KRLab软件的开发手册

[第一部分 KRLab系统 3](#_Toc64888854)

[第1章 整体说明 3](#_Toc64888855)

[1.1 知识网开发 4](#_Toc64888856)

[1.2 ITS系统 4](#_Toc64888857)

[1.3 数据及其文件 6](#_Toc64888858)

[第2章 KRLab的Core模块 7](#_Toc64888859)

[2.1 Project类 7](#_Toc64888860)

[2.2 KRModuleSNet及其派生类 8](#_Toc64888861)

[2.3 DiagramEditor的Models类 8](#_Toc64888862)

[第3章 GUI 9](#_Toc64888863)

[2.1 主窗口 9](#_Toc64888864)

[2.2 工作区窗口 10](#_Toc64888865)

[2.3 画布 10](#_Toc64888866)

[2.4 工具栏 10](#_Toc64888867)

[2.5 标签栏 11](#_Toc64888868)

[2.5 Workspace 11](#_Toc64888869)

[2.6 ModelView 11](#_Toc64888870)

[第4章 DiagramEditor 13](#_Toc64888871)

[4.1. 图形绘制 13](#_Toc64888872)

[4.2. Diagram文件的导入 14](#_Toc64888873)

[第5章 添加新语义连接 14](#_Toc64888874)

[5.1 语义连接类型的作用 14](#_Toc64888875)

[5.2 KRLab中的语义连接类型 15](#_Toc64888876)

[5.3 如何在KRLab中增加新的语义连接类型 16](#_Toc64888877)

[5.4 如何在KRLab中更新已有的语义连接类型 16](#_Toc64888878)

[第6章 语义网类型 17](#_Toc64888879)

[6.1 一级语义网 18](#_Toc64888880)

[6.2 二级语义网 22](#_Toc64888881)

[第二部分 FVCTutor系统 22](#_Toc64888882)

[第8章 Domain Module 23](#_Toc64888883)

[8.1 知识组织(Knowledge Organization KO) 23](#_Toc64888884)

[8.2 ITS系统开发模块 25](#_Toc64888885)

[第9章 文件系统 25](#_Toc64888886)

[9.1文件结构 25](#_Toc64888887)

[9.2文件管理 26](#_Toc64888888)

[第10章 MaterialModule 26](#_Toc64888889)

[10.1 PQA类及其相关类 27](#_Toc64888890)

[10.2 PQAFactory类及其派生类 28](#_Toc64888891)

[第11章 Student Module 29](#_Toc64888892)

[11.1 Student类 29](#_Toc64888893)

[11.2 LearningStragety 30](#_Toc64888894)

[第12章 TutorModule 30](#_Toc64888895)

[12.1 Tutoring类 31](#_Toc64888896)

[12.2 TeachingScheme 31](#_Toc64888897)

[12.3 MaterialSelection 31](#_Toc64888898)

[12.4 ResultAnalyzer 32](#_Toc64888899)

[第13章 ClientSystem 32](#_Toc64888900)

[13.1 OnStart(Action<string> callback) 32](#_Toc64888901)

[13.2 OnChooseCourse (string course,Action<string> callback) 32](#_Toc64888902)

[13.3 OnStartAQuestion (Action<bool,string> callback) 33](#_Toc64888903)

[13.4 OnAnswerSubmitted (string answer,Action<string> callback) 33](#_Toc64888904)

[13.5 OnObtainCorrects (Action<string> callback) 33](#_Toc64888905)

[第14章 FVCTutorGUI 33](#_Toc64888906)

[14.1 主窗口 34](#_Toc64888907)

[第15章 FVCTutor的扩充 34](#_Toc64888908)

# 第一部分 KRLab系统

## 第1章 整体说明

KRLab软件系统有两个方面的应用，一是用于制作知识网络，以此构建知识库，二是用于创建应用系统，用于智能指导学习系统。

### 1.1 知识网开发

### 1.2 ITS系统

ITS-Intelligent Tutoring System，即所谓的智能指导系统，其主要目标是实现指导过程中的智能和自适应，所谓自适应就是系统能判断学生的知识水平(knowledge state)和喜好(prefernces)，以迎合不同学生的学习需求。ITS的基本框架包含以下几个核心模块：

* Domain module，知识表达模块，存储领域知识。该模块的构建需要大量的工作量，并且与特定的领域有关，不同的领域需要构建不同的领域知识，而且构建方法也不是一成不变，要根据指定的领域选用适当的建模方法。该模块包括了四个主要子模块：

DomainTopics，用于组织课程中的章节和学习课题的先后依赖关系。

DKR，domain knowledge representation采用语义网的方式存储学习主题中的概念、原理和公式。

MaterialRepository，用于处理课程中的学习资料，直接以文字、图片或声音的方式呈现给学生。

QAEninge，基于KnowledgeNetwork的问答系统，基于知识语义网，自动提出问题，并向学生提供相应的解答。

* Tutor Module，教学模块，用于制定适当的教学策略。

In any ITS, a tutor module is required which will control and conduct the tutoring processes of the system. This module is the nucleus of the ITS which communicates with the other modules and plans the tutoring trategies to be taken for individual students.

Among the various tasks it needs to carry out, one of the most important ones is to plan the sequence of the various curriculum and course elements to teach a particular course. 该模块必须与StudentModule紧密联系，实现对学生有针对性的个性化的指定。

PerformanceEvalution采用相关算法对学生学习结果进行评估。

FeedbackProvider将评估结果反馈给学生模块，并更新学生模块当中的参数，比如学生的认知水平，等等。

TSchemeMaker(tutoring scheme maker),根据Student的认知水平，形成一份详细的教案：（1）选定学习主题、明确教学目的、重点和难点；（2）制定相应的教学策略；（3）选定教学内容和材料；（4）选定测试材料。

* Student Module，学生模块。该模块要反映真实学生的认知能力，学习水平，以及学生与系统的交互。由于认知能力和学习水平与特定的知识有关，所以Student模块往往与Domain模块有关。可以用贝叶斯网和模糊逻辑进行学生模块的建模。StudentModule包括3个主要子模块：

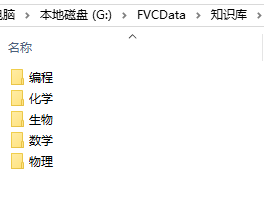
Student，

* TutorGUI，教师端的图形界面，用于查看学生学习结果相关的数据。
* StudentGUI，用户端GUI系统，设置Learning environment/User interfaces，用户交互。

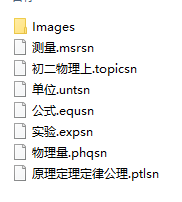
Domain module用于处理ITS的“教什么”的问题，而Teaching Module和Student module 实现“如何教”的问题。

### 1.3 数据及其文件

我们的ITS系统依赖于语义网文件，利用KRLab创建的语义网数据保存在XML文件中。目前，以不同学科分类，建立物理、数学、化学、生物和编程类文件夹。所有相关课程都放置在自己所在的学科文件夹底下。



每个学科文件夹下面包括不同类型的xml文件，每一个这样的文件保存了一类知识的语义网。比如物理下面的文件如下：



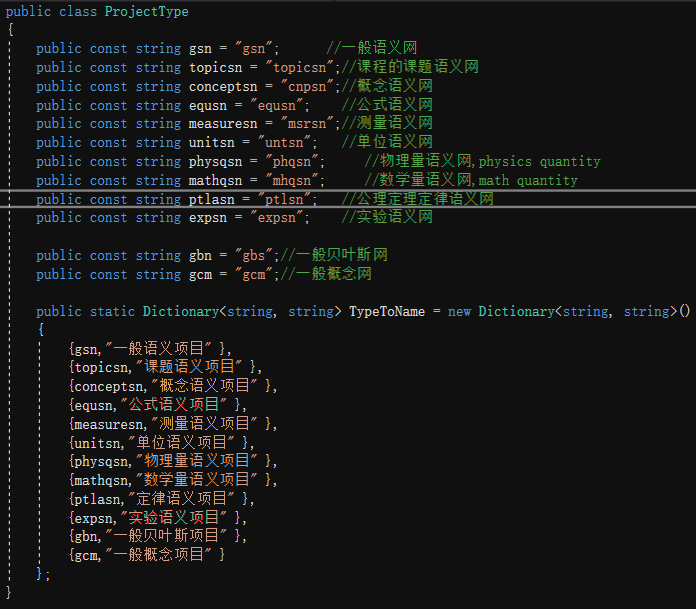
不同后缀名的文件表示了一类语义网，后缀名非常重要，不能改动。

## 第2章 KRLab的Core模块

该模块包括了系统的核心数据结构和算法，以及语义网、贝叶斯网、BDI、模糊逻辑和概念图。

### 2.1 Project类

这个类是KRLab编辑器建立的一个项目，对应一个KRLab文件。点击KRLab编辑器的新建项目，选择适当的类型，如果选择gsn，表示构建一般语义网。

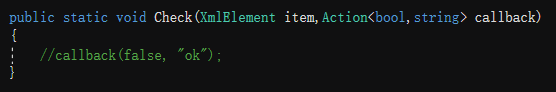


### 2.2 KRModuleSNet及其派生类

KRModuleSNet及其派生类描述各种类型知识的语义网。目前，对应ProjectType类型主要建立了如下几个知识类型模块：

* ConceptKRModuleSNet,
* DomainTopicKRModuleSNet,
* EquationKRModuleSNet,
* ExperimentKRModuleSNet,
* MathQuantityKRModuleSNet,
* MeasureKRModuleSNet,
* PhysQuantityKRModuleSNet,
* PTLAKRModuleSNet,
* UnitKRModuleSNet,

