C++ pointer and database components, and so on.

3. Unified handling to the exception

The underlying exception information usually cannot be understood by users. In order to make the procedures more user-friendly and the structure clearer, this iteration will be consolidated into a single exception class for all exceptions. The exception is thrown up one layer at a time, then unify handling in the view layer, and prompts the user by an comprehensible method.

Design

Iterative deployment based on the “Data structure design” function.

Exception handling is a kind of handling when errors or unrespectable conditions occurred during the program running, and be able to handle the exception to ensure the program normal running when exception occurred.

There are two types of exception handling mechanism in C++: C++ exception handling mechanism and the MFC exception handling mechanism.

This project uses the C++ exception handling mechanism, customize a exception class, and unify handle various exception in the program.

1. Design of exception handling

(1) C++ exception handling mechanism

Customize exception class CAppException, and use the try, catch, throw exception handling mechanism in the C++.

(2) MFC exception handling mechanism: CException+ macro

Customize exception class CAppException, as this class inherits CException, we process exception handling by macro TRY，CATCH，END_CATCH，THROW.

Note:

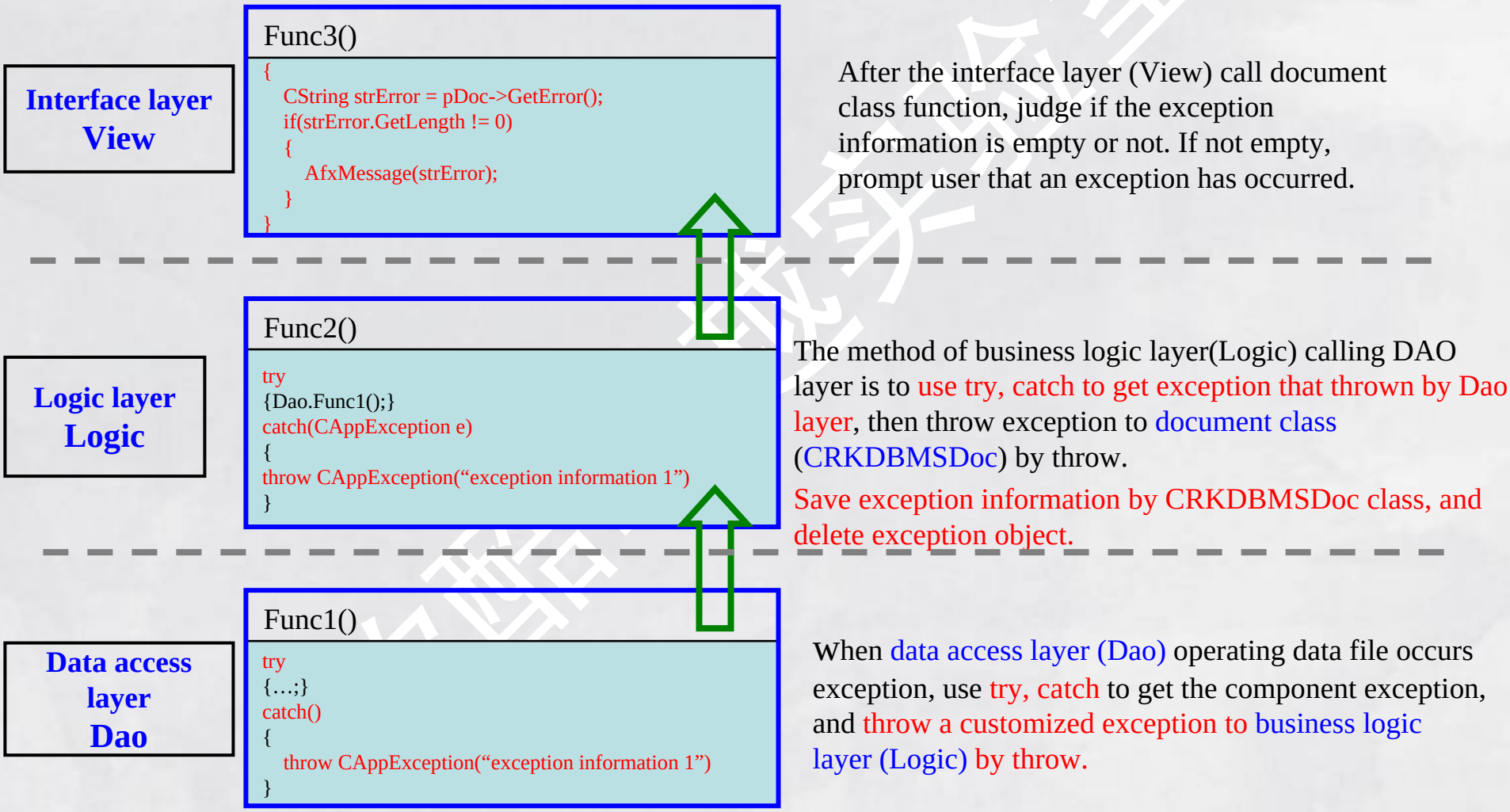
CException class is a virtual base class which cannot instantiated an object.

