1. 目的

本文档目的在于解释分析BHO各种常见行为，进而构建一个BHO框架。

1. 分析：

BHO全程浏览器帮助对象，支持在IE浏览器端对浏览器动作及页面内容作修改和定制。一个完整的BHO经常可以由多个流程组成。每个流程是由一个或者多个页面组成，页面之间涉及到参数传递、通信等动作，每个流程完成一个事务逻辑，即每个流程具有事务性。一个BHO系统可以包含多个流程。除了与页面以及浏览器交互之外，整个BHO系统往往需要和传统的Win平台交互，如文件操作，网络通信操作，以及窗口消息操作等。

BHO由于网络及IE浏览器的不稳定的特点，需要有一定的failure recovery能力。BHO需要能够及时感知IE浏览器的各种操作失败与错误，如IE浏览器失去响应导致的操作失败，流程中某个页面逻辑错误导致的操作错误等。

对于每一种错误，框架都应该支持其恢复操作，或者采用弹性的架构方式，如对于Process A（流程A）加入了超时恢复操作之后还应该是Process A。

对于BHO范围外的内容，如网络通信，界面交互，框架应该提供一种组件式的可配置的方式，在该种方式下，使用框架开发的BHO系统应当可以静态或者动态地添加网络通信、界面交互等模块。

为了界定框架的功能边界，第一版框架的功能制定如下：

1. 框架支持多个Process，每个Process支持多个ProcessStep，每个ProcessStep可以重新执行的BHO模型；

2. 错误恢复方面，框架支持超时（包括ProcessStep级和Process级）、Process过程中逻辑错误的检测与恢复；

3. 框架支持方面包括网络通信组件、界面交互组件；

1. 下面首先对Process设计进行讨论。
   1. 基本定义：
2. ProcessStep：BHO所处理的一个页面（该页面不严格对应到一张网页，如当某项功能需要对某个网站的所有页面做Title替换，则该网站所有的网页只对应到一张BHO ProcessStep，即BHO ProcessStep是一种功能的划分，是Process中的一步功能）。每个ProcessStep拥有唯一的ID，该ID可以为网址，可以为网页中的某个特征，但是一定是这个ProcessStep即这个功能的一个唯一标示。

每个ProcessStep载入会触发该ProcessStep所重载的一些事件，每个ProcessStep在其所关注的事件回调函数中执行计算。ProcessStep之间的参数可以查询框架内置的key-value table得到。（如何处理逻辑错误？？如何处理ProcessStep级超时？？）

1. Process：由多个ProcessStep组成的一个完整的功能模块，每个Process具有事务性。若Process中某个ProcessStep执行失败，框架负责逐级回退，重新执行？那些不可重新执行的操作呢？如付款操作？？框架需要对ProcessStep分类处理，一类为不可重新执行的ProcessStep，叫做NonRetryProcessStep，对于NonRetryProcessStep，其执行结果是非常重要的，因而，需要将执行结果的页面保存下来。NonRetryProcessStep暂时不在第一版框架中支持。NonRetryProcessStep不包含所有的Post操作的网页，只包含result-critical的Post网页，因而是Post操作页面的一个子集。这里回退操作不仅仅只包含将页面冲定向到之前的页面，同时还包含将BHO系统状态恢复到之前页面的状态，即页面地址只是BHO系统状态的一部分。每个ProcessStep被调用之前，都需要将当前BHO系统的状态压入状态栈，如果没有注册状态，则在页面失败回退时候该ProcessStep会被跳过。
2. 技术实现：

当前世界上的浏览器可分为如下几大类：IE, Chrome/Safari (Webkit), Opera，由于本框架专注于页面的自动化解析与数据获取，因而在此我们不多考虑用户对浏览器的喜好，我们认为对浏览器的选择只需要从对业务系统的页面是否兼容触发，而由于多数企事业单位的业务系统在2000年左右的阶段开发，其页面支持IE居多；然纵观浏览器的发展，Chrome/Safari市场份额不断上升已成为主流浏览器，因而本框架开发之初打算支持的插件模型为IE/Webkit，因而对于Opera我们暂时不打算支持。

在编程语言上，我们选择C++作为开发语言；在面向的发布平台上我们支持Windows/Linux，但是由于我们精力有限暂时面向的平台为Windows。

1. 设计