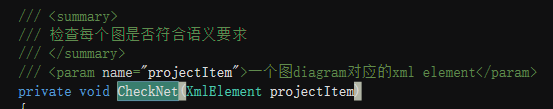
以上模块在进行建模时，要满足一定的规范，所以在每个函数中提供了一个静态函数进行检查：



该函数在Project类的CheckNet函数中被调用。

### 2.3 DiagramEditor的Models类

保存文件时不进行语法检查。语法检查在Model类中的CheckNet函数中进行：

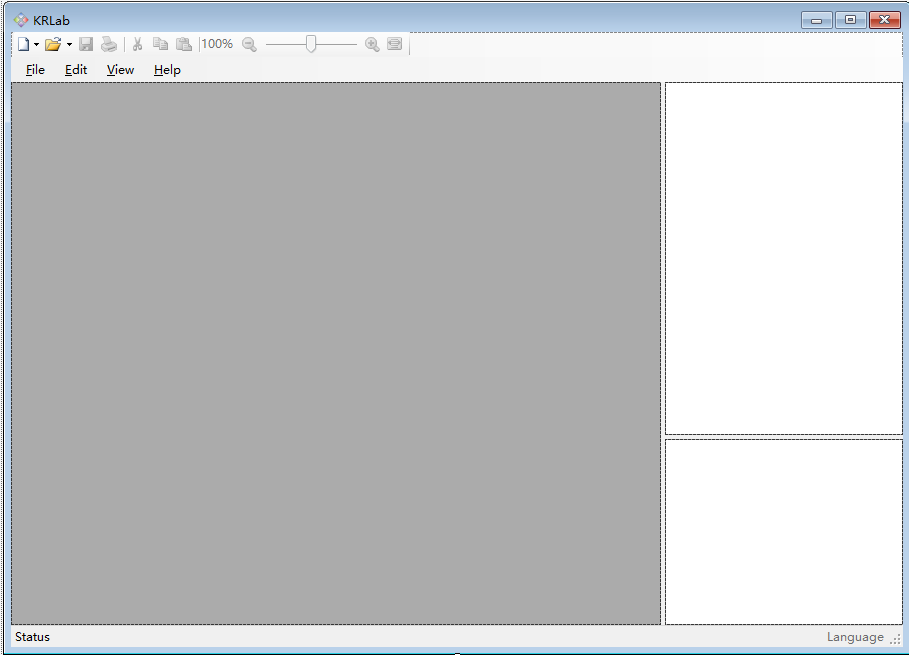


为了表达各种不同类型的知识，KRLab用文件的后缀名区分。目前，建立了多个文件类型，用ProjectType类表示。

## 第3章 GUI

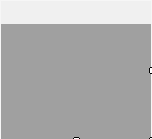
本章从用户使用的角度，介绍KRLab的GUI系统的各个功能模块的实现细节。

### 2.1 主窗口



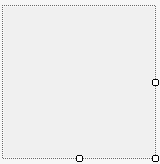
主窗口的代码对应GUI下的MainForm.cs。

### 2.2 工作区窗口



嵌在主窗口中，对应主窗口左边的灰色区域，相关代码文件为TablledWindow.cs

### 2.3 画布



在工作区窗口TabbedWindow中定义了一个画布实例，代码文件是DiagramEditor. Canvas.cs。

* 用于图形显示，图形放大ZoomOut和缩小ZoomIn显示。
* Canvas类中的document用于指向当前打开的Diagram。
* 重载了UserControl和Control中的OnKeyDown, OnMouseDoubleClick, OnMouseUp, OnMouseMove和OnMouseDown的函数，用于鼠标和键盘响应。从而操作画布当前文档，也就是当前打开的一个Diagram。
* 重载了OnPaint函数，用于画布的图形绘制。

### 2.4 工具栏

DiagramDynamicMenu类处理GUI的工具栏，



### 2.5 标签栏



标签栏在工作区窗口的上边，用于显示打开的每个Diagram名称。代码文件为GUI.TabBar。

### 2.5 Workspace

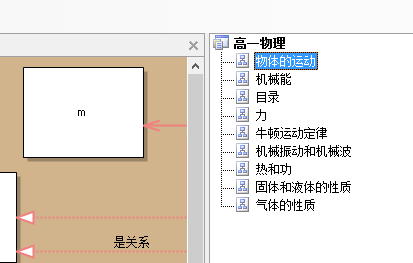
* KRLab软件的工作空间，保存了所有的工程Project：

List<Project> projects=new List<Project>();

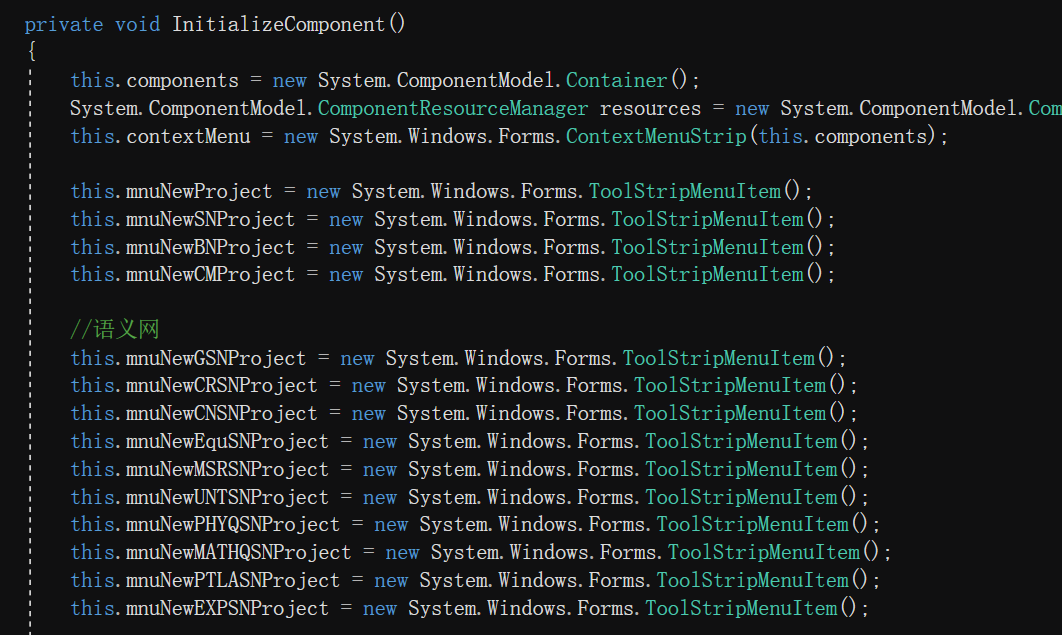
* 并维持一个当前活动工程activeProject;
* 保存一个Project：SaveProjectAs()
* 打开一个Project： OpenProject()

### 2.6 ModelView

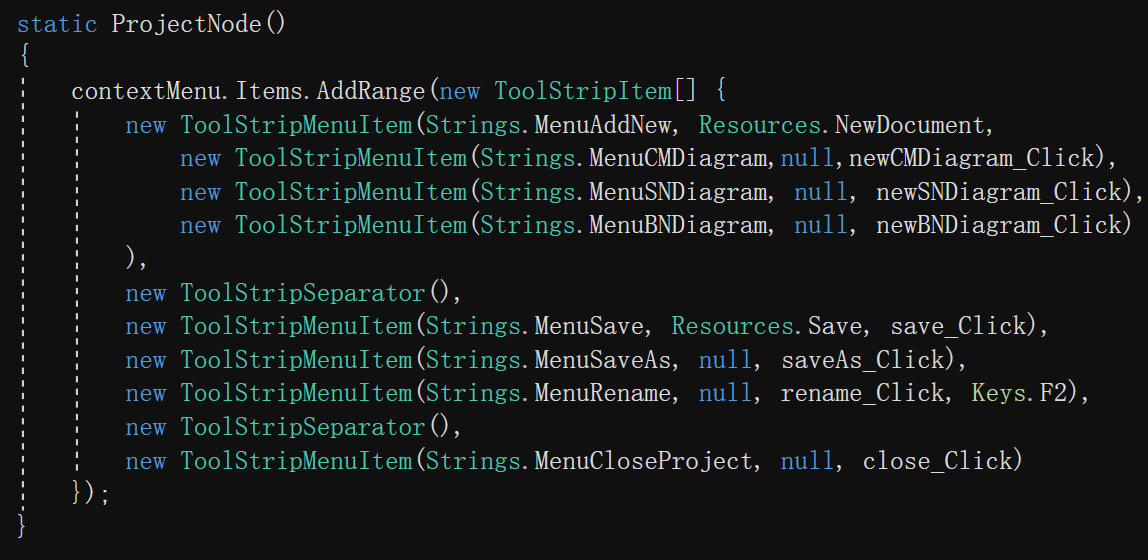
ModelView类创建和管理右边窗口，用于显示项目名称和Diagram名称列表，如下图所示。



