Зертханалық жұмыс 11.

Жұмыс мақсаты:

Rational Rose 98 код генерациясын үйрену.

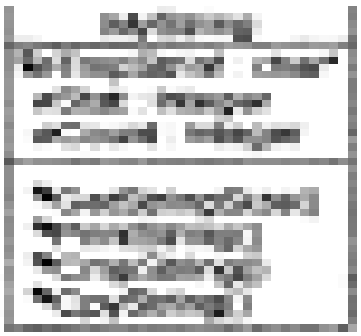
Үлкен класстарды жобалау керек кезде барлық өңдеушілер ахуалға тап болады. Қолмен енгізгенде және хабарлағанда келесілер болады: біріншіден, есепті қоюшы ереже бойынша не істеу керектігін сөз жүзінде немесе қағаз жүзінде айтады, екіншіден, жүйені құрушы өңдеуші программалық код сүйемелдеу керек түсініктемелерді көп жағдайда ескермейді. Rational Rose код генерациялаушы жүйесі жобалаудың басқа құрылғыларымен бірге программалық қамтаманың өңдеу процесін рөлдері, қызметтері және т.б қатаң бөлінген өндірістік процесс сияқты құруға мүмкіндік береді. Мақсаттарды көрсету үшін бір ғана классты жобалау жеткілікті. Оны String деп атайық. Оның міндетіне массивтердегі негізгі операциялар кіруі керек(басып шығару, көшіру, салыстыру, өлшемді алу). Мысал ретінде берілген классты C++-те сипаттаймыз:

```
Class String

Protected:

 Char *TmpString;

Public:

Int Counter;

Int Stat;

 Int GetStringSize(Cliar *);

Int PrintString(Char *);

Int CmpString(Cliar *, Char *);

Int CpyString(Char *, Char *); }
```

Енді Rose құралдарымен барлығын графикалық түрде жобалаймыз. Әрбір атрибут түсініктемелерімен жеке беріледі және тип жазылады (public, protected, private). 15.1 сурет

15.1 сурет

Осындай жолмен барлық айнымалылар жазылады. Функцияны сипаттау кезінде барлығы ұқсас, тек функцияның өзін сипаттаудан өзге (қайтарылатын мәннің типі) нақты түсініктемелерді беріп әрбір кіріс параметрлерінің ерекшелігін жазу керек. Rational компаниясының барлық өнімдерінде ақырында басшылыққа көрсету үшін немесе өңдеушіге техникалық тапсырма ретінде беру үшін есепті генерациялау кезінде құжатты қолмен жеткізудің қажеті болмайтындықтан кішігірім операциялардың барлығына түсініктемелер беру бекітілген. Жоғарыда көрсетілген әрекеттердің нәтижесі болып класстың көрсетілген ерекшеліктерімен пайда болуы болып табылады. Rational Rose 98 дайын кодты бере алмайды, ол жай ғана классты жобалай алады және әрбір элементтің ерекшеліктерін жаза алады, класс мүшелеріне ары қарай кодпен толтыруы үшін үлгі бере алады. Ал C++-те жұмыс кодын 100 %-ды генерациялау үшін берілген жағдайда қарастылырмайтын Rational Rose RealTime қолданылады. Сонымен, код генерациясына оралайық(анығырақ, класс генерациясына). Меню(Tools) жүйесі арқылы жобаланатын классты сипаттау үшін қолданылатын тілді таңдаймыз, CodeGenerational-ды шақырамыз.



15.2 сурет

Нәтижесі болып MyString.h және MyString.cpp файлдарының пайда болуы табылады. Біріншісінде класстың өзі жазылады, ал екіншісі сәйкес кодпен толтырылуы үшін үлгі болып табылады. Төменде екі файлдың да жазылуы көрсетіледі. Келтірілген түгел материал жөндеулердің өзгеруінсіз және қосымша түзетусіз алынған. Бұндай үлгіге ие бола отырып, қандай өңдеуші класс логикасын кодтауды бастағаны маңызды емес болып қалады. Ары қарай класс бойынша нақты есепті немесе техникалық тапсырманы алу үшін Rational SoDA құралын пайдалануға болады. Келесі Rational Rose орындай алатын тапсырма әрекеттегі жүйені талдау. Егер бар кодтан визуалды модельді құрастыруға болатын және қажетті қасиеттерді және атрибуттарды аяғына дейін визуалды жазуға, жаңа класстарды жазуға мүмкіндік беретін қайта жобалау функциясын қолдануға болса, онда көлемді жүйелерді жаңадан қайта жазып және құжаттап керегі жоқ. Ал соңында бағдарламалаушының ары қарайғы жұмысы үшін барлық файлдарды генерациялау қажет. Бұндай тәсіл Round-Trip деп аталады және RationalRose-та түгел бар.

