RUP ، XP ،SCRUM

اشاره :

در حقیقت ساختن یك نرم‌افزار فقط نوشتن كدهای برنامه نیست. رویه ساخت نرم‌افزارها مراحل متعددی را دربرمی‌گیرد؛ از جمع آوری نیازهای كاربران گرفته تا طراحی، نوشتن كد و در آخر امتحان نرم افزار. روش تولید نرم‌افزارهای كوچك با نرم‌افزارهای بزرگ متفاوت است و طبعاً رویه تولید نرم‌افزارهای كوچك نیز متفاوت خواهد بود. البته این رویه نباید سنگین و حجیم باشد، باید مستقیماً به تمامی فعالیت‌های لازم برای تولید نرم‌افزاری با كیفیت بالا نظارت داشته باشد و از تمامی رویه‌های آسان و متمركز استفاده كند. با استفاده از تكنیك‌هایی مفید، از روش‌هایی مانند Scrum،XP و RUP می‌توان رویه‌ای مناسب برای تولید نرم‌افزارهای كوچك به‌وجود آورد. همچنین می‌توان از روش‌هایPSP و TSP نیز كه برای تولید نرم‌افزارهای كوچك مناسب هستند استفاده نمود و به‌وسیله این روش‌ها كیفیت و قابلیت‌های نرم‌افزارها را بالا برد و در حداقل زمان ممكن نرم‌افزار را تهیه نمود. این مقاله با بررسی روش‌های جدید و متداول امروزی در تولید نرم‌افزار، بهترین و مناسب‌ترین روش تولید نرم‌افزارهای كوچك را به شما نشان خواهد داد. گفتنی است نوشتار حاضر نتایج تحقیقات من در گروه تحقیقاتی مهندسی نرم‌افزار دانشگاه ساندرلند انگلستان است و آمار و نتیجه‌گیری‌های آن براساس مصاحبه‌های انجام شده با چندین شركت كوچك و بزرگ تولید نرم‌افزار آن كشور است.

فرایند تولید نرم‌افزار

پیروی از یك رویه منظم تولید نرم‌افزار به تولیدكنندگان نرم‌افزار كمك می‌كند امور مربوط به‌تولید نرم‌افزار را منظم و پروژه را در حداقل زمان ممكن و با كارایی بالایی انجام دهند. در حقیقت یك رویه یا Process از مراحل مختلفی تشكیل شده است. این مراحل فعالیت‌های مربوط به رویه را مشخص می‌نمایند و تعیین می‌كنند كه این فعالیت‌ها باید چگونه انجام شوند. پیروی از این مراحل به اعضای پروژه دریابند یاری می‌رساند كه چه كاری را چه موقع و چگونه انجام دهند همچنین این كار میان اعضای گروه نیز

هماهنگی به وجود میآورد. از آن جایی که منابع موجود و نیازهای کاربران هر نرمافزار با دیگری تفاوت دارد، فرایند تولید نرمافزارهای گوناگون نیز متفاوت است.

انجمن IEEE رویه یا فرایند تولید نرمافزار را این گونه تعریف می‌کند: رویه تولید نرمافزار در واقع شامل مراحلی مانند جمعآوری نیازهای کاربران ، طراحی سیستم با استفاده از تحلیل این نیازها و نوشتن کدهای نرمافزار با استفاده از طرح نرمافزار است. همچنین بر این‌باور است که از آن جایی که کیفیت و بهره‌وری نیروی کار با کیفیت روند تولید نرمافزار ارتباط مستقیم دارد، طراحی و مدیریت رویه تولید نرمافزار از اهمیت شایانی برخوردار است.

برای طراحی یك رویه تولید نرمافزار می توان از روش‌های متفاوتی استفاده نمود و از آن جایی که هر پروژه نرمافزاری با دیگر پروژه‌ها متفاوت است، می‌توان گفت که رویه تولید آن پروژه نیز با دیگر پروژه‌ها تفاوت دارد. در واقع می‌توان گفت: انتخاب این روش‌ها رابطه مستقیمی با اندازه گروه در پروژه دارد و نرمافزارهای بزرگ و کوچك نیاز به رویه‌های تولید متفاوت دارند.

در ادامه این مقاله روش‌های تولید نرمافزارها، به خصوص نرمافزارهای نسبتاً کوچك که از گروه‌های تولید نرمافزاری کوچك‌تری استفاده می‌کنند، بررسی می‌شوند و مورد ارزیابی قرار می‌گیرند.