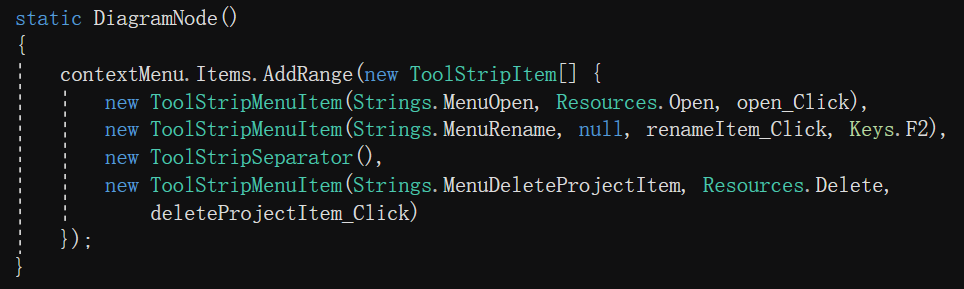
* 在MianForm的InitializeComponent()函数中初始化处理鼠标右键点击窗口出现的窗口，



* ProjectNode类创建和管理右边窗口的项目节点。EmptyProjectNode类处理空项目节点。下面的代码负责处理选中项目节点，点击右键出现的窗口：



* DiagramNode类创建和管理右边窗口项目节点下的图节点。下面的代码负责处理选中图节点点击鼠标右键出现的窗口，



## 第4章 DiagramEditor

Diagram对应KRLab的一个工作区，在每个工作区编辑图形。一个KRLab项目可以包含多个Diagram。通过画布Canvas与用户进行交换。请参见Canvas。

### 4.1. 图形绘制

在Canvas类的DrawContent()函数中调用Diagram类的Display()函数进行图形绘制。

1.1 结点的绘制：在SimpleNodeShape.Draw()函数中。

2. 图形的拷贝，剪切，粘贴和删除功能

* 图形拷贝：Diagram::Copy()函数负责图形的拷贝
* 图形剪切：Diagram::Cut()函数负责图形的剪切
* 图形粘贴：Diagram::Paste()函数负责图形的粘贴
* 图形删除：Diagram::KeyDown(KeyEventArgs e)函数中处理Key.Delete键的响应。

3. 图形的移动和缩放处理

* 图形移动：Diagram::MouseMove()函数进行处理
* 图形缩放：在Diagram中指定Shape的事件响应函数：

Shape.Resizing+=new ResizeEventHandler(shape\_Resizing);

Shape.OnMouseMove()函数中调用PerformResize()函数。

在Shape.PerformResize()函数中调用Resize回调函数，该函数定义在Diagram中，为shape\_Resizing()函数，所以正真处理图形缩放的是Diagram中的shape\_Resizing()函数和Shape中的PerformResize()函数。

最后在Shape.Size成员属性中进行大小的控制。

* 图形的最小值控制

在调整图形大小时，受到系统的最小值设置。在Shape中设定了缺省的最小值。在NodeShape类中有一个函数UpdateMinSize()可以对最小值进行更新，目前该函数被设置为空，不起作用，将来可以进行改进。

### 4.2. Diagram文件的导入

导入Diagram文件时要可视化文件中所有的结点和边线，确定结点的位置和大小。所涉及的类和函数为：

* Model.Deserialize()
* Model.LoadEntities()
* Shape.Deserialize()
* Shape.OnDeserializing()

## 第5章 添加新语义连接

### 5.1 语义连接类型的作用

If one wants to connect entities of a knowledge representation by typed realations, an inventory of relations is needed. Ideally, this inventory should be characterized by universality and domain-independent applicability. Such properties would ensure its usability as broad and independent as possible from a specific domain knowledge.

Each application domain usually has a specific spectrum of relationships that can be established between its concepts. An illustrative example of the domain-specific range of possible relations are properties and roles that can be taken by individuals and that can lead to relationships to other entities of a knowledge representation.

It can be expected that the total amount of relations would exceed any vision of a managable measure. So a clear demand for decreasing the number of relations appears.

To represent a document aboutness, it is necessary to remember the distinction between two fundamentally different forms of relationships between concepts:

(1) Relationshps between concepts that are independent of any reference to documents. We can interpret them as parts of world knowledge and have already introduced them as a priori relationships. We model them as different types of relations between entities within the kwonledge representation.

(2) Relationships between concepts that are given only by the aboutness of the document. By this aspect, they exist independently from any relation to world knowledge.

### 5.2 KRLab中的语义连接类型

(1）ISA, 是一个 is a

(2）IS, 是关系，也用来表达父子关系Parent-Child，宽窄Broader-narrower关系，

(3) ISO, 其中之一，is one of，群体与个体关系，比如人群-个人

(4) ISP, 组成部分，

(5) ASSOC，关联关系

(6) ATT，表示一个事物的属性，比如人的身高，年龄等。或一个事件的属性，发生地点，时间等。

(7) TIME，表示概念在时间上的关系，先后关系，

(8) SPACE,//空间关系

(9) RESULT，推理关系，也可以表示一个事件的结果关系，因果关系，

(10) ACT, 与一个操作结点想连接,执行一个动作，引发一个事件，动作的执行者

(11) ACTED, 与一个操作结点相连接，表示操作动作的被作用者，

(12) DEPT，表示概念之间的依赖关系，

(13) COMP,//相互比较，比如性格、相貌、兴趣等等，可以处理逻辑关系：大于，小于，等于等等。

COMP

(14) has, 有。

以上语义关系在不同的知识领域有不同的解析。

### 5.3 如何在KRLab中增加新的语义连接类型

在语义连接编辑器上，为了查看方便，所有连接按特征进行分类，比如有“IS类型”，“行为”类型等等。为此，首先在SNConnectionDialog.Designer.cs中创建若干TreeNode，

在KRLab中添加新的连接语义类型，步骤如下：

（1）在枚举类型SNRelationshipType中添加枚举，用以描述新的连接类型；

（2）在SNRational类中添加对应的连接类型的字符表达，这是因为在语义解析中都是用名称进行解析的；

（3）到SNRelationship类中添加新连接类型的反序列化代码，用于语义网的数据读取。

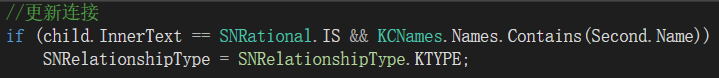
（4）在SNConnectionDialog类中的ChangeSNRelationType()函数中添加新连接类型的处理代码；

（5）到RelationTypeNode类中添加新连接类型，用于可视化连接选择。

### 5.4 如何在KRLab中更新已有的语义连接类型

首先，重新定义新的连接，

在SNRelationship类中的Deserialize()函数中修改并更换已有的连接类型，





在Relationship类中的Deserialize()函数中修改标签，



这样在打开文件时，自动修改已有的连接。

如果需要使某个不需要的连接无法使用，需要在RelationTypeNode类中删除相应的节点，使该连接编辑器上不再显示这个连接。

## 第6章 语义网类型

为了便于建模，语义网不能太过笼统。我们基于知识类型定义多种、多级，便于建模和解析。类型名称定义在KCNames类中。

public const string Catalogue = "目录";

public const string Concept = "概念";

public const string Measure = "测量";

public const string Unit = "单位";