Тапсырма

1. Бірнеше класстар жүйесі үшін кодты генерациялау және модель мен алынған негізгі кодты салыстыру.
2. Код генерация мысалы
   Файл MyString.h
   //## begin module%1.3%.codegen_version preserve=:yes
   // Read the documentation to learn more about C+-)- code generator versioning.
   //## end module%1.3%.codegen_version

   //##begin module%395AF70D0321.cm preserve=no

```
// %X% %Q% %Z% %W%
//## end module%395AF70D0321.cm
//## begin module%395AF70D0321.cp preserve=no /
/## end module%395AF70D0321.cp
//## Module: MyString%395AF70D0321; Pseudo Package specification
//##Source file: C: \Program Files\Rational\Rose\C++ \source\MyString.h
#ifndef MyString_li
#define MyString__h 1
//##begin module%395AF70D0321.additionalIncludes preserve=no
//##end module%395AF70D0321.additionalIncludes
//##begin module%395AF70D0321.includes preserve=yes
//##end module%395AF70D0321.includes
//##begin module%395AF70D0321.additionalDeclarations preserve=yes
//##end module%395AF70D0321.additionalDeclarations
//##begin MyString%395AF70D0321.preface preserve:=yes
//##end MyString%395AF70D0321.preface
//## Class: MyString%395AF70D0321 // Данный класс позволяет проводить различные операции // над массивами символов.
 //##Category: <Top Level>
//##Persistence: Transient //ФФ Cardinality/Multiplicity: n
class MyString
{
//##begin MyString%395AF70D0321.initiaroeclarations preserve=yes
 //##//ФФ end MyString%395AF70D0321.initialDeclarations

public:
//## Constructors (generated)
MyStringO;
//## Destructor (generated) –
MyStringO;
//## Assignment Operation (generated)
MyString &; operator= (const MyString fcright); /
/## Equality Operations (generated)
int operator==(const MyString &right) const;
int operator!=(const MyString bright) const;
//## Other Operations (specified)
//## Operation: GetStringSize%395AF87900E9
```

```
// Подсчитывает количество символов в переданном массиве IntGetStringSize
(Char *massiv // Указатель на массив

);
        //## Operation: PrintString%395AF88800B9
        // Печатает на экране переданный массив
        Int PrintString (Char *Massiv // Указатель на массив );
        //## Operation: CmpString%395AF892013F
        // Сравнивает два массива.
        Int CmpString (Char *Strl, // Указатель на первый массив
        Char *Str2 // Указатель на второй массив );
        //## Operation: CpyString%395AF89C00D5
        // Копирует один массив в другой
        Int CpyString (Char *Dest, // Назначение Char * Source // Источник );
        //## Get and Set Operations for Class Attributes (generated)
        //## Attribute: Stat%395AF8BB0289
        // Общедоступная переменная числа обращений к Print St ring
        const Int get _ St at () const;
        void set_Stat (Int value);
        //## Attribute: Count%395AF8C20148
        // Определяет статус определенного объекта
        const Int get_Count () const;
        void set_Count (Int value);
        //Additional Public Declarations
        //## begin MyString%395AF70D0321.public preserve^^yes
        //## end MyString%395AF70D0321.public
        protected:
        // Additional Protected Declarations
        //## begin MyString%395AF70D0321.protected preserve==yes
        //#7^ end MyString%395AF70D0321.protected
        private:
         //## Get and Set Operations for Class Attributes (generated)
        //## Attribute: TmpString%395AF8B201E5
        // Временный указатель на строковый массив. Можно использовать
        // в качестве буфера
        const Char * get_TmpString () const;
        void set_TmpString (Char * value);
        // Additional Private Declarations
        //## begin MyString%395AF70D0321.private preserve==yes
        //## end MyString%395AF70D0321.private
```

```
private:
//## implementation
// Data Members for Class Attributes
//## begin MyString::TmpString%395AF8B201E5.attr preserve=no
// private: Char * U Char *TmpString;
//## end MyString::TmpString%395AF8B201E5.attr
//## begin MyString::Stat%395AF8BB0289.attr preserve^no public: Int U Int Stat;
//## end MyString::Stat%395AF8BB0289.attr
//## begin MyString::Count%395AF8C20148.attr preserve=:no public: Int U Int Count;
//## end MyString::Count%395AF8C20148.attr
// Additional Implementation Declarations
//## begin MyString%395AF70D0321.implementation preserve=yes
//## end MyString%395AF70D0321.implementation };
//## begin MyString%395AF70D0321.postscript preserve=yes
//## end MyString%395AF70D0321.postscript