روش SCRUM

در روش‌های قدیمی و معمول ساخت نرمافزار، طراحان نرمافزار معمولاً ابتدا فرض می‌کنند که تمامی نیازهای کاربران سیستم را درك کرده‌اند. اما همیشه نیازهای کاربران سیستم در ابتدا مشخص نیست و کاربران ممکن است در همان مراحل ابتدایی، نیازهای خود را تغییر دهند و این چیزی است که برنامه‌نویسان و طراحان سیستم همیشه از آن شکایت می‌کنند و به دنبال راه‌حلی برای رفع این موضوع می‌گردند. به‌عنوان مثال مدل قدیمی آبشاری (waterfall) را در نظر بگیرید.

این مدل حاوی مشکلات فراوانی است که به صورت مستقیم به غیرقابل انعطاف‌بودن این مدل ارتباط دارد. این مدل مانند یك جاده یك طرفه می‌باشد که وقتی اتومبیل در آن حرکت می‌کند، نمی‌تواند مسیر خود را تغییر دهد و در جهت دیگری حرکت کند. در ابتدای کار کاربر سیستم ممکن است نظراتی در مورد سیستم داشته باشد ولی نمی‌تواند ببیند که سیستم چگونه کار خواهد کرد و در نتیجه ممکن است وقتی که سیستم آماده شد، از ساختار و کارایی آن راضی نباشد و تقاضای تغییر در سیستم را بنماید. در نتیجه اگر بتوانیم کاربر را از ابتدا در جریان ساخت نرم‌افزار قرار دهیم، ممکن است که این مشکل حل شود؛ زیرا می‌تواند نظرات خود را در طول مدت ساخت و قبل از اتمام کار اعلام کنند و در نتیجه از نرم‌افزار تهیه شده راضی باشد.

امروزه یکی از روش‌های تولید نرم‌افزار که به خصوص برای پروژه‌های نرم‌افزاری کوچك مورد استفاده قرار می‌گیرد و توسط بسیاری از اساتید و صاحب‌نظران مورد تأیید قرار گرفته است، روش SCRUM است. با استفاده از این روش که روشی به اصطلاح (iterative تکراری یا چرخشی) می‌باشد، می‌توان نرم‌افزارهای کوچك را با کیفیت بالا تهیه نمود. در این روش که به روش هوشمند یا Agile نیز مشهور است، مدیریت قوی تولید نرم‌افزار وجود دارد که به برنامه‌نویسان اجازه می‌دهد با استفاده از آن در پروژه‌ها به سرعت نرم‌افزار موردنظر را تهیه نمایند. اسم Scrum در حقیقت از بازی راگبی گرفته شده است (در بازی راگبی Scrum تیمی متشکل از هشت نفر است که با همکاری بسیار نزدیك با یکدیگر بازی می‌کنند).

در این روش هر عضو از گروه موظف به درك وظیفه خود در پروژه است و باید یك هدف مشخص را در تمامی مراحل عملیاتی یا فازهای اجرایی دنبال کند. لازم به ذکر است که در Scrum هر فاز عملیاتی سیستم به Sprint مشهور است.

روش Scrum همانند پروسه‌های دارای مرحله برنامه‌ریزی مقدماتی یا Initial Planning است. در این

فاز اعضای تیم باید یك نقشه مقدماتی و یك معماری سیستم قابل تغییر به وجود آورند. بعد از این فاز یك سری از Sprintها به صورت مرتب و جزء جزء نرم‌افزار مورد نظر را به وجود می‌آورند. انجام دادن هر Sprint ممکن است از یك تا چهار هفته به طول بینجامد و مجموع این Sprintها نرم‌افزار کاملی را به‌وجود میآورند.

فهرست تکالیف در هر Sprint به Backlog مشهور است که تکالیف تیم عملیاتی در هر Sprint را مشخص می‌کند. این Backlog در هر Sprint بروز می‌شود و هر تکلیف براساس اهمیتی که دارد در فهرست تکالیف تعیین اولویت می‌گردد. هر فرد در گروه یك کار یا تکلیف خاص از این فهرست را به عهده می‌گیرد و موظف می‌شود تا شروع Sprint بعدی آن را به اتمام برساند. وقتی که یك Sprint شروع شد، دیگر انجام هیچ تغییری در تکالیف امکان ندارد و حتی درخواست‌کننده نرم‌افزار نیز حق تغییر یا درخواست نیاز دیگری در نرم‌افزار را نخواهد داشت.

البته درخواست‌کننده می‌تواند از قسمتی از نرم‌افزار که باید در هر مرحله تولید شود بکاهد، اما نمی‌تواند تاریخ تحویل آن قسمت را تغییردهد. شاید بتوان گفت که این کار باعث ایجاد نظم در گروه می‌شود و تاریخ تحویل نرم‌افزار به تعویق نخواهد افتاد. علاوه بر این، در طول هر Sprint گروه موظف است روزانه جلساتی جهت بررسی روند پیشرفت و قابلیت‌های نرم‌افزار داشته باشد که این نیز به هماهنگی بیشتر گروه کمك خواهد کرد. در این جلسات که معمولاً به صورت روزانه است، سه گروه می‌توانند شرکت کنند: گروه تهیه‌کننده نرم‌افزار، مدیریت، و درخواست‌کنندگان نرم‌افزار.