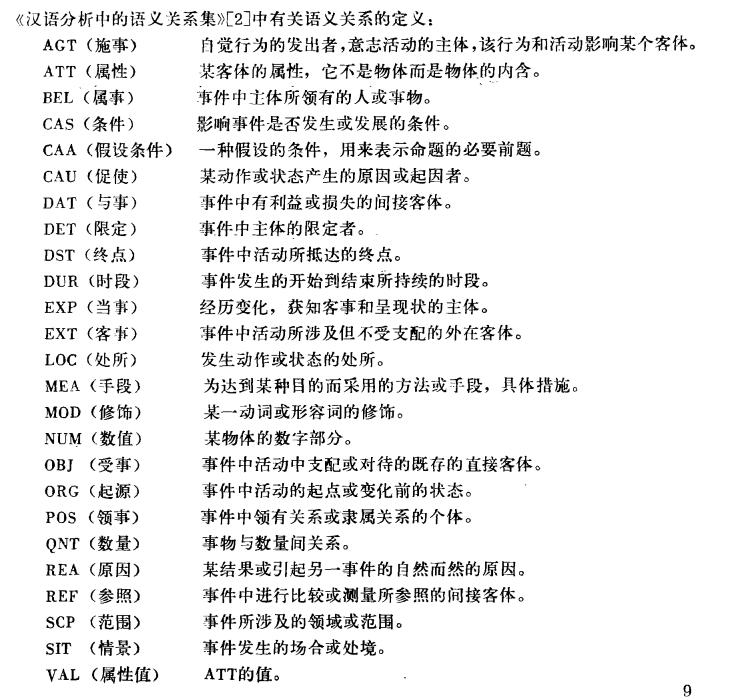
public const string Equation = "公式";

public const string PTLA = "原理定理定律公理";

public const string Experiment = "实验";

public const string Phenomena = "现象知识";

public const string Procedural = "过程";



### 6.1 一级语义网

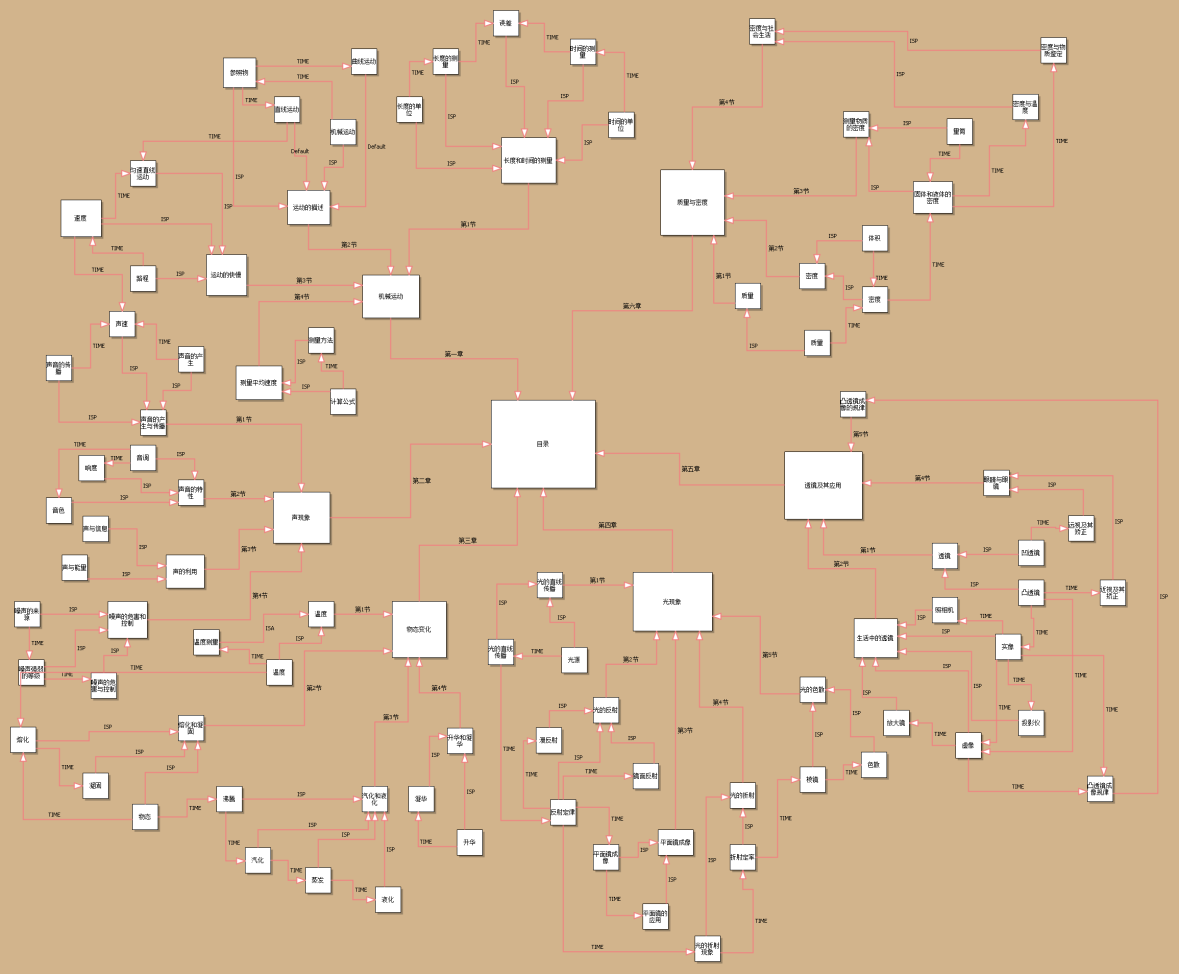
一级语义网项目下的每个语义图对应单个知识点，这个知识点与知识类型用ISO关系连接，表示这个知识点是“其中之一”。一级语义网主要包括，概念语义网，公式语义网，定律定理原理语义网，现象知识语义网，过程语义网。