Need to create an exception object by “new” operator to be thrown for exception inherited from CException.

Design

2. Idea 1: use C++ exception handling mechanism

Customize exception class CAppException, and process exception handling by try, catch, throw.



After the interface layer (View) call document class function, judge if the exception information is empty or not. If not empty, prompt user that an exception has occurred.

The method of business logic layer(Logic) calling DAO layer is to use try, catch to get exception that thrown by Dao layer, then throw exception to document class (CRKDBMSDoc) by throw.

Save exception information by CRKDBMSDoc class, and delete exception object.

When data access layer (Dao) operating data file occurs exception, use try, catch to get the component exception, and throw a customized exception to business logic layer (Logic) by throw.

Design

3. Idea 2: Use exception handling method in MFC (CException + marco)

Customize exception class Cappexception to inherit CException class, and handle exception by TRY，CATCH，END_CATCH，THROW.

Interface layer
View

```
Func3()
{
   CString strError = pDoc->GetError();
   if(strError.GetLength != 0)
   {
      AfxMessage(strError)
   }
}
```

After the interface layer (View) call document class function, judge if the exception information is empty or not. If not empty, prompt user that an exception has occurred.

Logic layer
Service

```
Func2()
TRY
{   Dao.Func1();  }
CATCH(CAppException, e)
{
   THROW(new CAppException("exception information 1"));
}
END_CATCH
```

The CXXLogic::Func2() method of business logic layer(Logic) is to use try, catch, end_catch to get exception that thrown by Dao layer, then throw customized exception to document class (CRKDBMSDoc) by THROW.

Save exception information by CRKDBMSDoc class, and delete exception object.

Data access layer
Dao

```
Func1()
try
{ …;
}
catch(_com_error e)
{
   throw new CAppException("exception information 1")
}
```

When data access layer (Dao) operating data file occurs exception, use try, catch to get the component exception, and throw a customized exception to business logic layer (Service) by THROW.

4. Program frame design

Exception handling class is used in the Dao, Logic, and View layers. It is unified handle to the exceptions. Therefore, place the customized exception class CAppException in "Utill".

5. CAppException class design

Customizing an exception class, typically add a data member to the customized exception class to represent the exception information, and an exception number to identify exceptions.

(1) Data member

1) CString m_szError: exception information, access permission is private.

2) int m_nCode: exception number, access permission is private.

(2) Constructor CAppException(CString strError)

When an exception occurs, the exception information is saved to the exception class object. When customize exception class, initialize the exception class information in the constructor of customized exception.

(3) Member function

CString GetErrorMessage()

Get the value of a customized exception class members m_szError.

Implementation

Iterative development based on the “Data structure design” function, the steps are as follows:

Step 1: add CAppException class

Step 2: test CAppException class

www.ruanko.com

Thanks

Exception handling