LD语法检查和逻辑转换是一个连续执行的流程，主要包括两个步骤：

1. LD程序语法检查：检查变量关联的文本信息是否合法，检查变量类型是否合法；
2. 逻辑转换：主要完成将用户的程序逻辑转换成AT自定义的中间代码的树结构

在语法检查阶段，还将生成一些中间变量，这些中间变量将在生成中间代码树的时候应用到。

错误信息显示在工程的输出窗口，显示的信息包括：错误索引号、错误类型、错误的详细位置和错误的描述信息。

**LD语法检查**

LD语法检查完成的主要功能是：根据LD图形语言规则，检查用户编写的LD程序是否符合语法要求；如果用户编写的LD程序存在错误，则提示用户错误出现的位置和错误原因。

语法检查的总体实现流程



**触点语法检查**

如果节点是触点类型，则直接进行语法检查

触点元件进行语法检查的内容如下：

1. 触点元件文本不能为空；
2. 触点元件文本不能为关键字；
3. 触点元件文本字符串必须是合法字符串的集合；
4. 触点元件文本必须是个已定义常量或者变量；
5. 触点元件文本关联的必须是一个BOOL类型变量。

BOOL CLDContact：：TextAnalyzeAndCheck（）；

触点的语法检查实现流程为：



**块元件语法检查**

块元件进行语法检查的内容如下：

1. 如果是功能块类型，元件所带功能块实例文本须为对应功能块类型变量名；
2. 块元件文本必须为一个存在块类型名称；
3. 输入量类型必须与块元件引脚类型相匹配；
4. 块输出引脚与输入量相匹配；

BOOL CLDBox：：TextAnalyzeAndCheck（）；



**线圈语法检查**

线圈元件进行语法检查的内容如下：

1. 线圈元件文本不能为空；
2. 线圈元件文本不能为关键字；
3. 线圈元件文本字符串必须是合法字符串的集合；
4. 线圈元件文本字符数必须在限制数内；
5. 线圈元件文本必须是一个已定义变量或者常量；
6. 线圈元件文本关联必须是一个BOOL类型变量。

线圈语法检查的实现流程图如下：



**跳转语法检查：**

跳转语法检查的实现流程图如下：

1. 跳转名不能为空
2. 跳转名为有效的跳转名



引脚的语法检查：打印预览是调用的VC里面的系统函数，VC6.0自带有中英文的打印预览资源，所以显示正确。但是没有俄文资源，我们这边能做到的就是总控和AT保持一致，打印预览那块俄文调用英文的资源显示出英文的信息

引脚包括输入引脚和输出引脚，调用的都是各自类的TextAnalyzeAndCheck（），