（1）课程目录语义网

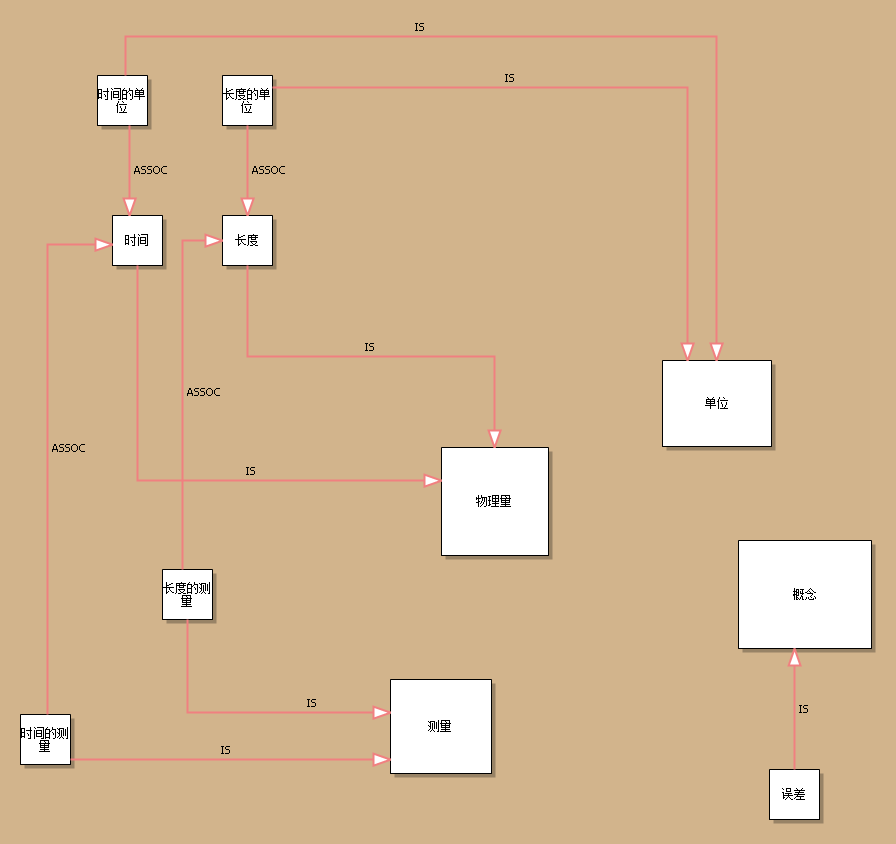
主要包括如下几个语义项目：

* 以课程名称命名，建立“课程结构语义项目”。

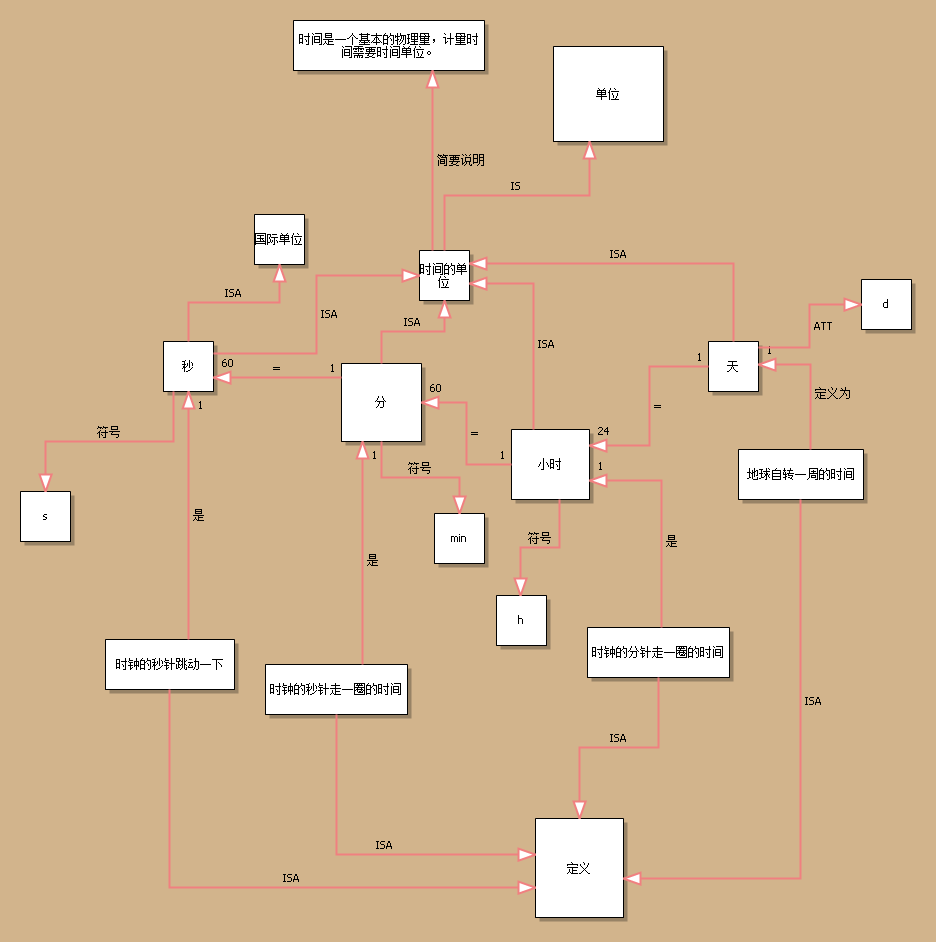
课程结构语义项目包含多个语义图。第一个是“目录”语义图，该语义图严格按课程的章节顺序，建立各个学习课题的学习网络，即课题之间的关联关系，以此构建各个学习课题的学习先后顺序。



* 除了“目录”语义图，针对课程的每一章，以章的名称，建立每一章的语义图。特别注意，语义网的名称必须与目录语义网的章名称+连接标签构成的全名，比如“第一章 机械运动”，如图所示。

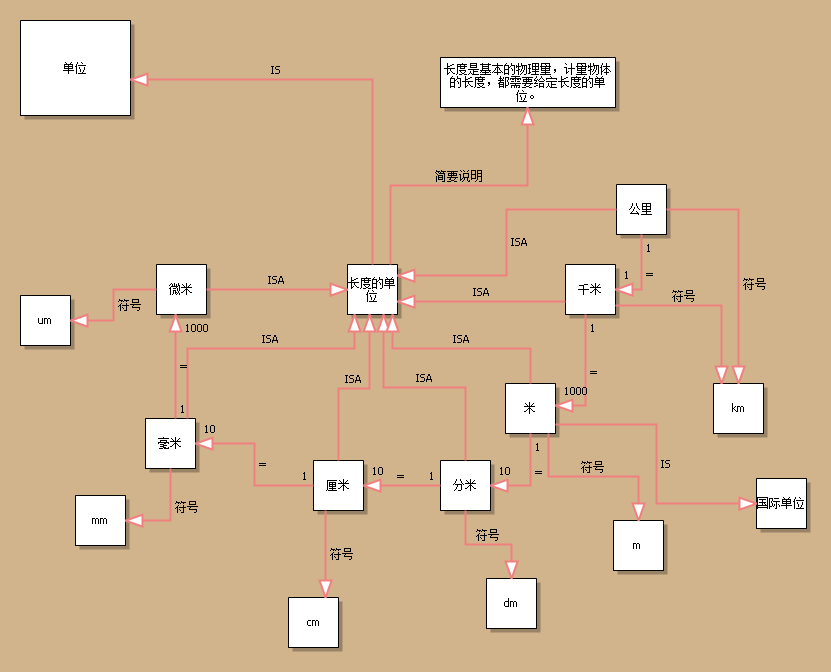


该语义网以目录语义网为基础，建立知识点之间的依赖关系，以及每个知识点的知识类型。首先从目录语义网获取到一个学习课题，再到章节语义网中查询该学习课题（注意，章节语义网中一定要有该课题）。找到该学习课题之后，再查询是否有依赖知识点，如果有，先学习依赖知识点。



（2）单位语义项目

每个学习课题图必须包含一个知识类型名称的结点。比如，“单位”。学习课题结点，“长度的单位”，与知识类型结点相连，连接类型为KTYPE。学习课题结点下面就是该学习课题的所有知识。学习课题图的名称尽量与学习课题结点名称一致。



下图是另外一个学习课题，时间的单位

（3）概念语义网项目

根据认识论中概念的定义进行概念语义网的构建，主要包括以下几个方面的内容：

* 概念的内涵和外延
* 上位概念和下位概念
* 内涵中的特征，该特征与上位概念之间的联系
* 区别特征，即该概念与邻近概念的区别特征
* 改变的方法，如果概念“是”一个量值，而不是“具有”量值的属性，则要求提供改变该量值的方法。这里所谓的量值，也就是标量，但量值要指明改变量值的方法。是本项目特定引入的一个类型，该类型特别强调其改变方法。

（4）公式语义网项目

（5）定理定律原理语义网项目

（6）现象知识语义网项目

（7）过程语义网项目

过程语义网主要用来描述步骤、过程和算法知识。

### 6.2 二级语义网

（1）课题语义网

（2）测量语义网

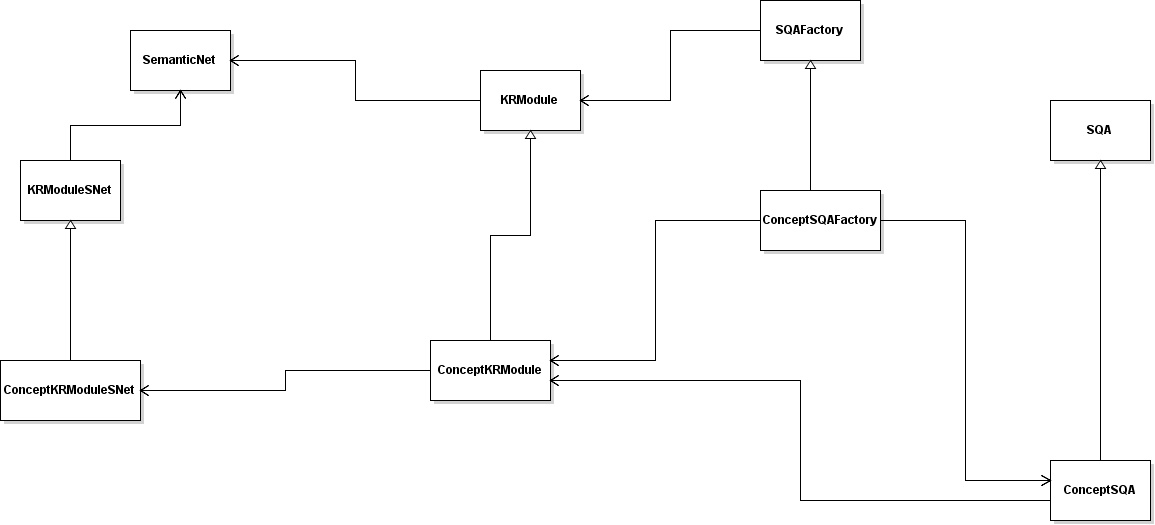
（3）单位语义网

（4）实验语义网

一级语义网会引用二级语义网的知识。

# 第二部分 FVCTutor系统

FVCTutor系统的核心是ITSEngine，顾名思义就是ITS的开发引擎。ITSEngine包括四个功能模块DomainModule, MaterialModule, StudentModule, TutorModule，以及ITSEngine系统与客户端图形用户界面GUI的接口模块ClientSystem，语音处理模块ITSSpeech。下面分别介绍这些子系统。下图，虚线左边是语义网，中间是基于语义网的知识模块，右边是基于知识模块的学习材料模块。



## 第8章 Domain Module

领域(Domain)模块是系统的知识库，决定了ITS系统“学什么”的问题。领域模块包括了知识组织（Knowledge organization）和学习材料工厂(learning Material Factory)两部分。目前，前者是用语义网的方法将知识建模，保存在文件中。后者是基于知识组织进行的问答系统(QA system)。

