رستا روغنی(۹۷۱۰۵۹۶۳)

۱ مقایسه FDD و XP

۱.۱ Work Units

۱.۱.۱ مراحل FDD [۱]

FDD پنج مرحله دارد:

۱. develop an overall model: یک object model ساخته می‌شود که ساختار مسئله را مشخص می‌کند. این کار به صورت iterative انجام می‌شود و تیم‌ها به تیم‌های کوچکتر تقسیم می‌شوند تا هربار مدل‌های مختلف پیشنهاد دهند و فیدبک دریافت کنند و انقدر این کار تکرار می‌شود که تمامی domain ها پوشش داده‌شوند.

۲. build a features list: در این مرحله area های مختلف و activity هایی که در هرکدام انجام می‌شوند و قدم‌های هر activity و هایی‌feature که برای پیاده‌سازی‌شان نیاز است، مشخص می‌شوند.

۳. plan by feature: در این قدم تیم برنامه‌ریزی (شامل مدیر پروژه و برنامه‌نویسان اصلی و یک مدیر توسعه) تشکیل می‌شود و تاریخ پیاده‌سازی هر activity مشخص می‌شود. درنهایت هر activity به یکی از برنامه‌نویسان اصلی داده می‌شود و کلاس‌ها نیز بین برنامه‌نویسان دیگر تقسیم می‌شوند.

۴. design by feature: تحت نظر هر برنامه‌نویس اصلی، یک تیم تشکیل می‌شود و افرادی که صاحبان کلاس‌های مربوط به آن feature set مخصوص هستند، کنار هم قرار می‌گیرند. هدف آن‌ها در این مرحله تولید sequence diagram ها و بهبود object model به‌گونه‌ای است که آن‌ها را دربر بگیرد.

۵. build by feature: کلاس‌ها و متدهای لازم پیاده‌سازی می‌شوند و unit test ها روی آن‌ها اجرا می‌شوند تا از کارکرد صحیح‌شان اطمینان پیدا کنند. درنهایت اگر به درستی پیاده‌سازی شده‌باشند، در ساختار اصلی پروژه وارد می‌شوند.

۲.۱.۱ مراحل XP [۱]

XP شش مرحله دارد:

۱. exploration: مشتری یک لیست اولیه از نیازمندی‌های سطح بالا (در قالب user story ها) را تعریف می‌کند و یک طراحی کلی از سیستم مشخص می‌شود.

۲. planning: در این مرحله تیم نیازمندی‌ها را اولویت‌بندی می‌کند و زمان لازم برای هر نیازمندی را مشخص می‌کند. درنهایت یک برنامه برای ریلیس پروژه می‌دهد.

۳. Iterations to First Release: هدف این مرحله تولید اولین ریلیس پروژه طبق برنامه‌ای است که در مرحله‌ی قبلی مشخص شد. بطور کلی سه عمل آنالیز، طراحی و تست در هر ایتریشن انجام می‌شوند و در انتها فیدبک داده می‌شود که ممکن است منجر به تولید user story ها و نیازمندی‌های جدید و یا تغییر در نیازمندی‌های کنونی شود.

۴. productionizing: تمرکز این مرحله روی verification و validation هر ریلیس و پیاده‌سازی آن در محیط کاربری است.

۵. maintenance: در این مرحله هدف پیاده‌سازی نیازمندی‌های باقی در سیستم کنونی است.

۶. death: مرحله‌ی پایانی است که در آن مستندسازی و تبادل نظر بین مشتری‌ها و تیم صورت می‌گیرد.