در طول این جلسات مسئول جلسه که اغلب مدیر پروژه است، از تمامی اعضای تیم سه سؤال می پرسد:

۱- مسئولیت شما (تکالیف) از جلسه قبلی تاکنون چه بوده است و آیا توانسته‌اید این تکالیف را به اتمام برسانید؟

۲- در طول این دوره به چه مشکلاتی برخورده‌اید؟

۳- بر طبق فهرست وظایف، مسئولیت شما از حالا تا جلسه بعدی چه خواهد بود؟

مدیر Scrum در حقیقت مسئولیت شناسایی تکالیف محوله به اعضا، بررسی روند تکمیلی ساخت نرم‌افزار، بررسی قابلیت‌های اعضای گروه و فعالیت در راستای کم کردن ریسك در پروژه را داراست.

اما چه تفاوتی بین Scrum و دیگر روش‌های تولید نرم‌افزار وجود دارد؟ در جواب این سؤال باید یادآورشد که در Scrum هر مرحله یا Sprint قسمتی از نرم‌افزار را آماده می کند. در این روش می توان پیشرفت در تولید نرم‌افزار را در هر مرحله به خوبی احساس نمود. علاوه بر این، گروه می‌تواند پس از اتمام هر Sprint تصمیم‌گیری‌کند که آیا می خواهد به کار روی پروژه ادامه دهد یا انجام پروژه مذکور غیرممکن است. روش Scrum وقتی می‌تواند بیشتر مفید باشد که در ابتدای پروژه نیازهای کاربران به صورت دقیق مشخص نباشد و یك گروه کوچك مسئول پروژه نرم افزاری باشد.

نتایج تحقیقاتی که اواخر سال ۲۰۰۵ روی چندین شرکت تولیدکننده نرم‌افزار در کشور انگلستان انجام دادم، نشان‌دهنده این بود که شرکت‌هایی که از Scrum استفاده کرده بودند با حدود چهارصددرصد افزایش در بهره‌وری کار مواجه بودند. البته این رقم در گروه‌های نرم‌افزاری مختلف متفاوت بود و می‌توان گفت عوامل انسانی از جمله مدیر پروژه نقش بسیار مهمی در افزایش یا کاهش راندمان پروژه ها دارند.

شاید این سؤال در ذهن شما به وجود آید که چرا Scrum ممکن است برای تولید نرم‌افزارهای کوچك راه حل خوبی باشد؟ در جواب می‌توان گفت، از آن جایی که در تیم‌های کوچك، اعضای گروه باید از تمامی مسائل پروژه آگاه باشند و در Scrum تمامی مراحل ساخت توسط تمامی اعضای گروه قابل

مشاهده است. لذا این روش می‌تواند روش مناسبی باشد.

معایب روش Scram

مزایای استفاده از Scrum بسیار است، اما این روش چند اشکال نیز دارد. از جمله:

۱- Scrum روش جدیدی است و با روش‌های مرسوم تفاوت‌های زیادی دارد.

۲- برخی از برنامه‌نویسان حرفه‌ای ممکن است از تکالیفی که مدیر Scrum به ایشان می‌دهد راضی نباشند و بخواهند روش قدیمی خود را اجرا نمایند و در صورت اجبار، در روند اجرای پروژه کارشکنی کرده و مشکل‌آفرینی کنند.

۳- از آنجا که مدیر Scrum هم از نظر کیفی و هم کمی باید پروژه را مدیریت کند، Scrum نیاز به مدیریت بسیار قدرتمند دارد.

٤- Scrum را می‌توان به عنوان روش تولید نرم‌افزار نام برد، اما این روش بیشتر روش مدیریت پروژه هوشمند خوبی است و نمی‌توان آن را به صورت منفرد استفاده نمود و می‌توان گفت برای حصول اطمینان از موفقیت پروژه‌های نرم‌افزاری (به خصوص در سطح کوچك) باید این روش را با روش‌های دیگر ادغام نمود. Scrum را از آن جهت می‌توان روش خوبی برشمرد که روشی تحقیقی براساس تخمین، اولویت‌بندی، عملکرد گروه و بررسی نتایج است که اگر به صورت صحیح مورد استفاده قرار گیرد و قبل از استفاده به صورت کامل آموزش داده شود، می‌تواند راندمان پروژه‌های نرم‌افزاری، به خصوص تولید نرم‌افزارهای کوچك را به صورت بسیار محسوسی بالا ببرد.