### 8.1 知识组织(Knowledge Organization KO)

FVCTutor系统所涉及的所有知识都以语义网的方法构建，用本项目开发的KRLab软件进行制作。相关语义网的定义，参见第一部分。按知识类型，建立了多个C#类，分别对应每一个知识类型。

#### 8.1.1 ConceptKRModuleSNet

概念语义网络，对应某个单独建模的概念语义图。

#### 8.1.2 ConclusionKRModuleSNet

结论语义网络，对应单独建模的结论语义图。

#### 8.1.3 DomainTopicKRModuleSNet

某门课程的目录和小节结构。

#### 8.1.4 EquationKRModuleSNet

公式语义网络，对应单独建模的某个公式。

#### 8.1.5 ExperimentKRModuleSNet

实验语义网络，对应单独建模的某个实验。

#### 8.1.6 InstrumentKRModuleSNet

器材语义网络，对应单独建模的某个实验器材。

#### 8.1.7 PhenomenaKRModuleSNet

现象知识语义网络，对应单独建模的现象知识。

#### 8.1.8 ProceduralKRModuleSNet

流程或算法知识语义网络，对应单独建模的流程或算法。

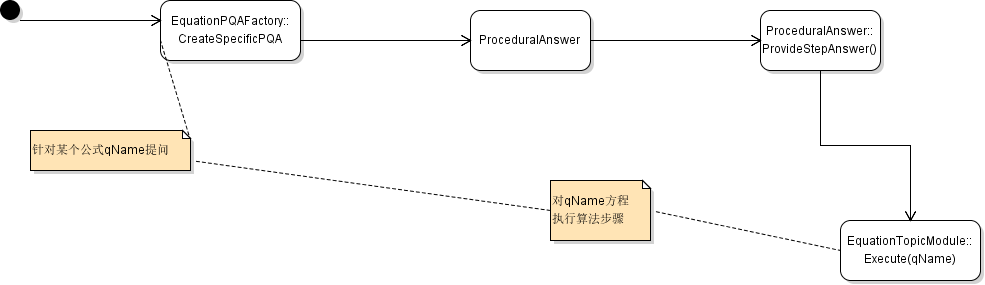
#### 8.1.9 UnitKRModuleSNet

物理或数学单位语义网络，对应单独建模的单位。

#### 8.1.10 公式语义网及其相关类

公式语义网用于表达公式和方程，特别重要的是要结合算法语义网进行方程的求解。为了指导学生学习，需要根据算法语义图对公式语义图进行修改和变换，以此反映公式的求解过程。

EquElem类，封装了一个语义图中的某个公式，包括整个公式形成的一个子语义网，所以公式的变换、方程的求解所设计的过程在EquElem类中进行。



### 8.2 ITS系统开发模块

在系统开发中，建立了一个DomainModule子系统进行处理，包括KRModule及其派生出的若干个处理各种类型知识的子模块。

一个类型的KRModule对应一个语义项目（project），封装其课程、知识类型，用来对一个项目中的内容处理。

共有属性：给出某类知识的共同属性；获得语义项目；获得项目中的所有语义图；获取特定名称的语义图等。

重要方法: public abstract TopicModule CreateTopicModule(string netName);

为一个语义图创建一个学习课题。这块内容和8.3 TopicModule紧密结合。

#### 8.2.1领域课题模块DomainTopicKRModule，

该模块主要处理课程语义项目的数据处理和知识查询。比如获取某章某节的知识点。

直接实例化DomainTopicModule并返回

#### 8.2.2 单位知识模块UnitKRModule

从KRModule派生，用于单位语义项目数据处理。

直接实例化UnitTopicModule并返回

#### 8.2.3概念知识模块ConceptKRModule

直接实例化ConceptTopicModule并返回

#### 8.2.4测量知识模块MeasureKRModule（无）

#### 实验知识模块ExperimentKRModule

直接实例化EquationTopicModule并返回

#### 器材知识模块InstrumentKRModule

直接实例化InstrumentTopicModule并返回

#### 8.2.5 物理量知识模块PhysicsQuantityKRModule

还没建，有QhysicalQuantity类

#### 8.2.6 数学量知识模块MathQuantityKRModule（无）

#### 8.2.7 公式知识模块EquationKRModule

直接实例化EquationTopicModule并返回

#### 8.2.8 原理/公理/定律/定理知识模块PTLAKRModule

没建。目前放在ConclusionKRModule中

直接实例化ConclusionTopicModule并返回

#### 8.2.9 流程知识模块ProceduralKRModule

直接实例化ProceduralTopicModule并返回

### 8.3 TopicModule类及其派生类

封装一门课程和其中一个具体的学习课题及其对应的semantic net，为解析一个知识点的关键内容提供各种方法。为PQAFactory提供问答的解析基础。

#### 8.3.1 TopicModule

封装课程、学习课题（topic）和该课题对应的语义网络。学习课题大部分情况下与语义网络名称相同，但有时只是语义网络中的某个知识点，所以对应的语义网络有可能为空。比如在“七年级数学上.概念”语义项目里，“有理数”语义图，是一个课题（topic），分数和整数是它的关联知识点，他们有自己的语义网络；“角”语义图也是一个topic，但对于它的关联知识点“角度”则没有单独建模。

重要方法：用于学习结果的反馈

public virtual string Parse(){}

public virtual string Parse(string name){}

#### 8.3.2 DomainTopicModule

#### 8.3.3 ConceptTopicModule

解析上位概念和外延概念

#### 8.3.4 ConclusionTopicModule

#### 8.3.5 EquationTopicModule

#### 8.3.6 UnitTopicModule

获得单位的符号表示，国际单位，单位间的转换关系等。

#### 8.3.7 ExperimentTopicModule

#### 8.3.8 InstrumentTopicModule

#### 8.3.9 PhenomenaTopicModule

#### 8.3.10 ProceduralTopicModule

### 8.4其他杂类

#### 8.4.2ChapterItem和SectionItem类

SectionItem的存储格式，比如“第1节 时间和长度”，属性Index为“第1节”，属性Name为“时间和长度”；ChapterItem的存储格式，比如“第1章 机械运动”，属性Index为第1章，属性Name为“机械运动”。

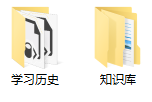
#### 8.4.2 LearningTopic类与KnowledgeTopic类

KnowledgeTopic类和LearningTopic类：两者都是描述学习内容的类，不同的是LearningTopic描述的是小节下学习课题的章节分布及权重分布，有单独建立语义网，有章节课题权重；KnowledgeTopic不仅可以描述学习课题的分布，还可以对应学习课题下更小知识点的分布，这个知识点没有单独建语义网，依附在学习课题下建的，没有详细权重。KnowledgeTopic是LearningTopic的父类。

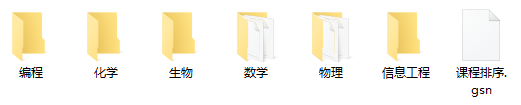
## 第9章 文件系统

### 9.1文件结构

FVCTutor系统中的知识库是以文件形式保存，在根目录FVCData下，有两个文件夹，学习历史和知识库，分别用于保存学习的相关数据和知识语义网络，



以及一个名为UserLog的学习日志文件。在知识库文件夹下面按学科类型建立对应的文件夹以及学科中的课程排序文件，



在每个学科文件夹下，按知识类型，对应不同年级的课程，建立相应的语义网络。每个文件的命名严格规定：课程名称+后缀，比如，七年级数学上，这门课程，我们建立的文件如下，



在学科目录底下，有两个文件夹，图片和音效，分别保存相关的图片和声音文件，



### 9.2文件管理

## 第10章 MaterialModule

该模块是基于知识模块产生学习材料，包括问题的故事，问题和答案。与领域知识模块中知识类型相对应，该模块建立了相应的问题产生工厂，生产每类知识的问题，用于学生学习。

针对一个学习主题，该模块要构建一个学习的故事Story，问题Question和答案Answer。在FVCTutor系统中，用SQA类描述。

### 10.1 PQA类及其相关类

#### 10.1.1 PQA类

用于封装一个学习课题Topic相关的问题Problem、提问Question和提问所对应的答案Answer，用于构建学习课题的教学计划Teaching Scheme，实现ITS系统的教学功能。

#### 10.1.2 Problem类

给出学习问题，其内容主要包括问题的文本描述以及所涉及的图片、图形、动画和声音。一个学习课题的PQA中的Problem可以为空，即没有前面所说的内容。

#### 10.1.3 Question类

给出学习问题所涉及的提问，一个学习问题至少要包括一个Question，一个Problem可以有多个Question。Question类包含的内容主要有：提问的文本、声音、图片、图形和动画。

