**Methodologies**

**درس :** مهندسی نرم افزار

**نگارنده :** عبدالرحمن بیگناهی

**مدرس :** محمد احمدزاره

:Methodologiesمتدلوژی، شامل راهنماها و دستورالعمل هایی است که ترتیب و ارجاعات[[1]](#footnote-1) را بین فعالیت ها و فرایندهای توسعه نرم افزار پیشنهاد می دهد و ساختار ارتباطی و اطلاعاتی را بین اجزا، فعالیت ها و افراد بهینه می کند. متدولوژی کمک می کند که با رویکردی سیستماتیک[[2]](#footnote-2)، بین مشتری و تیم توسعه درک بهتری شکل بگیرد. تیم هماهنگی بهتری پیدا کند و با حداکثر سازی کارایی و مدیریت ریسک، بهترین محصول در مناسب ترین زمان به مشتری ارائه شود.

انتخاب متدولوژی برای تیم توسعه، امری حیاتی است. اما شرایط پروژه، منابع در اختیار، خصوصیات تیم توسعه و ویژگی های مشتری؛ تعیین می کند که کدام متد می تواند مناسب تر باشد. هر روش توانمندی ها و محدودیت هایی را ایجاد می کند که باید برای انتخاب مورد توجه قرار گیرد.

در این مقاله سعی داریم با چند متدولوژی اشنا شویم و انها را باهم دیگر مقایسه کنیم.

1. **:scrum** اسکرام یک روش سریع برای مدیریت پروژه‌هاست که در آن پروژه به بخش‌های کوتاه و قابل مدیریت به نام اسپرینت[[3]](#footnote-3) تقسیم می‌شود. تیم‌ها در هر اسپرینت به طور منظم پیشرفت‌ها و مشکلات خود را بررسی می‌کنند. این متودولوژی به اعضای تیم کمک می‌کند تا به طور مستمر و با انعطاف به تغییرات پاسخ دهند و بهبود یابند. پایان هر اسپرینت فرصتی برای ارزیابی و تنظیم برنامه‌هاست.

مزایا:

**انعطاف‌پذیری بالا:** اسکرام به تیم‌ها اجازه می‌دهد که به سرعت به تغییرات نیازها و اولویت‌ها پاسخ دهند.

**شفافیت بیشتر:** جلسات روزانه و بازبینی‌ها باعث شفافیت بیشتر در فرایندها و پیشرفت پروژه می‌شود.

**تحویل مداوم:** اسپرینت‌های کوتاه باعث می‌شود که محصول به‌طور مداوم و در زمان کوتاه‌تری تحویل داده شود.

معایب:

**نیاز به تعهد بالا:** اسکرام نیازمند تعهد بسیار بالای تیم و اعضا است که می‌تواند چالش‌برانگیز باشد.

**احتمال استرس:** به دلیل بازه‌های زمانی کوتاه اسپرینت، ممکن است اعضای تیم تحت فشار قرار بگیرند.

**مناسب نبودن برای پروژه‌های بزرگ و پیچیده:** مدیریت پروژه‌های بسیار بزرگ با اسکرام ممکن است نیازمند تقسیم‌بندی به تیم‌های کوچکتر و خودمختار باشد که چالش‌هایی را به همراه دارد.

1. **: (Extreme Programming) XP**روشی سریع برای توسعه نرم‌افزار است که کیفیت کد و بازخورد سریع[[4]](#footnote-4) را در اولویت قرار می‌دهد. در این روش، تکنیک‌هایی مانند جفت برنامه‌نویسی (دو نفر همزمان روی یک کد کار می‌کنند) و تست خودکار برای اطمینان از کیفیت کد استفاده می‌شود. تیم‌ها با مشتریان همکاری نزدیکی دارند و به‌طور مداوم بازخورد می‌گیرند. این کار کمک می‌کند تا نیازهای مشتری به سرعت شناسایی و برآورده شوند و همچنین بهبود مستمر در فرآیند توسعه حاصل شود. XP به تیم‌ها کمک می‌کند تا به سرعت به تغییرات پاسخ دهند و کیفیت نهایی نرم‌افزار را افزایش دهند.

مزایا:

کد با کیفیت بالا: استفاده از تکنیک‌هایی مثل جفت برنامه‌نویسی و بازبینی کد به بهبود کیفیت کد و کاهش باگ‌ها کمک می‌کند.

بازخورد سریع: تست واحد خودکار و بازبینی مداوم باعث دریافت سریع‌تر بازخورد و بهبود مستمر می‌شود.