همانطور که در بالا می‌بینیم، گام اول هر دو پروسه نسبتا کار مشابهی می‌کنند (به‌دست آوردن نیازمندی‌ها) با این تفاوت که این گام در FDD به‌صورت iterative انجام می‌شود اما در XP مشتری در ابتدا user story هایش را می‌نویسد و تیم از روی آن‌ها نیازمندی‌ها را بدست می‌آورد. البته درطول این فرایند ممکن است تغییری در user story ها داده شود که در این صورت آن‌ها را نیز درنظر می‌گیرند. اما در کل در XP انرژی زیادی روی جمع‌آوری نیازمندی‌ها گذاشته نمی‌شود (که می‌تواند باعث ایجاد استرس شود) درحالیکه یکی از عوامل موفقیت یک پروژه‌ی FDD ، تعریف دقیق آن‌هاست. معمولا در انتهای این مرحله FDD مستندی از کلیه‌ی راه‌های پیشنهادی‌ای که در این مرحله بدست آمدند و راه‌های جایگزین در صورت برخوردن به مشکل، ارائه می‌دهد اما XP این کار را در گام آخر انجام می‌دهد. گام دوم FDD برای XP تعریف نشده‌است چراکه در XP موضوع feature sets مطرح نیست. گام سوم FDD و گام دوم XP هر دو به برنامه‌ریزی و تعیین ددلاین‌ها به شکل iterative می‌پردازند. تفاوت در این است که در XP همه‌ی اعضای تیم جلسه‌ای با مشتری خواهند داشت تا او مشخص کند که کدام نیازمندی‌ها پیاده شوند درحالیکه در FDD ، درهر iteration برنامه‌نویسان اصلی (chief programers) تیم‌های جدیدی را تشکیل می‌دهند و مجموعه‌ای از feature ها که بهم مرتبط هستند را به‌صورت یک پکیج دریافت می‌کنند و آن را بین اعضای تیم‌شان تقسیم می‌کنند. (این iteration ها هم برای هر تیم مستقلا و درگروه‌های کوچک بین ۳تا ۵ نفره انجام می‌شوند که متفاوت با XP است). FDD به‌طور مجزا برای طراحی گامی درنظر گرفته‌است (گام چهارم) که XP این عملیات را در ابتدای گام سوم خود انجام می‌دهد. FDD در این مرحله مستندسازی دارد (که مانند گام ششم XP است است اما به شکل iterative) و جلساتی را برگزار می‌کند که به تشخیص زودهنگام تصمیمات اشتباه کمک می‌کند. درنهایت گام پنجم FDD بصورت iterative کار مشابهی با گام سوم و چهارم و پنجم XP انجام می‌دهد. پس بطور خلاصه میتوان گفت که برای پروژه‌هایی که در آن‌ها نیازمندی‌های کاربر مشخص نیستند و یا دائما تغییر می‌کنند XP بهتر است و درمقابل برای پروژه‌هایی که از ابتدا نیازمندی‌ها در آن نسبتا پایدارتر باشند، FDD بهتر عمل می‌کند. همچنین در FDD تحت نظر گرفتن iteration ها ساده‌تر است چراکه meeting های مجزایی برای هر کار دارد و progress tracking دقیقی را تعریف کرده‌است و البته طول دوره‌هایش نیز کمتر است (بین دو تا ده روز دربرابر حدود ۶ هفته‌ی XP). از دیگر تفاوت‌های FDD و XP تاکید XP بر مکالمه‌ی افراد است درحالیکه FDD بیشتر روی تولید مستند تمرکز می‌کند. درنهایت اینکه عملیات بازخوانی کد در XP طی فرایند pair programming انجام می‌شود اما در FDD به ازای هر iteration جلسه‌ای برای آن برگزار می‌شود که چون تعداد حاضرین جلسه بیشتر است، احتمال پی بردن به مشکل کد از زمانی که تنها یک نفر بر آن نظارت می‌کند بیشتر است. [۲]

Products ۲.۱

XP	FDD
metaphor	object model
user story cards	domain walkthroughs
task cards	features
-	features list (activities)
-	model notes
release plan	development plan
-	design package
iteration plan	iteration plan(work package)
-	class and method prologues
unit tests	unit tests
-	unit test reports
post-mortem documentation	-

در جدول بالا محصولاتی که کاربرد نسبتا مشابهی داشتند در مقابل هم قرار گرفتند و می‌توان تفاوت محصولات دو پروسه را دید. به‌طور مشخص می‌توان مشاهده کرد که در FDD به‌طور مرتب مستندهایی ساخته می‌شوند اما دربرابر آن XP تنها به تولید یک مستند در پایان کار پروژه بسنده می‌کند. [۳] البته به این نکته هم باید توجه کرد که metaphor درواقع یک توصیف ساده و لول بالا از روش کار سیستم است و یک ایده‌ی کلی درباره‌ی معماری سیستم به ما می‌دهد اما object model شامل class diagram ها و حتی sequence diagram ها است تا پترن‌های رفتاری ضروری را مشخص کند و بنابراین بسیار باجزئیات‌تر از metaphor است.