每一个问题在构造时都会计算其等级。

public string GetQuestionLevel(Dictionary<string, double> levels)改进

#### 10.1.4 Answer类

对应Question的答案，包含的内容可以有：文本、声音、图片、图形和动画。Answer类有多个派生类，处理多种答案类型，

* TextAnswer纯文本
* DrawingAnswer画图(图形)
* ProceduralAnswer执行某个算法，比如求解某个方程，必须按照某种算法指导下完成。该类根据算法语义网中的算法步骤指导EquationTopicModule中的公式的求解步骤，即根据算法语义网中的步骤节点生成一个EquProceduralStep实例，利用该实例执行EquationTopicModule中Execute()函数。

#### 10.1.5 QAPair类

Question-Answer对。一个problem可以有多个QAPair。

### 10.2 PQAFactory类及其派生类

PQAFactory类族用于创建不同知识类型的学习课题的PQA实例。PQAFactory类是所有其它类的基类，包含一个最重要的抽象方法：

public abstract PQA CreateSpecificPQA(string topic);

在PQAFactory类的派生类不断被重写，针对不同知识类型的学习课题topic（在对应类型的TopicModule派生类基础上）创建一个PQA实例，其中包括了若干个Question及其答案。而PQA中所涉及的所有文本、声音、图片、图形和动画在对应的语义网中进行描述。基于语义连接的提问在PQAFactory类中。

#### 10.2.1 ConceptPQAFactory

设计了（1）概念内涵的填空题（2）区别概念题（3）近似概念题

#### 10.2.2 ConclusionPQAFactory

测试结论的内容填空

#### 10.2.3 DomainTopicPQAFactory

没设计问题答案对

#### 10.2.4 EquationPQAFactory

方程的知识模块与算法知识模块有关联，所以需要处理这两者之间的关系。在CreateSpecificPQA()中的提问主要包括：（1）公式或方程所需要的算法有哪些；（2）如果在课程的流程语义网中找到相应的算法语义图，则提问如何利用该算法进行方程或公式的求解。相应的求解答案用ProceduralAnswer类实例进行处理。

#### 10.2.5 ExperimentPQAFactory

设计的问题包括：（1）实验原理是什么（2）阐述实验方法或步骤（3）实验需要的器材有哪些（4）实验的目的是什么

#### 10.2.6 InstrumentPQAFactory

设计的问题包括：（1）图中显示的是什么仪器（2）某仪器是用来作什么的（3）使用某一仪器有什么注意事项

#### 10.2.7 PhenomenaPQAFactory

现象内容填空题

#### 10.2.8 ProceduralPQAFactory

设计的问题包括：（1）列出算法中的主要步骤（2）执行某一步需要什么条件

#### 10.2.9 UnitPQAFactory

设计的问题：（1）列出某知识点有哪些单位（2）写出单位的符号表示（3）其中哪个是国际单位（4）某单位是如何被定义的（5）计算单位转换

### 10.3其他杂类

## 第11章 Student Module

ITS的一个核心概念就是在没有真正老师的干预下，对不同学生进行因材施教，这也是系统所谓智能化的体现之处。学生模块就是用来实现ITS这些特性的功能模块，它连接系统与学生，描述学生信息及其学习进程。该模块包含的主要C#类如下,

### 11.1 Student类

ITS的基本学习和教学过程围绕学生展开，所以Student类是ITS系统的核心。该类主要包含的数据或对象有，LearningTopicRecord和LearningHistory什么区别

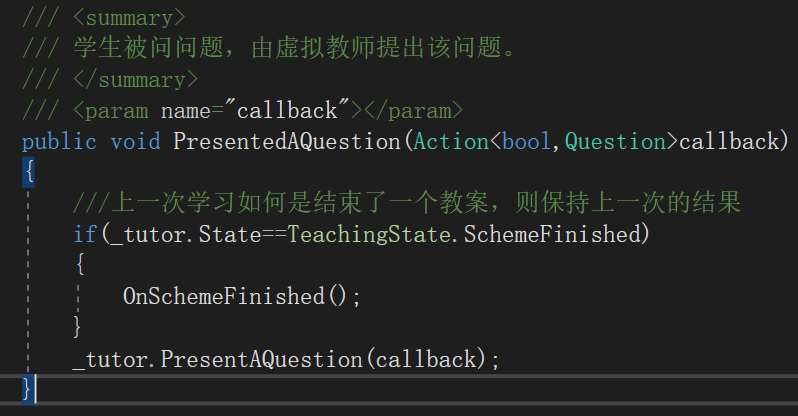
* VirtualTutor虚拟教师，用于制定教学策略，结合学生水平制定学习课题
* LearningHistory学习历史，记录学生的学习历史。反应全局状况。
* Ability当前的学习能力, 认知水平，包括综合能力(Comprehension ability)和解决问题技能(Problem-solving Skills)。综合能力主要体现学生掌握、理解概念的能力，而解决问题技能表示应用概念解决问题的能力。这些能力是动态变化的，每测试一次，这些值都发生变化.
* UserLog日志 内部包裹一个最重要的字段\_topics，键为课程，值为LearningTopicRecord类型的学习课题记录。

当系统启动时，Student也随之启动，用户向系统发出学习申请，虚拟教师根据学生的学习历史和能力，推荐学习课题。学生通过Student类的PresentedAQuestion()函数从虚拟教师那里得到一个学习问题。

Student类包括下面几个主要的函数，

* Start(Action<bool,Question>callback)，用户启动GUI，Student类调用该函数，该函数进一步启动虚拟教师，\_tutor.OnStart(callback)。
* ChangeLearningCourse(string course,Action<bool,Question>callback)，学生选择学习课程。此时，要保存当前的学习进程，要导入选择的课程的学习历史。同时发送消息给虚拟教师，启动新的课程教学。

最重要的一个函数就是PresentedAQuestion()，该函数通过虚拟教师得到一个学习问题。



### 11.2 LearningStragety

用于学生的学习决策，指导学生下一步学习计划。这里会使用到相当的AI 技术。目前，有用Fuzzy-state方法，贝叶斯方法等进行决策的。

### 11.3杂类详解

#### 11.3.1 KnowledgeTopicRecord类与LearningTopicRecord类

对应一个学习课题，都用于保存学习课程的记录，包括学习课题KnowledgeTopic或者LearningTopic和学习结果LearningResult、学习时间。

#### 11.3.2Section类与Chapter类

两个类都是记录学生的学习进度，不同在于：

Section包裹的\_topicDict字段的键为学习课题（topic），值为学习结果（LearningResult类，测试得分）。

Chapter包裹的\_sectionDict字段的键为小节名，值为对应的Section类型的实例。注意章名和小节名都应为全名，比如“第1章-机械运动”，“第1节-时间和长度”。

## 第12章 TutorModule

TutorModule用于控制和执行系统的辅导过程。该模块是ITS的核心模块，一是该模块在其它模块中起到连接作用，二是提供教学策略。Tutoring模块与Student模块一起形成了ITS“如何教？”的问题。具体而言， FVCTutor系统的辅导工作可以分成下面几个小部分：

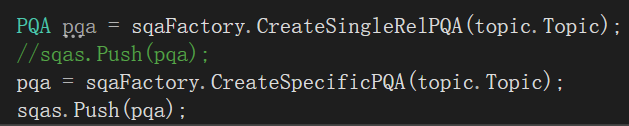
### 12.1 Tutoring类

一门课程是由许多的学习主题topic构成的，这些主题按照一定的循序排列，构成一个教学计划。ITS必须能够根据不同学生制定不同的教学计划，所以ITS的TutorModule模块有一个Tutoring类来执行这部分的工作。为每个学生选择最合适的学习主题topic，即为制定教学计划。在Tutoring模块所需要执行的所有任务中，其中之一是制定教学计划，这个计划必须是因学生而定。

在Tutoring类中的MakeASchemem()函数，根据选中的学习主题，从Domain模块中的学习材料仓库中选择最合适的学习内容和学习问题。

### 12.2 TeachingScheme

TeachingScheme类描述一个教学计划，称为教案。一个学习课题topic对应一个教案。一个学习课题可以有多个学习问题。一次只是呈现一个学习问题给用户，学生解答一个问题之后，自动进行到下一个问题，直到该教案中的所有问题都完成。教案中的学习问题封装在PQA类当中，P表示学习问题，Q表示问题的提问，A表示Q的答案。我们开发的系统针对多种不同知识类型开发了相应的PQA工厂类，用于产生特定的PQA。针对通用的语义进行提问也是知识学习的一部分，目前对这部分的处理还不够完善，需要进一步处理。下面的代码段是Tutoring中的MakeAScheme()函数中的一部分，我们暂时忽略了一般的语义连接所产生的提问。



### 12.3 MaterialSelection

学生的学习主题确定之后，要给定学习材料study materials。学习材料的给定也是因人而异，要做到个性化。

### 12.4 ResultAnalyzer

* 收集和分析从学生Student模块而来的反馈数据，如果反馈数据显示学生学习过程中的不足和错误(drawbacks)，制定如何改进的计划。信息反馈也是在该模块完成。ResultAnalyzer模块

