**CDT基本框架**

数据流可视化工具在界面上分为以下几个模块，分别是代码编辑部分，工程视图部分，工具视图部分，菜单视图部分以及结果及错误显示视图。在功能上分为流语言文法及错误分析模块，性能分析模块，SDF流图显示模块，参数设置模块，流语言工程支撑模块。插件编程工具的基本框架如下图所示。按照以上分类：代码编辑框负责流程序代码的打开，编辑，在该部分还会给出一些传统的流程序运行过程在编译阶段才会给出的错误提示。编译部分集成了流程序编译器和gcc预处理器，在代码编辑框中编写的流程序代码会先经过gcc预处理器的预处理之后在进行编译，这是因为在流程序中支持部分C语言的库函数，这些库函数的定义需要用到C语言的头文件，因此需要调用gcc预处理器将头文件展开。

在编译阶段过后，程序员可以调用Graphviz脚本绘图工具将编译阶段生成dot脚本语言绘制成静态同步数据流图(SDF)，SDF给出了整个程序的结构以及任务划分和调度之后的actor分配情况，有便于程序员可视化的验证程序的正确性和划分算法的性能。编译阶段生成的C++代码要得到输出结果需要在不同的后端执行，由于流程序编译器支持生成针对X86、GPU与集群这三种后端的代码，因此，对于生成后的代码执行时，需要对不同的后端设定不同的执行方案，例如集群后端的代码需要把生成代码先复制到各个服务器上，然后运行执行脚本文件。

对于流程序生成代码的调试与普通的C++代码不同，流程序的生成代码一般都是多线程的代码，所以针对流程序的代码的调试需要用到gdb调试器的多线程调试部分，然后执行生成的可执行文件得到结果，在这一过程中会将错误或结果输出到错误及结果显示框内。由于流程序的性能是衡量数据流优势的关键，流程序性能分析工具就显得格外重要，在流程序代码执行完成程序员可以调用性能分析工具对流程序进行性能分析，性能分析工具可以模拟单核执行，用来得到程序多线程执行的加速比，得到流程序在核间、集群节点间、GPU与CPU间的通信时间以及actor的计算时间，以此来分析流程序执行时的计算、通信以及同步开销。

