Atitit。 工作流引擎的发展趋势

[1.1. 图灵完备 1](#_Toc23412)

[1.2. 图形化与文本化 1](#_Toc15884)

[1.3. Jit 1](#_Toc21672)

[1.4. Dsl化 2](#_Toc15320)

[1.5. Oo 2](#_Toc4780)

[1.6. 托管与本地代码的互操作 2](#_Toc186)

[1.7. 大型标准库 2](#_Toc22499)

[1.8. 异常处理机制 2](#_Toc17310)

[2. 参考 2](#_Toc27964)

## 图灵完备

现有工作流引擎 比如jbpm 的最大问题是不是图灵完备的，不能作为一个通用图形化编程语言来实现所有系统。

 简单判定图灵完备的方法就是看该语言能否模拟出图灵机  
  
图灵不完备的语言常见原因有循环或递归受限(无法写不终止的程序,如 while(true){}; ), 无法实现类似数组或列表这样的数据结构(不能模拟纸带). 这会使能写的程序有限

图灵不完备也不是没有意义, 有些场景我们需要限制语言本身. 如限制循环和递归, 可以保证该语言能写的程序一定是终止的.

作者:: 绰号:老哇的爪子 （ 全名：：Attilax Akbar Al Rapanui 阿提拉克斯 阿克巴 阿尔 拉帕努伊 ） 汉字名：艾龙，  EMAIL:1466519819@qq.com

转载请注明来源： http://blog.csdn.net/attilax

## 图形化与文本化

文本化面向专业人士

工作流将会分为俩个部分，第一部分是图形化编程，适合于非专业人员。。

第二部分是基于文本源码的DSL编程，适合于专业编程人员。。

一个程序的静态代码，动态线程模型，内存模型等等都是远远高于三维的（参见《人月神话》）。如果硬要往图形 UI 上靠，恐怕只有语法树能够图形化。而把语法树图形化实在是得不偿失，不但没有图形化本质的东西，丢掉了文本易于被键盘操作的能力

## Jit

解决性能问题的大法宝

## Dsl化

## Oo

## [托管与本地代码的互操作](http://blog.csdn.net/attilax/article/details/50655470" \l "t4)

## 大型标准库

## 异常处理机制

# 参考

Jvm clr的发展趋势