روش XP

اشتباه نکنید! منظور از روش اکس‌پی، ویندوزاکس‌پی نیست. اکس‌پی مخفف Extreme Programming یا برنامه‌نویسی سریع می‌باشد که مانند Scrum روشی هوشمند در تولید نرم‌افزار است. در اکس‌پی دو برنامه‌نویس کار را انجام می‌دهند و قبل از اتمام برنامه آن را چندین‌بار امتحان می کنند. اکس‌پی در حقیقت روشی از تولید نرم‌افزار است که براساس آسانی، ارتباط، واکنش و تصمیم‌گیری سریع استوار است. شکل ۲ اصول روش اکس‌پی را نشان می‌دهد.

در روش اکس‌پی اعضای گروه (که کاربر سیستم نیز عضوی از آن است) در ابتدا جلسه‌ای تشکیل می‌دهند و اولویت‌های پروژه را مشخص می‌کنند. این گروه ممکن است از چند برنامه‌نویس، امتحان‌کننده نرم‌افزار یا Tester و تحلیلگر سیستم تشکیل شود که با هم از ابتدا تا انتهای پروژه همکاری می‌کنند. معمولاً در اکس‌پی برنامه‌نویسان در گروه‌های دوتایی قرار می‌گیرند و وظیفه تکمیل قسمتی از برنامه را برعهده می‌گیرند و هر دوی این برنامه نویسان در مورد هر کدام از نیازهای کاربران با هم بحث می کنند و قدم به قدم کلاس های برنامه را آماده می‌کنند.

بدین ترتیب که در ابتدا کلاسی را به صورت ابتدایی و بدون هیچ طراحی اولیه به وجود می‌آورند و این کلاس را امتحان می‌کنند. در صورتی که این کلاس فاقد هر گونه اشکال باشد، کد اصلی برنامه را بر آن اساس می‌نویسند. وقتی یکی از برنامه‌نویسان مشغول نوشتن قسمتی از برنامه است، برنامه‌نویس دیگر وظیفه کنترل صحت این کدها را عهده‌دار است و در صورت مشاهده هر گونه اشکال، نویسنده کد را مطلع می‌کند.

مانند Scrum، در اکس‌پی نیز اعضای گروه می‌توانند روند تکمیلی تولید نرم‌افزار را مشاهده کنند و در جریان کار قرار گیرند.اکس‌پی روش مناسبی برای مدیریت پروژه‌های کوچك می‌باشد که از دو تا ده برنامه‌نویس تشکیل شده است. اگر چه اصولاً اکس‌پی هیچ رویه خاص و مراحل پیوسته‌ای را مشخص نکرده اما می توان گفت که اکس‌پی داری چهار مرحله اصلی می باشد:

الف: مرحله زمانبندی پروژه: در این مرحله اعضای گروه با توجه به اندازه نرم‌افزار و کارایی آن برنامه زمانبندی را با هم تنظیم می کنند.

ب: طراحی ابتدایی

ج: نوشتن کدهای برنامه

د: امتحان‌کردن کدهای نوشته شده

مطابق تحقیقاتی که توسط نویسنده انجام شد، مشخص گردید که اکس‌پی در پروژه‌های بزرگ با تعداد اعضای بالای ده نفر اصلاً موفق نخواهد بود و تنها می‌تواند برای پروژه‌های کوچك مفید باشد. دلیل آن را نیز می توان در طبیعت این روش دانست؛ زیرا مستندات چندانی برای نرم‌افزار وجود ندارد و فقط دو نفر یا حداکثر چهار نفر می‌توانند در مورد قسمتی از نرم‌افزار اطلاعاتی داشته باشند. همچنین نرم‌افزار تولیدشده توسط این روش هیچ‌گونه طراحی سازمان یافته‌ای ندارد که این موضوع می‌تواند برای مراحل پس از نصب یعنی تعمیرات و نگهداری سیستم باعث بروز مشکلاتی گردد.

از جمله مزایای اکس‌پی می‌توان به این نکته اشاره نمود که از آن جایی که یك برنامه‌نویس به صورت مستقیم کدهای برنامه را کنترل می کند، می‌توان گفت که کیفیت نرم‌افزار تولیدی بالا می‌رود. همچنین می‌توان گفت از آن جایی که دو برنامه‌نویس با هم کار می‌کنند، آموزش کمتری نیاز است و در نتیجه هزینه تولید نرم‌افزار پایین خواهد آمد. اما این روش مشکلات خاص خود را نیز دارد. مثلاً تصورکنید اگر در یك گروه، یك برنامه‌نویس تمایلی برای کار با برنامه نویس دیگری را نداشته باشد یا در یك روز یکی از دو عضو گروه غایب باشد یا ... در نتیجه چون نمی‌توان با یك برنامه‌نویس به ادامه کار پرداخت، اتمام پروژه با تأخیر مواجه خواهد شد.

