**手机屏幕截图

描述已自动生成**

上图是解决需求更改的阶段，即MDA和敏捷方法的需求变化。

MDA 建模涵盖的项目时间比敏捷开发中多，因为生成了代码。 所有问题都必须解决，因此所有需求更改都必须解决，直到此阶段。 另一方面，在敏捷方法编码中，编码涵盖了最多的项目时间，因此，大多数（或全部）需求更改，在此阶段得到解决。

解决模型变化的优点是：

1. 在发生更大变化时花费时间。如果更改是基本的，程序员可以非常快速地解决它们。但是，如果更改更重要或影响其他部分，则最好在项目生命周期的更快阶段解决它们，因为没有完成太多工作。
2. 对其他项目部件有影响的变化。 某些更改可能还需要更改项目的其他部分，在这种情况下，最好在模型中解决这些问题，在这些模型中可以对这些依赖性进行建模和更好地识别。
3. 记录更改。 如果在模型中描述并解决了更改，则可以保留更改历史记录。 如果更改仅在代码中求解，则很少保留有关每个更改的详细历史记录。 程序员通常快速继续解决以下任务，以便节省时间，而不是详细记录更改。

解决模型中更改的缺点：

1. 可以看到客户的变化。 如果更改直接在代码中完成，并且在使用系统时可以直接看到，则客户可以更好地指定此更改是否已完成。 客户可以更好地讨论他需要什么，以及完成的工作是否良好。

2. 基本变化时所花费的时间。 如果更改不是那么大，那么如果只有程序员解决它，而不是将开发团队中更多人员之间的更改传达给其他用户，并且一次又一次地向其他成员描述更改，它总是会更快。

前面描述的关于比较方法的观点基于差异：建模量和生命周期中解决需求变化的位置。 MDA方法还有一个更重要的事实。 此方法需要兼容的 CASE 工具，该工具可以从模型中生成质量良好的代码。 当代码在生命周期结束时生成时，当完成大量项目工作（非常详细的模型），如果生成时出现问题，并且必须重新创建代码，则可以定义风险，成本增长非常快。 如果开发团队使用经验证的 CASE 系统并非常了解其功能，此风险可能会非常低。 此风险取决于开发人员团队的经验，而不是所选方法，因此，在比较这两种方法的情况下，则不相关。 这是非常重要的，因此必须提及。