2025

Requirement Engineering Report

SOFTWARE ENGINEERING

ALIREZA HASSANZADEH

MELI MAHARAT UNIVERSITY | Mohammad Ahmadzadeh

فهرست مطالب

❖ **مقدمه**

مهندسی نرم‌افزار یکی از حوزه‌های کلیدی علوم کامپیوتر است که بر توسعه، مدیریت و نگهداری نرم‌افزارهای کارآمد، پایدار و بهینه تمرکز دارد. در این تحقیق، به بررسی عناصر اصلی مهندسی نرم‌افزار، روش‌های توسعه و مقایسه بین آن‌ها پرداخته می‌شود تا بهترین رویکردها در شرایط مختلف مشخص شوند.

❖ **عناصر مورد نیاز برای مهندسی نرم‌افزار**

❖ **تحلیل نیازمندی‌ها**

تحلیل نیازمندی‌ها شامل جمع‌آوری و مستندسازی نیازهای کاربران، اهداف سیستم و محدودیت‌های فنی است. این بخش معمولاً شامل دو دسته نیازمندی‌های عملکردی و غیرعملکردی می‌شود.

❖ **طراحی نرم‌افزار**

پس از تحلیل نیازمندی‌ها، طراحی سیستم شامل معماری، طراحی پایگاه داده، رابط کاربری و اجزای نرم‌افزار صورت می‌گیرد. این مرحله شامل طراحی سطح بالا و سطح پایین است.

❖ **پیاده‌سازی**

در این مرحله، برنامه‌نویسی نرم‌افزار بر اساس طراحی انجام شده صورت می‌گیرد. توسعه‌دهندگان از زبان‌های برنامه‌نویسی مناسب و ابزارهای مختلف برای پیاده‌سازی سیستم استفاده می‌کنند.

❖ **تست و اعتبارسنجی**

پس از پیاده‌سازی، تست‌های نرم‌افزاری انجام می‌شود تا از صحت عملکرد سیستم اطمینان حاصل شود. تست‌ها شامل تست واحد، تست یکپارچه‌سازی، تست سیستم و تست پذیرش کاربر می‌شوند.

❖ نگهداری و توسعه

پس از انتشار نرم‌افزار، نگهداری و به‌روزرسانی آن برای اصلاح مشکلات، بهبود عملکرد و اضافه کردن ویژگی‌های جدید ضروری است.

❖ روش‌های مهندسی نرم‌افزار

❖ مدل آبشاری(Waterfall)

مدل آبشاری یک روش سنتی است که به‌صورت خطی مراحل تحلیل، طراحی، پیاده‌سازی، تست و نگهداری را دنبال می‌کند. این مدل مناسب پروژه‌های کوچک و دارای نیازمندی‌های ثابت است اما در برابر تغییرات انعطاف‌پذیری پایینی دارد.

❖ مدل چابک(Agile)

مدل چابک یک روش انعطاف‌پذیر است که بر تکرارهای کوتاه‌مدت توسعه، بازخورد مشتری و تحویل مداوم تأکید دارد. این مدل برای پروژه‌هایی که نیازمند تغییرات مکرر هستند، بسیار مناسب است.

❖ مدل مارپیچی(Spiral)

مدل مارپیچی ترکیبی از مدل آبشاری و تکراری است که بر مدیریت ریسک تأکید دارد. در این مدل، توسعه در چندین فاز تکرارشونده انجام می‌شود که در هر تکرار، نیازمندی‌ها بازبینی و بهبود داده می‌شوند.

❖ مدل توسعه افزایشی(Incremental)

در این روش، نرم‌افزار به‌صورت تدریجی و در چندین نسخه ارائه می‌شود که هر نسخه قابلیت‌های جدیدی به سیستم اضافه می‌کند. این روش باعث کاهش ریسک و بهبود بازخورد مشتری می‌شود.

❖ **DevOps**

DevOpsترکیبی از توسعه و عملیات است که همکاری بین تیم‌های توسعه و IT را افزایش می‌دهد و از ابزارهای خودکارسازی برای تسریع فرایندهای تحویل و استقرار نرم‌افزار استفاده می‌کند.

- **مقایسه روش‌های مهندسی نرم‌افزار**

❖ **کارایی و بهره‌وری**

- مدل چابک و DevOps بهره‌وری بالایی دارند زیرا امکان تحویل مداوم را فراهم می‌کنند.
- مدل آبشاری بهره‌وری کمتری دارد، زیرا نیازمند تکمیل هر مرحله قبل از ورود به مرحله بعدی است.

❖ **سرعت توسعه**

- مدل چابک و DevOps سریع‌تر از سایر روش‌ها هستند.
- مدل آبشاری و مارپیچی معمولاً کندتر عمل می‌کنند، زیرا به مستندسازی و تحلیل بیشتری نیاز دارند.

❖ **هزینه‌ها و منابع**

- مدل چابک و افزایشی هزینه‌های کمتری دارند زیرا توسعه به‌صورت تدریجی انجام می‌شود.
- مدل آبشاری و مارپیچی به دلیل نیاز به تحلیل و طراحی جامع، هزینه‌های بیشتری دارند.

❖ **انعطاف‌پذیری در برابر تغییرات**

- مدل چابک و DevOps بیشترین انعطاف‌پذیری را دارند.
- مدل آبشاری کمترین انعطاف را دارد، زیرا تغییرات در مراحل بعدی هزینه‌بر هستند.

▪ **چالش‌ها و راهکارهای مهندسی نرم‌افزار**

❖ **چالش‌های رایج در مهندسی نرم‌افزار**

- تغییرات مداوم نیازمندی‌ها
- مشکلات مربوط به کیفیت کد و امنیت
- مدیریت زمان و منابع

5.2بهترین راهکارها برای افزایش کیفیت نرم‌افزار

- استفاده از رویکردهای تست مداوم
- به‌کارگیری ابزارهای DevOps وCI/CD

- مستندسازی مناسب و توسعه مبتنی بر نیازهای کاربران

- **فناوری‌های نوین در مهندسی نرم‌افزار**
  - **هوش مصنوعی در توسعه نرم‌افزار**
  - **توسعه نرم‌افزارهای مبتنی بر بلاک‌چین**
  - **اینترنت اشیا (IoT) و تأثیر آن بر نرم‌افزار**

- **نتیجه‌گیری**

در این تحقیق، به بررسی روش‌های مختلف مهندسی نرم‌افزار و مقایسه آن‌ها پرداخته شد. روش چابک و DevOps مناسب پروژه‌هایی با نیازمندی‌های متغیر و سرعت بالای توسعه هستند، درحالی‌که مدل آبشاری برای پروژه‌هایی با نیازمندی‌های ثابت مناسب‌تر است. انتخاب روش به ماهیت پروژه، منابع موجود و نیازهای مشتری بستگی دارد.

- **منابع**

- Pressman, R. S., & Maxim, B. R. (2019). *Software Engineering: A Practitioner's Approach*. McGraw-Hill.
- Sommerville, I. (2015). *Software Engineering*. Pearson.
- IEEE Software Engineering Standards.
- منابع آنلاین مستندات Agile و DevOps.