本篇论文讲述了系统工程随着需要不断被修改而引发的不便，以及对逆向工程程序的理解与逆向工程技术。下面我将从九个方面做出对应的阅读报告。

一、背景。在工程中，程序都是存在或多或少的漏洞的，但一般对于它被开发出来时的需求而言，其漏洞与不足基本在当时都是可以忽略的，这样就导致许多应用广泛的程序都存在问题。随着时间流逝，程序相关的文档与说明出现了不同程度的丢失，因此出现了后来的人无法完全理解遗留下来的代码的现象，而后人在这样的情况下为了满足新的需求对代码进行修改与维护，导致程序在修改过程中与原始规范的差距越来越大，加深了无法完全理解前人代码的问题。也正是由此出现了逆向工程技术，来满足人们对于分析程序代码作用以及从中获取信息的需求。

二、正向工程。有两种方法可以减小程序和原始规范之间的差距。第一种是在规范和中间代码之间引入系统的中间描述，起到一个过渡连接的作用。但个人认为中间描述和说明文档同样存在容易丢失的问题，并没有从根源上解决问题。第二种是在基于给出明确规范的情况下，在系统开发过程中对其进行形式化的转换，保证它是符合规范的，由此解决遗留代码难以理解的问题。个人认为第二种方法比第一种要好，但如何普及统一的规范是其存在的较大的问题。

三、逆向工程。为了解决当下已经存在的程序难以理解的问题，逆向工程的存在是必要的。与正向工程的第二种方法所对应的，可以通过逆向一点点抽象代码，并以规范的格式来实现，由此做到复现（或许用词不太准确）现有的程序。

四、代码逆向工程。在解决实际问题的过程中，逆向的对象其实主要是代码，而为了有效解决技术问题，应当提高逆向工具的自动化特性，由此提高逆向工程的成熟度。

五、数据逆向工程。相对于代码逆向工程来说，数据逆向工程更加不具有代表性。

六、逆向工程工具。可以得出通过计算机辅助来获得遗留程序的决策与设计等信息是最可靠的，并且可以将逆向工程师从繁琐的劳动中解放出来。

七、逆向工程工具的有效性。逆向工具的普及性并不高，由于大多数程序员缺乏正确使用逆向工具的必要技能与知识经验，因此即使给这些程序员最好的工具也不会有太大帮助。由于大多数逆向工程工具都试图创建一个完全集成的环境，因此在这种假定下，逆向工程工具很难被直接集成到学术研究和工业常用的工具集中。

八、评估逆向工程工具。无论逆向工程师对程序的理解有多少，总是认为更有效的工具可以减少理解正在被维护的程序所需要的时间

九、结论。逆向工程技术不应关注对程序的理解，而应关注对系统的维护。最终应开发出用户可编程的工具。逆向工程开发仍有很长的路要走，控制成本和风险也是需要考虑的策略之一。