طبق نتایج تحقیقات به عمل آمده، وقتی یك برنامه‌نویس در کدهای برنامه به دنبال اشکال می گردد، حداکثر می‌تواند ده تا پانزده‌درصد از اشکالات برنامه را پیدا کند. اما وقتی در روشی مثل اکس‌پی دو برنامه‌نویس با هم کار می کنند و یکی از این برنامه‌نویسان کدها را کنترل می‌کند، بیست تا چهل‌درصد از اشکالات ساختاری برنامه خود را نشان می‌دهد. اما با استفاده از روش‌های PSP و TSP که در ادامه این مقاله توضیح داده می‌شوند حتی می‌توان تا هشتاددرصد اشکالات برنامه (که رقم بسیار خوبی است) را قبل نهایی‌شدن برنامه شناسایی و رفع کرد.

روشRUP Rational Unified (Process)

در این بخش یکی از معروف‌ترین رویه‌های تولید نرم‌افزار که توسط شرکت آی‌بی‌ام طراحی گردیده‌است را معرفی می‌کنیم. این روش با نام RUP Rational (Unified Process) در بسیاری از پروژه‌های نرم‌افزاری به کار گرفته می‌شود.
در حقیقت هدف اصلی RUP مطمئن‌شدن از این موضوع مهم است که آیا نرم‌افزار تولیدشده نیازهای کاربرانش را به صورت کامل، با کیفیت بالا، در زمان معین و با بودجه مشخص برآورده کرده است یا خیر.

مطابق تحقیقات انجام شده، از آن جایی که RUP به تمامی اعضای تیم، اطلاعاتی مشترك می‌دهد و تمامی اعضای گروه با یك زبان مشترك با هم مرتبط هستند، بازده کاری گروه را بالا می‌برد.

RUP دارای سه جزء اصلی است. جزء اول از مجموع راه‌حل‌های خوب که در رویه می‌تواند مورد استفاده قرار گیرد تشکیل شده است. جزء دوم همان مراحل تهیه نرم‌افزار است و جزء آخر قسمت‌های تشکیل‌دهنده این رویه می باشد.

RUP شش راه‌حل خوب که می‌تواند در مراحل مختلف این رویه به ما کمك کند را معرفی کرده است. از آن جمله:

۱- استفاده از USE CASEها که می‌توانند در جمع‌آوری نیازهای کاربران مفید باشند.

۲- استفاده از معماری نرم‌افزار قابل تغییر (component reuse)

۳- استفاده از روش‌های تکمیلی و Iterative برای کنترل بهتر و آسان پروژه نرم‌افزاری

٤- استفاده از نمودارهای UML

٥- کنترل تغییرات در نرم‌افزار

٦- کنترل کیفیت نرم‌افزار با توجه به درخواست‌های اولیه کاربران

شکل ۳ رویه RUP را نمایش می‌دهد. همان‌طور که در این شکل مشخص شده است چرخه تولید نرم‌افزار به چهار قسمت اصلی تقسیم شده است:

الف: Inception phase یا مرحله آغازین:

در این مرحله اهداف پروژه مشخص شده و درخواست‌های اولیه کاربران تعریف می‌شود. از خروجی‌های این مرحله می‌توان به مدل اولیه Use Case، آزمون ریسك در پروژه و برنامه زمانبندی پروژه اشاره کرد.

ب: Elaboration phase یا مرحله مقدماتی:

در این مرحله نیازهای کاربران تحلیل و بررسی شده و راه‌حل کلی طراحی سیستم ترسیم می‌شود. از خروجی‌های این مرحله می‌توان از مدل کامل شده Case Use، فهرست نیازهای کامل کاربران و طرح کلی سیستم نام برد.

ج: Construction phase:

یا مرحله ساخت و توسعه: در این مرحله نرم‌افزار طراحی شده ساخته می‌شود و به اصطلاح، کد برنامه نوشته شده و قسمت‌های مرتبط به هم ارتباط داده می‌شوند. از خروجی این مرحله می‌توان از نرم‌افزار، راهنمای استفاده از نرم‌افزار و مستندات سیستم نام برد.

د: Transition phase یا مرحله تغییرات:

در این مرحله اگر نرم‌افزار به وجود آمده در مرحله ساخت دچار مشکل شود، مشکل رفع خواهد شد.

