需求审查清单

161220135-吴德亚

在原来需求文档的基础上对每一个需求加上审查清单，在不符合某个特性的情况下给出**补充说明**。

(1) 需求是否完整？ (2) 需求是否一致？

(3) 需求是否可理解？ (4) 需求是否明确？

(5) 需求是否可实现？ (6) 需求是否可跟踪？

(7) 需求是否易于修改？ (8) 需求规格说明文档是否完整？

1. **功能需求**
2. **快捷键：**Vs 2019 的快捷键需要符合用户使用习惯，最好和其他主流IDE类似。

* **需求明确性： 比如，用户量广泛的jetBrains系列IDE，快捷键包括：自动对齐，查找删除等**

1. **编译器版本：**支持用户自主导入不同版本编译器

* **需求明确，容易实现**

1. **自动补全：**自动补全尽量人性化，尽量做到需要补全时给出待补全内容，但是不会错误理解用户意图（如：用户只是修改某一个部分，但是触发了自动补全，导致用户需要手动取消）

* **需求可理解，明确**。

1. **内存分析：**Vs新增加的内存分析功能可以很好的帮助解决和修复应用中的内存问题。

* **需求明确性：**应用场景可以是，对内存要求较高的应用程序在开发运行调试时可以查看自己对内存的消耗。
* 可实现且可理解。

1. **测试阶段：**使用全面的测试工具编写高质量代码，如：实施单元测试或使用测试资源管理器将自动化与测试用例工作项关联

* **需求明确性：**便于导入测试库（例如java中的junit模块），且对测试代码的支持性较高。

1. **命令行：对命令行支持效果较好，方便对命令行依赖程度较高的程序员，能通过命令行对程序进行编译运行，调试，更新软件，导入必要的库等功能**

* **需求明确性：可以和控制台相关联，方便在控制台、命令行、日志界面来回切换。控制台可以和windows或linux中的命令行界面相似。**

1. **自动保存：添加基本的保存功能，并允许用户设置自动保存时间等其他选项**

* **需求明确性：可以类比windows中microsoft软件的自动保存功能，避免突然断电、关机等行为。**

1. **文件对比：允许进行文件的纵向、横向对比（即不同历史文件对比，两份文件的差异化对比）**

* **需求明确性：考虑应用场景，两份代码执行结果不同但单凭人眼又看不出来差异（尤其是大批量的代码），就只能使用文件对比功能查找差异。**

1. **文件处理：保存单个文件修改步骤，允许回退到某一个时间点，允许删除文件时，仅仅将其从本项目中删除，但允许文件恢复，只有当删除本项目时才删除所有文件（不可逆）**

* **需求可理解性：此功能在IDEA中已经存在，专门针对手误删除文件的情况。且其内部实现加入了时间线的功能，即你可以恢复到一定时间之前。**

1. **引入Git：鉴于Git现在的流行，所以vs引入了“发布到Git服务”的特性，让开发者可以在vs团队服务、GitHub或一个私有版本库上发布一个新的项目**

* **需求明确性：现有的Git工具可以很好地支持软件开发，适合多人协作。同时用户可以使用git连接自己的github账户进行代码pull，push等操作。使编辑提交一体化，开发更流畅。**

1. **扩展：**允许自定义扩展，根据自身喜好进行自定义。此外，vs还提供了许多丰富，有价值的扩展。

* **需求可理解性：**类似于浏览器脚本，用户可以添加插件支持（编写，编译），并随时进行删改。
* **需求可实现性：**可以在用户信息一栏中隐式地保留已安装插件，当用户进行登录时动态加载。

1. **Office开发人员工具：**包含了附带最新功能和更新的Office开发人员工具

* **需求可理解性：**可以进行连接服务，与Office 365进行集成。方便进行Office项目的调试。

1. **多人协作：**使用版本控制、具备敏捷性且高效协作（如团队资源管理器中新增Git功能并增强了连接体验）

* **需求重复：与之前的关于Git的支持重复**

1. **系统需求**
2. **权限：**需要系统管理员权限才能进行安装，且需要安装 .NET framework。

* **需求可理解性：**安装vs IDE需要修改系统变量，管理员权限是必要的

1. **软硬件：** Visual Studio允许在windows上进行安装，硬件需要1.8GHZ或更快的处理器，2GB以上的运行内存，硬盘空间一般需要20GB~50GB。

* **需求可理解性：**vs由于集成了较多语言的支持，所以需要较大的磁盘空间和运行内存。但最终实现时可以考虑将vs拆分成支持单个语言的IDE。

1. **非功能需求**
2. **下载：**Vs 下载速度慢，需要手动修改特定的源，否则可能出现下载卡顿

* **需求完整性：**最终实现时可以给出推荐源（常用、稳定），并尽可能提高安装的自动化（用户只需要进行勾选和点击）

1. **安装卸载：**可能会出现无法卸载干净，以及因为没有卸载干净导致无法安装新版本或者版本回退的情况。例如：“安装向导”在您的机器上检测到Visual Studio 6.0产品潜在的版本冲突，可能是由于安装时下载了不必要的组件。

* **需求可实现性：**给出安装卸载引导程序，用户只需要类似安装的操作就可以完成卸载且安全无残留。
* **需求可理解性：**操作系统内不同软件可能使用了不同版本的vs组件，随意删除可能会导致原有程序运行异常也可能无法卸载干净，导致下一次安装失败。

1. **不单一：**由于没有将支持不同语言的功能单独拆分成新的IDE，导致vs能支持多种语言，使得vs有些杂乱

* **需求部分重叠：**与vs需要较大磁盘空间的原因重叠。但同时也会导致vs运行缓慢，多个语言的支持也会显得臃肿。

1. **冗余项：**下载时出现很多暂时用不到的其他语言组件，导致软件庞大

* **需求可实现：**在用户下载安装时进行默认配置，例如：用户选择某一种语言时，只给他安装该语言需要的组件。后续用户需要的情况下可以通过接口进行获取。

1. **界面设计：**界面设计扁平化，界面简洁，将同一类型的功能隐藏了一个下拉选项内部，但同时，不能让该选项包含过多内容。支持不同背景颜色修改，最好能支持自定义背景图片（透明化，尺寸等）

* **需求可理解性：**使界面简洁，避免较多接口暴露在界面上，支持用户自定义，可以对界面进行美化。
* **需求可实现：**界面扁平化设计已经形成了一个趋势，产品界面也是用户选择某产品的重要考量因素，具有高度可实现性。

1. **软件对比：**由于每次vs推出新特性时，工程人员都不免将其与Eclipse进行对比，由于vs存在占用磁盘空间和机器内存的缺点，所以微软最近加入了Eclipse基金，开始将二者结合起来，未来，开发人员可以在vs内部使用Eclipse，反之，亦然。

* **需求明确性：**我们需要参考借鉴已有的软件架构和功能，但同时也需要给出自己的创新之处。善于发现已有功能的不足和改进，对于提升软件的依赖度有极大的帮助。
* **需求实现性：**在用户删除软件或者间隔一定的时期，调研用户对该软件的满意度和反馈，其中提到与主流IDE的差别和优势。

**实验体会**

1. **首先是自己之前的需求调研是不完整的，在同一小组内部也有未调查到的需求。**
2. **发现自己之前需求调查时仅仅是提出需要实现什么功能，而对该功能的描述并不是很清楚，导致理解可能产生歧义（最好附上一个实例或者应用场景）。提出的功能需要考虑是否是可实现且易于修改的（最后编程实现环节是否存在重大挑战）。**
3. **功能之间产生了重叠且功能分类并不细致。**