مشارکت بالای مشتری: همکاری نزدیک با مشتری‌ها و دریافت بازخوردهای مداوم باعث می‌شود نیازهای آنها بهتر برآورده شود.

معایب:

نیاز به آموزش: تکنیک‌های XP مانند جفت برنامه‌نویسی و تست خودکار نیازمند آموزش و تجربه زیاد هستند.

نیاز به تعهد: این روش نیازمند تعهد بالا و مشارکت فعال تمامی اعضای تیم است.

مناسب نبودن برای پروژه‌های بزرگ: ممکن است پیاده‌سازی XP در پروژه‌های بزرگ و پیچیده چالش‌برانگیز باشد.

1. **:Canban**انبان یک متدولوژی سریع است که به‌جای استفاده از تکرارهای زمانی ثابت[[5]](#footnote-5) مانند اسکرام، بر مدیریت جریان کار تمرکز دارد. این روش از تخته کانبان استفاده می‌کند که شامل ستون‌هایی مثل «انجام نشده»، «در حال انجام» و «انجام شده» است. تیم‌ها می‌توانند با مشاهده وضعیت کارها و شناسایی گلوگاه‌ها[[6]](#footnote-6)، بهره‌وری را افزایش دهند و زمان اتلاف شده را کاهش دهند.

مزایا:

شفافیت بالا: تخته کانبان به‌طور دیداری جریان کارها را نشان می‌دهد.

کاهش اتلاف: با شناسایی و رفع گلوگاه‌ها، اتلاف زمان کاهش می‌یابد.

انعطاف‌پذیری: می‌تواند به‌راحتی با تغییرات نیازها سازگار شود.

معایب:

نیاز به نظارت مداوم: نیازمند نظارت و به‌روزرسانی مداوم است.

پیچیدگی مدیریت: ممکن است برای تیم‌های بزرگ و پروژه‌های پیچیده چالش‌برانگیز باشد.

محدودیت کارها: ممکن است محدودیت کارها در هر زمان باعث استرس اعضای تیم شود.

1. **:(Unified Process)UP**یک چارچوب توسعه نرم‌افزار است که شامل مراحلی مانند تحلیل، طراحی، پیاده‌سازی و تست است. هر مرحله به تکرارهای کوتاه تقسیم می‌شود تا ریسک‌ها کاهش یابد و کیفیت بهبود یابد. UP به تیم‌ها کمک می‌کند تا با یک رویکرد ساختارمند، پروژه‌ها را به صورت تدریجی و افزایشی توسعه دهند.

مزایا:

مدیریت ریسک: با استفاده از تکرارهای کوتاه، ریسک‌های پروژه به‌طور مؤثر مدیریت می‌شود.

کیفیت بالا: تمرکز بر تحلیل دقیق و طراحی قبل از پیاده‌سازی به بهبود کیفیت کمک می‌کند.

مناسب برای پروژه‌های بزرگ: ساختارمندی UP آن را برای پروژه‌های بزرگ و پیچیده مناسب می‌کند.

معایب:

نیاز به مستندسازی زیاد: این روش نیازمند مستندسازی جامع و دقیق است که می‌تواند زمان‌بر باشد.

کمبود انعطاف‌پذیری: در مقایسه با روش‌های سریع مانند اسکرام، UP انعطاف‌پذیری کمتری دارد.

پیاده‌سازی پیچیده: نیازمند مهارت‌ها و دانش فنی بالایی برای پیاده‌سازی موفق است.

1. **:AUP (Agile Unified Process)**نسخه سریع‌تر فرآیند یکپارچه (UP) است که تلاش می‌کند تا مزایای هر دو روش را ترکیب کند. AUP از تکرارهای کوتاه و بازبینی‌های مداوم استفاده می‌کند تا تیم‌ها به سرعت به تغییرات پاسخ دهند و در عین حال ساختار و مستندسازی UP را حفظ کنند. این متودولوژی شامل مراحلی مانند تحلیل نیازمندی‌ها، طراحی، پیاده‌سازی، و تست است که در دوره‌های تکراری اجرا می‌شوند. هدف AUP کاهش ریسک، بهبود کیفیت و افزایش انعطاف‌پذیری تیم‌های توسعه است. به دلیل ترکیب روش‌های چابک با فرآیند یکپارچه، این روش به تیم‌ها کمک می‌کند تا با بهینه‌سازی کارایی و مستندسازی، به بهترین نتایج دست یابند.