تمامی مراحل توسط خطوط عمودی از همدیگر جدا شده‌اند و هر مرحله با یك milestone یا نقطه مهم تمام می‌شود. روش RUP با استفاده از مدل‌های مختلف همچنین مشخص می‌کند چه کسی، چگونه و چه وقت چه کاری را انجام خواهد داد.

همان‌طور که در این قسمت ذکر شد، روش RUP روشی انعطاف پذیر، قابل تغییر و پیشرفته است که می‌تواند در صورت استفاده صحیح، باعث افزایش کارایی و کیفیت نرم‌افزار تولیدی گردد. اما آیا RUP می‌تواند رویه خوبی برای تولید نرم‌افزارهای کوچك باشد؟ در جواب باید گفت که RUP را طوری طراحی کرده‌اند که بتواند برای انواع پروژه‌های نرم‌افزاری در هر اندازه مفید باشد و از آن جایی که از ابزارهای خوبی مثل UML نیز استفاده می‌کند، UML) در گروه‌های کوچك که نرم‌افزارهای کوچك طراحی می‌کنند ابزار مدلی خوبی است) می‌تواند باعث همکاری و هماهنگی بیشتر گروه گردد.

اما همان‌طور که در ادامه این بحث خواهید دید، اگر بتوانیم رویه‌های ساده‌تر را با یکدیگر ادغام کنیم،

شاید بتوانیم راه حلی با کارایی بالاتری داشته باشیم.

روش های PSP و TSP

PSP یا Personal Software Process در حقیقت روش تولید نرم‌افزار نیست بلکه روشی است نوین که با ملزم نمودن اعضای گروه پروژه‌های نرم‌افزاری به رعایت اصولی مشخص و استفاده از فرم‌ها و تکالیفی مشخص به آن‌ها کمك می‌کند کارایی و بهره‌وری کاری خود را بالا ببرند. این روش همچنین حاوی تکنیك‌های خوبی برای کنترل، اندازه‌گیری و تشخیص اشکالات می‌باشد که می‌تواند به شخص (مثلاً برنامه‌نویس) کمك کند تا مثلاً با اندازه‌گیری نرم‌افزار، یادداشت میزان فعالیت روزانه و ساعات هدر رفته، و اشکالات به وجود آمده، مشکلات را حل کند و در نتیجه بهره‌وری خود را بالاتر ببرد. TSP یا Team Software Process مانند PSP است، ولی برای یك تیم طراحی شده و با طرح روش‌های منظم جهت کنترل و جمع‌آوری اطلاعات روزانه به اعضای تیم کمك می‌کند تا کارایی خود را بالا ببرند.

راه‌حل‌های پیشنهادی

تا این قسمت با برخی از روش‌های تولید نرم‌افزار آشنا شدیم. اگر دقت کنید تمامی این روش‌ها و رویه‌ها می‌توانستند برای تولید نرم‌افزارهای کوچك مورداستفاده قرار گیرند، اما در ادامه مقاله با چند روش جدید آشنا خواهید شد که در چندین گروه نرم‌افزاری کوچك مورد آزمایش قرار گرفته‌اند و در تمامی موارد بازدهی درخور داشته‌اند. در واقع نمی‌توان گفت تمامی روش‌های زیر روش‌های جدیدی هستند، بلکه برخی از آن‌ها از ادغام روش‌های بالا به وجود آمده‌اند.

روش RUP + Scrum

همان‌طور که قبلاً اشاره شد، روش Scrum روشی آسان برای تولید نرم‌افزار است که مدیریت پروژه و نظم موجود در آن می‌تواند بسیار کارگشا باشد. حال تجسم کنید که روش RUP را اجرا و قسمت‌هایی از Scrum را در آن ادغام کنیم. پس از این کار متوجه خواهید شد که روش RUP می‌تواند

از مدل Scrum کمك بگیرد و با ادغام این دو می‌توان پروسه‌ای منظم برای تولید نرم‌افزارهای کوچك سازماندهی کرد. اما همان‌طور که می‌دانید نمی‌توان دو رویه ناهمگون را با هم ترکیب نمود. آیا RUP و Scrum با هم شباهت‌هایی دارند؟

همان‌طور که قبلاً بیان شد، هر دو رویه ساخت نرم‌افزار روش حلقه‌ای تکرارکننده یا Iterative را خط مشی خود قرار داده‌اند(البته در RUP تعریف بهتر و کامل‌تری از Iterative شده است). در Scrum تعریف نیازهای کاربران توسط اعضای تیم انجام می‌پذیرد، اما در RUP تنها یك شخص Requirement Engineer) یا مهندس مسئول نیازهای کاربران) است که این مسئولیت را برعهده دارد. در زمینه مدل سیستم اگر چه Scrum مسئولیت انجام این کار را به تمامی اعضای گروه داده است، اما هر دو روش از مدل UML پشتیبانی می‌کنند و استفاده از آن را پیشنهاد می‌دهند.