## 第13章 ClientSystem

当用户启动一个用户GUI时，生成一个ClientSystem实例。ClientSystem是ITSEngine提供的与图形用户界面GUI的中间层，通过它将ITS系统与GUI连接起来。ClientSystem类作为ITS系统和用户之间的桥梁，生成一个虚拟学生Student类实例，通过Student类完成所有的工作，系统所有的工作是围绕学生而展开的。

### 13.1 OnStart(Action<string> callback)

当启动用户启动程序，启用GUI窗口时，调用该函数，该函数完成的工作包括：

* 启动虚拟学生Student。

### 13.2 OnChooseCourse (string course,Action<string> callback)

当用户选择不同的学习课程时，调用该函数。比如，学生开始选择的时“初二物理上”，现在改成学习“初二数学”。或者，系统推荐学习课题是初二物理，现在用户想学习其他课程，都会调用该函数。

该函数调用Student类的OnCourseChanged()函数实现相关功能。

### 13.3 OnStartAQuestion (Action<bool,string> callback)

调用该函数得到一个新的学习问题，该函数在用户在客户端的GUI上点击“开始”按钮或点击“Shift+Return”按钮时调用。

该函数调用Student类的PresentedAQuestion()函数完成相关工作。

### 13.4 OnAnswerSubmitted (string answer,Action<string> callback)

当用户提交学习问题的解答的时候调用该函数，向用户显示解答结果，在GUI的ObtainAndSubmitAnswer ()函数中调用。

该函数通过调用Student类的Feedbacked()函数实现相关功能。

### 13.5 OnObtainCorrects (Action<string> callback)

反馈学习问题的正确答案，通过调用Student类的ObtainCorrects()函数完成相关功能。该函数在GUI的PresentCorrects()函数中调用。

## 第14章 FVCTutorGUI

FVCTurtorGUI是学习系统的客户端应用程序，提供窗口系统实现学习问题、答案、反馈等文字、图片、公式等信息的显示和声音提示，实现用户与系统的交互。

### 14.1 主窗口

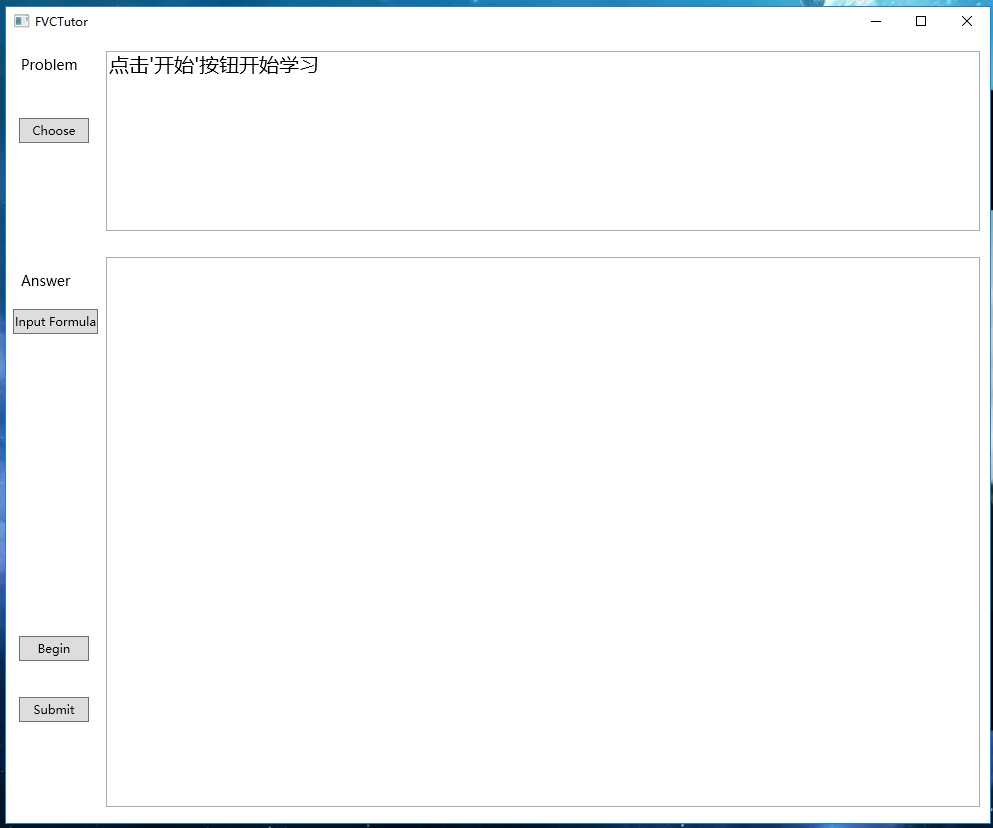


图3.1 主窗口

点击Choose按钮选择学习的文件，该文件包含了一个课本的知识。

人机交互步骤顺序如下：

（1）学生启动客户端程序，显示主窗口，如图3.1。这过程中，建立与服务器的连接，在本地实例化一个学生模块对象和一个辅导模块对象。

（2）通过学生模块，获取学生的学习偏好、学习历史和学习成绩。

（3）如果没有学习历史，学生点击Choose按钮选择自己的学习科目Course。

（4）如果有学习记录，但对应的学习科目已经结束，学生点击Choose按钮选择另外的学习科目。由辅导模块推荐一个学习主题及其对应的学习问题。

（4）暂时实现本地文件的读取，将来将文件放在服务器端。

## 第15章 FVCTutor的扩充

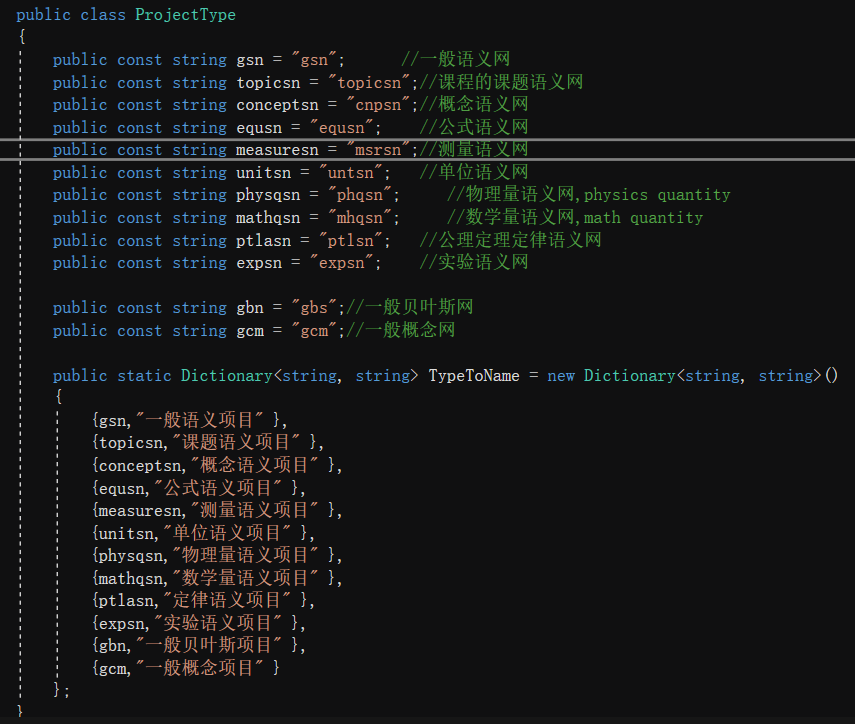
目前，大致完成了FVCTutor的基本框架，后续工作主要包括：

（1）制作课程语义网；

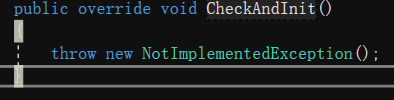
（2）扩展相应的课程工厂；

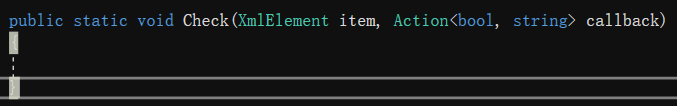
主要有如下几个步骤：

* 为此，先要分析该课程的知识构成，将知识进行分类，并将相应的类型名添加到ProjectType中，



* 从KRModuleSNet类派生，新增知识类型语义网类。参见UnitKRModuleSNet类。特别注意添加下面的两个函数，





* 从KRModule类派生，新增知识类型模块。参见UnitKRModule类。
* 从SQAFactory类派生，扩充这类知识的问题答案工厂类， 参见UnitSQAFactory。
* 从CourseFactory类派生新的课程类，参见PhysicsFactory。

以上步骤中，最核心的是新增知识类型语义网类，并进行该语义网的语法设计和解析。