Roles ۳.۱

۱.۳.۱ نقش‌ها در FDD [۴، ۵، ۶]

در FDD نقش‌ها در سه دسته‌ی کلی اصلی، ساپورت و اضافی قرار می‌گیرند:

- key roles

- product manager: رهبر پروژه است و بر بودجه، تعداد اعضای تیم‌ها، رساندن به موقع فیچرها و درست کردن مستندات پروژه نظارت می‌کند. علاوه‌بر این وی وظیفه دارد که پیشرفت پروژه را با مشتری درمیان بگذارد و از درستی روند آن اطمینان حاصل کند.
- chief architect: طرح کلی سیستم (blueprint) را می‌دهد و آن را به اعضای تیم پروژه یاد می‌دهد تا با اطلاعات کافی تسک‌هایشان را انجام دهند. همچنین پروژه را از موانع فنی عبور می‌دهد.
- development manager: تیم توسعه‌دهندگان (developers) را مدیریت می‌کند تا کارشان را سر وقت تمام کنند و به همین منظور روزانه بر عملکرد تیم‌ها نظارت می‌کند. درصورتیکه میان تیم‌ها مشکلی باشد، آن را حل می‌کند. همچنین با product manager و chief architect در ارتباط است و هرمدت یکبار به آن‌ها گزارش می‌دهد.
- chief programmer: یکی از باتجربه‌ترین برنامه‌نویسان است و در طراحی و آنالیز به اعضای یک تیم فیچر (feature team)کمک می‌کند و مطمئن می‌شود که درجهت درستی در حرکت هستند.

- class owners: افرادی هستند که در تیم‌های کوچکتر فیچرها را طراحی و پیاده می‌کنند. آن‌ها از chief programmer پیشنهادات و راهنمایی‌هایش را دریافت می‌کنند و در نهایت به development manager گزارش می‌دهند. تست کردن اولیه و ساختن مستندات از دیگر وظایف آن‌هاست.
- domain experts: افرادی هستند که اطلاعات لازم را درمورد یک دامنه‌ی خاص و نیازمندی‌های کاربر دارند و به تیم در فهمیدن آن‌ها کمک می‌کنند. این افراد می‌توانند کاربر، مشتری و یا اسپانسر باشند.

- supporting roles
  - release manager: بر روند پروسه نظارت می‌کند.
  - language lawyer: درمورد زبان برنامه‌نویسی مورد استفاده و تکنولوژی‌ها اطلاعات گسترده دارد.
  - build engineer: مسئولیت فرایند ساخت و کنترل ورژن‌ها برعهده‌ی او است.
  - tool-smith and system administrator: پیشتیبانی فنی پروژه را به عهده دارد.
- additional roles
  - testers: از اینکه سیستم نیازمندی‌ها را برطرف می‌کند اطمینان حاصل می‌کنند.
  - deployers: سازگاری داده‌ها را حفظ می‌کنند و ریلیس‌های جدید را آماده می‌کنند.
  - technical writers: مستندات کاربر را آماده می‌کنند.

۲.۳.۱ نقش‌ها در XP [۷، ۸]