ضمناً هر دوی این روش‌ها روش‌های هوشمند و Iterative هستند که مدیریت و اندازه گیری کیفیت نرم‌افزار در تمامی مراحل این رویه‌ها به خوبی دیده می‌شود. همچنین هر دوی این روش‌ها انجام تغییرات را در طول پروژه مجاز می‌دانند. البته همان‌طور که در قسمت Scrum توضیح داده شد، این روش تغییرات را در طول مراحل Sprint مجاز نمی‌داند، اما مدیر Scrum می‌تواند تغییرات درخواستی توسط کاربران را جمع‌آوری و در جلسه بعدی مطرح نماید.

به تازگی RUP نیز ابزارهای جدیدی مانند RUP Builder و RUP modeller را عرضه کرده که به مدیران پروژه‌ها اجازه می‌دهد تا برخی از اصول Scrum را درRUP اجرا کنند. در نتیجه این دو پروسه تولید نرم‌افزار می‌توانند به کمك بیایند و روشی مناسب برای تولید نرم‌افزارها به‌خصوص در اندازه کوچك باشند.

روش RUP + XP

روش دومی که مورد آزمایش قرار گرفت، تلفیقی بود از اکس‌پی و RUP. ولی می‌توان گفت ادغام این دو رویه بسیار متفاوت است.

RUP رویه‌ای بسیار سنگین و اکس‌پی روشی بسیار سبك است. می‌دانید که RUP را می‌توانیم تقریباً برای تمامی نرم‌افزارهای کوچك و بزرگ به کار برد. اکس‌پی نیز همانند RUP براساس Iterationها یا مراحل پیوسته مانند تحلیل، طراحی و امتحان نرم‌افزار استوار است.

از آن جایی که RUP و اکس‌پی از اساس با هم تفاوت‌های زیادی دارند و اکثراً تصور می‌کنند که RUP راه‌حلی برای تولید نرم‌افزارهای بزرگ و اکس‌پی برای تولید نرم‌افزارهای کوچك است، ممکن شما هم تصور کنید که استفاده همزمان از هر دوی این روش‌ها کاردرستی نیست.

اما مطابق تحقیقات انجام شده به نظر می‌رسد که برای تولید نرم‌افزارهای کوچك روشی بین RUP و اکس‌پی نیاز است.در نتیجه با اضافه‌کردن برخی از تکنیك‌های اکس‌پی به RUP می‌توان به رویه‌ای مناسب‌تردست یافت. قبلاً نیز محققانی روی RUP کار کرده‌اند تا آن را برای پروژه‌های کوچك مناسب سازند. مثلاً در سال ۲۰۰۰ یك نسخه از RUP به نام dX معرفی گردید که RUP مختصر شده‌ای بود. برای نرم‌افزارهای کوچك (که اعضای پروژه اغلب در یك محیط کار می‌کنند) اکس‌پی می‌تواند روشی بسیار خوب باشد، اما اگر اعضای تیم پراکنده باشند و سیستم بخواهد توسعه یابد، اکس‌پی قادر به جوابگویی نیست و می‌توان گفت که با استفاده از قسمت‌هایی از روش قدرتمند RUP می‌توان به اکس‌پی کمك نمود.

برای تلفیق این دو روش تصورکنید که پروژه‌ای شروع شده است. در مرحله Inception یا آغازین

می‌توان از تکنیك‌های اکس‌پی در زمینه برنامه‌ریزی زمانی و جمع آوری نیازهای سیستم استفاده نمود. البته نمی‌توان گفت که همیشه این دو روش با هم سازگار هستند. مثلاً در اکس‌پی مرحله‌ای به نام طراحی یا Design Phase وجود ندارد. در صورتی که RUP یك مرحله مجزا برای این قسمت دارد.

روش Iterative Process

شاید به نظر برسد که در پروژه‌های کوچك، اعضای گروه نیاز کمتری به ارتباط با یکدیگر دارند. اما از آن جایی که در این گونه پروژه ها ارتباط بین اعضای تیم و کاربر نزدیك‌تر است و عوامل خارجی نیز نقش مهمی را در پروژه بازی می‌کنند، در این پروژه‌ها نیاز به ارتباط بین اعضای تیم محسوس به نظر می‌رسد. همچنین اگرچه پروژه‌های نرم‌افزاری کوچك طبیعتاً نیاز به نوشتن کدهای کمتری دارند و ممکن است به چند مدیر نیاز نداشته باشند اما مانند پروژه‌های بزرگ باید نرم‌افزاری با کیفیت بالا ارائه دهند. در نتیجه می‌توان گفت که روشی برای تولید نرم‌افزار کوچك مناسب‌تر است که تمامی موارد مذکور را در نظر بگیرد و اجرا کند.