**مزایا:**

**انعطاف‌پذیری:** با استفاده از تکرارهای کوتاه و بازبینی‌های مداوم، تیم‌ها به سرعت به  تغییرات پاسخ می‌دهند.

**مدیریت ریسک** :مانند UP، تکرارهای کوتاه AUP به کاهش ریسک‌های پروژه کمک می‌کنند.

**توازن میان چابکی و ساختار:** ترکیب روش‌های چابک با UP باعث ایجاد توازن میان  انعطاف‌پذیری و مستندسازی می‌شود.

**معایب:**

**نیاز به دانش و تجربه:** پیاده‌سازی موفق AUP نیازمند مهارت‌ها و تجربه زیاد در زمینه  هر دو روش چابک و UP است.

**مستندسازی:** همچنان نیازمند مستندسازی جامع است که ممکن است زمان‌بر  باشد.

**پیچیدگی مدیریت:** به دلیل ترکیب روش‌ها، مدیریت AUP ممکن است برای برخی  تیم‌ ها چالش‌برانگیز باشد.

1. **:DSDM (Dynamic Systems Development Method)** یک متدولوژی چابک است که بر تحویل به‌موقع پروژه‌ها در چارچوب بودجه و زمان مشخص تمرکز دارد. این روش با همکاری نزدیک تیم با کاربران و مشتریان، تکرارهای کوتاه و بازخوردهای مداوم به بهبود کیفیت محصول کمک می‌کند. همچنین مستندسازی دقیق و منظم از مراحل پروژه اهمیت دارد. DSDM به تیم‌ها امکان می‌دهد که سریع به تغییرات پاسخ دهند و نیازهای مشتریان را برآورده کنند.

مزایا:

تحویل به موقع: تمرکز بر تحویل به موقع و در چارچوب بودجه مشخص باعث می‌شود پروژه‌ها بهتر مدیریت شوند.

کاربرمحور: همکاری نزدیک با کاربران و مشتریان به بهبود نیازمندی‌ها و رضایت آن‌ها کمک می‌کند.

انعطاف‌پذیری: تکرارهای کوتاه و بازخوردهای مداوم امکان تغییرات سریع و بهبود مستمر را فراهم می‌کند.

معایب:

نیاز به همکاری نزدیک: نیازمند تعامل و همکاری نزدیک تیم با کاربران و مشتریان است که ممکن است همیشه ممکن نباشد.

نیاز به آموزش: برای پیاده‌سازی موفق DSDM نیاز به آموزش و آشنایی کامل با این روش است.

مستندسازی: مانند بسیاری از متدولوژی‌های چابک دیگر، نیازمند مستندسازی منظم و دقیق است.

1. **:DevOps** یک فرهنگ کاری و متدولوژی است که همکاری نزدیک بین تیم‌های توسعه و عملیات را ترویج می‌دهد. این روش از اتوماسیون، یکپارچه‌سازی و تحویل مداوم استفاده می‌کند تا نرم‌افزار با کیفیت بالا و به‌سرعت تحویل داده شود. بازبینی‌های مداوم و تست‌های خودکار به بهبود کیفیت کمک می‌کنند. DevOps نیازمند تغییر فرهنگ کاری و استفاده از ابزارهای پیچیده است. هزینه‌های اولیه نیز ممکن است بالا باشد.

مزایا:

زمان تحویل سریع‌تر: اتوماسیون و CI/CD باعث می‌شود که نرم‌افزار با سرعت بیشتری تحویل داده شود.

کیفیت بالاتر: بازبینی‌های مداوم و تست‌های خودکار به بهبود کیفیت نرم‌افزار کمک می‌کنند.

همکاری بهتر: تعامل نزدیک بین تیم‌های توسعه و عملیات باعث کاهش تضادها و بهبود هماهنگی می‌شود.

معایب:

نیاز به تغییر فرهنگ: اجرای موفق DevOps نیازمند تغییر در فرهنگ کاری و همکاری بین تیم‌ها است.

پیچیدگی: نیاز به ابزارها و فناوری‌های پیچیده برای اتوماسیون و CI/CD.

هزینه‌های اولیه: اجرای DevOps ممکن است نیازمند سرمایه‌گذاری اولیه در ابزارها و آموزش باشد.