- customer(business/product owner): مشتری برای پروژه هدف‌گذاری و اولویت‌بندی انجام می‌دهد. مسئولیت نوشتن user story ها و توضیح آن‌ها به تیم، انتخاب سوالات مربوط به story ها و تامین بودجه با او است.
- coach: مسئولیت نظارت کلی بر پروژه و مسیری که طی می‌کند بر عهده‌ی اوست. درصورت بروز مشکل، وی دستورات XP که به حل آن کمک می‌کنند را تشخیص می‌دهد و به کمک تیم می‌رود. او جلسات را برنامه‌ریزی می‌کند و نتایج آن‌ها را به tracker می‌دهد.
- programmer: فردی است که در هر increment کارهایی چون درآوردن تسک‌ها از story ها، تخمین زمان هر تسک، پیاده‌کردن روایت‌ها و unit test ها را انجام می‌دهد.
- manager: به gold owner گزارش می‌دهد و از تحویل دادن پروژه اطمینان پیدا می‌کند. او برنامه‌ریزی جلسات برای iteration/release planning را انجام می‌دهد و درصورت نیاز می‌تواند مسئولیت‌های tracker و doomsayer را پوشش دهد و نتایج را به آن‌ها گزارش دهد. کارهای دیگری همچون نظارت برمدت زمان کار هر عضو تیم و مشکلات تست‌های عملکردی، تعیین قواعد برنامه‌ریزی و آشنا کردن اعضای تیم و مشتری با این قواعد را نیز به عهده دارد.
- tester: تست‌های عملکردی (functional) را پیاده‌سازی و اجرا می‌کند تا از کارکرد کد مطمئن شود و نتایج خود را به شکل گراف تحویل می‌دهد تا درهر زمان آن‌ها را دنبال کند و اگر مشکلی در نتایج بود، آن را به فرد مسئول اطلاع دهد.
- tracker: تمرکز او روی این است که همه‌چیز مطابق با برنامه پیش برود و بر برنامه‌نویسان نظارت می‌کند تا درصورت نیاز جلسه‌ی CRC و یا جلسه‌ای بین برنامه‌نویسان و مشتری برگزار کند.
- doomsayer: همه‌ی اعضای پروژه را از ریسک‌ها آگاه می‌کند و مانع فراموش شدن نتایج بد می‌شود.

- gold owner: کسی است که پول پیاده‌سازی پروژه را پرداخت می‌کند.

بنابراین FDD نقش‌های مدیریتی‌ای دارد که در XP برعهده‌ی کسی نیستند(همچون chief architect یا chief programmer) که بخاطر همین ساختار سلسله‌مراتبی، مقیاس‌پذیر است و برای تیم‌های بزرگ و سازمانی به کار می‌رود و مدیریت‌شان را آسان می‌کند اما تمرکز XP بر ارتباط بین اعضای تیم است و برای تیم‌های کوچک (بین ۲ تا ۱۰ نفره) بکار برده می‌شود. همچنین در FDD نیاز به حضور دائمی مشتری نیست و توسط نقش‌های دیگر به او گزارش داده می‌شود برخلاف XP که مشتری باید در روند تولید حضور دائمی داشته‌باشد. یکی دیگر از تفاوت‌ها در این است که در FDD برنامه‌نویسان دارای تجربیات و توانمندی‌های متفاوتی هستند که بخاطر نقش‌های مدیریتی مضاف در FDD ، امکان بهره‌مندی از آن‌ها وجود دارد درحالیکه در XP معمولا توانایی همه‌ی برنامه‌نویسان در یک سطح است. درنهایت در FDD بحث code ownership مطرح است و هر فرد صاحب کدی است که می‌زند که ممکن است برای سیستم‌های قدیمی مناسب نباشد اما در XP صحبت از collective ownership است که یعنی هرفردی می‌تواند در کد دیگر اعضای تیم تغییر ایجاد کند که این ویژگی زمانی که یکی از اعضا تیم را ترک می‌کند، به کمک می‌آید چرا که اعضای دیگر در مرور زمان دسترسی به کدهای او داشته‌اند و آگاهی زیادی نسبت به کل سیستم دارند.

البته در میان نقش‌ها شباهت‌هایی هم دیده می‌شود از جمله اینکه هردو به اهمیت مدیریت آگاهی دارند و نقش‌هایی در آن‌ها تعریف شده که بتوانند نظم را به واسطه‌ی آن‌ها حفظ کنند. علاوه‌براین، هردو برحضور analyst ها در کنار کاربران و توسعه‌دهندگان اصرار دارند.

۲ منابع

1. http://sharif.edu/~ramsin/index_files/Publications_PDF/Ramsin_Paige_Technical_Report.pdf
2. http://www.featuredrivendevelopment.com/files/FDD_vs_XP.pdf
3. https://www.informit.com/articles/article.aspx?p=26055&seqNum=4#:~:text=Both%20FDD%20and%20XP%20are,%2Dpage%20specifications%20to%20write).
4. https://link.springer.com/content/pdf/10.1007/978-3-540-85279-7_16.pdf
5. https://launchdarkly.com/blog/feature-driven-development-a-brief-overview/
6. https://medium.com/@avishaa27/roles-responsibilities-of-fdd-6e9bcff0141f
7. https://pm-training.net/extreme-programming-roles/
8. https://www.tutorialspoint.com/extreme_programming/extreme_programming_roles.htm