رویه Iterative یکی از این روش‌ها است. با استفاده از این رویه دو نوع محصول به نام‌های Actual و by-product تولید می‌گردد. در واقع محصولاتی که در موفقیت پروژه نقش اساسی بازی می‌کنند، Actulas و آن دسته که به وجود آمدن Actualsها کمك می‌کنند را By-Product می‌گویند (مثلاً طرح اولیه سیستم). در این مدل هر عضو از گروه مسئول انجام‌دادن قسمتی از کار می‌شود و این مدل شامل هشت مرحله یا فاز است.

اولین مرحله این رویه جمع‌آوری اطلاعات از کاربر است. در مرحله بعدی سیستم به صورت جامع تحلیل و آنالیز می‌گردد تا اعضای تیم با مدل کلی سیستم آشنا گردند. سپس در مرحله تحلیل، نرم‌افزار به صورت کلی مورد بررسی قرار می‌گیرد و پس از آن که مرحله معماری سیستم نام دارد، اجزای تشکیل‌دهنده سیستم مشخص می‌شوند و کارایی‌های سیستم مشخص می‌گردند. در

مرحله طراحی تمامی اجزای سیستم طراحی می‌شوند و در مرحله بعد کدهای سیستم نوشته می‌شود.

وقتی این مرحله تمام شد و کدهای سیستم نوشته شد، اعضای تیم در فاز جمع‌بندی کدهای سیستم را با توجه به مراحل اول تا پنج مرور می‌کنند. در مرحله آخر نیز اعضای گروه را امتحان می‌کنند تا اولاً نیازهای کاربران را تأمین کرده باشد و ثانیاً فاقد هرگونه اشکال باشد. اگر نرم‌افزار فاقد اشکال باشد، رویه تولید نرم‌افزار به آخر خواهد رسید. در غیر این صورت، اعضای گروه به دنبال منبع مشکل در مراحل قبلی می‌گردند و مجدداً رویه را از آن جایی که فکر می‌کنند باعث بروز اشکال شده است، ادامه می‌دهند.

نتیجه گیری

برای دستیابی به موفقیت در پروژه‌های نرم‌افزاری، اعضای گروه باید از یك رویه یا روش مشخص پیروی کنند. اما برای پروژه های کوچك (برای تولید نرم‌افزارهای کوچك) این رویه باید ساده و آسان باشد. اضافه براین، برای دستیابی به موفقیت در پروژه‌ها تنها داشتن یك رویه آسان و کارا کافی نیست بلکه مدیران پروژه‌های نرم‌افزاری باید به این نکته توجه کنند که اعضای گروه در موفقیت پروژه‌ها از اهمیت شایانی برخوردار هستند و باید در انتخاب این افراد حداکثر دقت را مبذول نمود. در ضمن موقع انتخاب یك رویه مناسب باید اندازه نرم‌افزار را معین نمود و براساس نیازهای کاربران پروسه تولیدی را طراحی کرد. برای تعیین رویه‌ای مناسب در تولید نرم‌افزارهای کوچك باید دقت داشت که این رویه باید بسیار ساده باشد تا به اعضای تیم کمك کند به‌راحتی مراحل تهیه نرم‌افزار را ادامه دهند.

از جمله این رویه‌های ساده می‌توان از Scrum نام برد. Scrum یك تکنیك مدیریت پروژه است که می‌تواند به تیم‌های نرم‌افزاری کوچك که روی پروژه‌های کوچك نرم‌افزاری کار می‌کنند کمك کند راندمان و کارایی بالاتری در کار داشته باشند. اما اگر این روش‌ها را با روش‌های مناسب دیگر ادغام

کنیم، می‌توانند بیشتر مفید واقع گردند.

پس از Scrum، روش اکس‌پی توضیح داده شد و به عنوان بهترین راه برای تولید نرم‌افزارهای کوچك از آن نام برده شد. اما این روش به تنهایی کارایی چندانی نخواهد داشت. سپس RUP که می‌تواند در تمامی پروژه‌ها استفاده شود تشریح شد و در ادامه سه روش مناسب برای تولید نرم‌افزارهای کوچك ارائه گردید. اما همان‌طور که بحث شد، داشتن یك روش مناسب به تنهایی نمی‌تواند ضامن موفقیت در پروژه باشد، بلکه انتخاب منابع انسانی مناسب و با تجربه می‌تواند راه را برای موفقیت پروژه‌های نرم‌افزاری هموارتر سازد

منبع: ماهنامه شبکه