جمع بندی:

| **متودولوژی** | **شرح** | **مزایا** | **معایب** |
| --- | --- | --- | --- |
| Scrum | مدیریت پروژه با  اسپرینت‌های کوتاه و  جلسات منظم | انعطاف‌پذیری بالا، شفافیت بیشتر، تحویل مداوم | نیاز به تعهد بالا، استرس،  مناسب نبودن برای پروژه‌های  بزرگ |
| XP | تمرکز بر کیفیت کد و  بازخورد سریع | کد با کیفیت بالا، بازخورد سریع، مشارکت بالای مشتری | نیاز به آموزش، تعهد بالا،  مناسب نبودن برای پروژه‌های  بزرگ |
| Kanban | مدیریت جریان کار  با تخته کانبان | شفافیت بالا، کاهش اتلاف، انعطاف‌پذیری | نیاز به نظارت مداوم، پیچیدگی مدیریت، محدودیت کارها |
| UP | چارچوب توسعه با  مراحل مختلف | مدیریت ریسک، کیفیت  بالا، مناسب برای پروژه‌های بزرگ | مستندسازی زیاد، کمبود  انعطاف‌پذیری، پیاده‌سازی  پیچیده |
| AUP | نسخه چابک UP  با تکرارهای کوتاه | انعطاف‌پذیری، مدیریت  ریسک، توازن میان چابکی و ساختار | نیاز به دانش و تجربه،  مستندسازی، پیچیدگی  مدیریت |
| DSDM | تحویل به‌موقع پروژه‌ها با تکرارهای کوتاه | تحویل به‌موقع، کاربرمحور، انعطاف‌پذیری | نیاز به همکاری نزدیک، نیاز به  آموزش، مستندسازی |
| DevOps | همکاری بین تیم‌های توسعه و عملیات با  استفاده از اتوماسیون | زمان تحویل سریع‌تر،  کیفیت بالاتر، همکاری  بهتر | نیاز به تغییر فرهنگ،  پیچیدگی، هزینه‌های اولیه |

|  |  |  |
| --- | --- | --- |
| متودولوژی | کاربرد | مناسب برای پروژه‌ها |
| Scrum | مدیریت پروژه با تمرکز بر بازخورد سریع و تحویل مداوم | پروژه‌های کوچک تا متوسط که نیازمند تغییرات سریع هستند |
| XP | افزایش کیفیت کد با استفاده از تکنیک‌های برنامه‌نویسی چابک | پروژه‌های نرم‌افزاری که کیفیت کد و بازخورد سریع اهمیت بالایی دارند |
| Kanban | مدیریت جریان کار و افزایش بهره‌وری با استفاده از تخته کانبان | پروژه‌هایی با فرآیندهای مداوم و نیازمند بهبود مستمر |
| UP | چارچوب ساختارمند برای مدیریت مراحل مختلف توسعه | پروژه‌های بزرگ و پیچیده که نیاز به مدیریت دقیق ریسک و مستندسازی دارند |
| AUP | نسخه چابک UP با انعطاف بیشتر و تکرارهای کوتاه | پروژه‌هایی که نیاز به ترکیب روش‌های چابک و ساختارمند دارند |
| DSDM | تحویل به‌موقع پروژه‌ها با تمرکز بر همکاری نزدیک و بازخوردهای مداوم | پروژه‌هایی که تحویل به‌موقع و تعامل مداوم با کاربران اهمیت دارند |
| DevOps | همکاری بین تیم‌های توسعه و عملیات برای تحویل سریع و با کیفیت نرم‌افزار | پروژه‌هایی که نیاز به اتوماسیون و تحویل مداوم نرم‌افزار دارند |

منابع:

1. [**scrum.org**](https://www.scrum.org/) - Scrum
2. [**extremeprogramming.org**](https://www.extremeprogramming.org/) - XP (Extreme Programming)
3. [**kanbanize.com**](https://kanbanize.com/kanban-resources/getting-started/what-is-kanban) - Kanban
4. [**rationalunifiedprocess.com**](https://rationalunifiedprocess.com/) - UP (Unified Process)
5. [**agilemodeling.com**](https://www.agilemodeling.com/essays/agileUnifiedProcess.htm) - AUP (Agile Unified Process)
6. [**dsdm.org**](https://www.agilebusiness.org/page/whatisdsdm) - DSDM (Dynamic Systems Development Method)
7. [**devops.com**](https://devops.com/) - DevOps

1. References [↑](#footnote-ref-1)
2. Systematic [↑](#footnote-ref-2)
3. sprint [↑](#footnote-ref-3)
4. Fast feedback [↑](#footnote-ref-4)
5. Constant time iterations [↑](#footnote-ref-5)
6. bottlenecks [↑](#footnote-ref-6)