// Class MyString //i^i^ Get and Set Operations for Class Attributes (inline)
inline const Char * MyString: :get_TmpString () const {
 //## begin MyString::get_TmpString%395AF8B201E5.get preserve=no return TmpString;
//## end MyString::get_TmpString%395AF8B201E5.get }
inline void MyString: :set_TmpString (Char * value) {
//## begin MyString::set_TmpString%395AF8B201E5.set preserve==:no TmpString = value; /
/## end MyString::set_TmpString%395AF8B201E5.set }
 inline const Int MyString: :get_ Stat () const {
//## begin MyString::get_Stat%395AF8BB0289.get preserve=no
return Stat;
//## end MyString::get_Stat%395AF8BB0289.get
}
inline void MyString::set_Stat (Int value)
{
//##begin MyString::set_Stat%395AF8BB0289.set preserve==no Stat = value;
 //## end MyString::set_Stat%395AF8BB0289.set
}
inline const Int MyString::get_Count () const {
```

```
//## begin MyString::get_Count%395AF8C20148.get preserve=:no return Count;
//## end MyString::get_Count%395AF8C20148.get
}
inline void MyString::set_Count (Int value) {
//## begin MyString::set_Count%395AF8C20148.set preserve=no Count = value;
//## end MyString::set_Count%395AF8C20148.set }
//## begin module%395AF70D0321.epilog preserve=:yes
//## end module%395AF70D0321.epilog #endif Файл MyString.cpp
//## begin module%1.3%.codegen_version preserve=yes // Read the documentation to learn more about C+4- code generator versioning.
//##end module%1.3%.codegen_version
//## begin module%395AF70D0321.cm preserve=no // %X% %Q% %Z% %W% //ΦΦ end module%395AF70D0321.cm
//## begin module%395AF70D0321.cp preserve=no
//## end module%395AF70D0321.cp
//## Module: MyString%395AF70D0321; Pseudo Package body
//##Source file: C:\Program Files\Rational\ Rose\C-f 4-\source\MyString.cpp/ЦΦΦbegin
module%395AF70D0321.additionalIncludes preserve=no
 //## end module%395AF70D0321.additionalIncludes
//## begin module%395AF70D0321.includes preserve=yes
//## end module%395AF70D0321.includes
// MyString
#include"MyString.h"
//##begin module%395AF70D0321.additionalDeclarations preserve=yes
//## end module%395AF70D0321.additionalDeclarations
// Class MyString MyString: :MyString()
//## begin MyString::MyString%395AF70D0321_const.hasinit preserve=no
//## end MyString::MyString%395AF70D0321_const.hasinit
//##beginMyString::MyString%395AF70D0321_const.initializationpreserve=yes
//## end MyString::MyString%395AF70D0321_const.initialization {
//## begin MyString::MyString%395AF70D0321_const.body preserve=yes
//## end MyString::MyString%395AF70D0321_const.body } MyString:: MyString() {
```

```
//## begin MyString:: MyString%395AF70D0321_dest.body preserve=yes
//## end MyString:: MyString%395AF70D0321_dest.body } MyString &
MyString::operator= (const MyString bright) {
//##begin MyString::operator=%395AF70D0321_assign.body preserve—
yes
//##end MyString::operator=:%395AF70D0321_assign.body
int MyString::operator==(const MyString &right) const {
//##begin MyString::operator==%395AF70D0321_eq.body preserve=yes
//## end MyString::operator==%395AF70D0321_eq.body }
int MyString: :operator!=(const MyString &right) const {
//## begin MyString::operator!=%395AF70D0321__neq.body preserve=yes
//## end MyString::operator!=:%395AF70D0321_neq.body }
//##Other Operations (implementation) Int MyString::GetStringSize (Char
*massiv) {
//## begin MyString::GetStringSize%395AF87900E9.body preserve=yes
//## end MyString: :GetStringSize%395AF87900E9.body }
Int MyString: :PrintString (Char *Massiv) {
//## begin MyString::PrintString%395AF88800B9.body preserve=yes
//## end MyString::PrintString%395AF88800B9.body }
Int MyString::CmpString (Char *Strl, Char *Str2) {
//## begin MyString::CmpString%395AF892013F.body preserve=yes
//## end MyString::CmpString%395AF892013F.body }
Int MyString::CpyString (Char *Dest, Char *Source) {
//## begin MyString::CpyString%395AF89C00D5.body preserve^yes
//## end MyString::CpyString%395AF89C00D5.body }
// Additional Declarations
//## begin MyString%395AF70D0321.declarations preserve=:yes
//## end MyString%395AF70D0321.declarations
//## begin module%395AF70D0321.epilog preserve=yes
//## end module%395AF70D0321.epilog
```

3. Бақылау сұрақтары.

1. Автоматты код генерациясының артықшылықтары қандай?
2. Код генерациясы үшін қандай диаграмма түрлері қолданылады?
3. RationalRose қандай негізгі код компоненттерін генерациялайды?
4. Негізгі кодта атрибуттар мен класс операциялары қалай көрсетіледі?
5. Модель компонеттеріне дәл түсініктеме берудің маңыздылығы қандай?