بسمه تعالی

محمدرضا سابقی ۹۹۲۰۱۴۲۱

متدولوژی ایجاد نرم افزار

استاد: دکتر رامسین

تمرین دوم

زمستان ۱۳۹۹

فهرست مطالب

۱ مقایسه ۳ متدولوژی با OPM

۱.۱ معیارهای فرآیندی

۱.۱.۱ تعریف

این معیار، نیازمندی ای که مطرح می کند این است که متدولوژی مدنظر باید به خوبی تعریف شده باشد(تعریفش به خوبی مستند شده باشد). یک توصیف خوب صفات جامع، روشن، منطقی، صحیح و دقیق، با ریز جزئیات و بدون تناقض را به همراه دارد. حال برای رسیدن به درک درستی از آن، مفاهیمی باید ثبت شود که ابتدا برای هر متدولوژی پرداخته و سپس ذکر می کنیم در کدام جنبه در مقام مقایسه، برتری وجود دارد.

- متدولوژی OPM: این متدولوژی قبل از آن که به شکل متدولوژی ارائه شود، یک روش مدلسازی بوده است(وجود دیاگرامی به نام OPD و مبنا قرار دادن آن و گذاشتن آن به عنوان یک مدل محوری در یک متدولوژی). به همین منظور واضع آن، متدولوژی OPM را تعریف کرده و در چارچوب بسیار ساده ای که مبتنی بر فرآیند جنریک ایجاد نرم افزار بوده، متدولوژی را بیان می کند. چرخه ی عمر آن بزرگ و پوشا بوده اما کم عمق می باشد و حاوی جزئیات نبوده و از این نظر مبهم می باشد. بیشتر تمرکز بر روی ورود زبان مدلسازی OPD به یک فرآیند که چرخه ی عمر دارد، بود(پس بدیهی است که زبان مدلسازی آن OPD می باشد). به همین منظور تعریف آن در سطح بالایی بوده و فارغ از جزئیات لازم برای تعریف چیستی فرآیند هستند. فراتر از آن هیچ تصریحی برای افراد و نقش ها در متدولوژی ذکر نشده است. به تعریف فعالیت های چتری و همچنین تکنیک ها و قواعدی که بیان کننده ی چگونگی اجرای واحد های کاری می باشند، دقتی نشده است. همچنین تولید محصولات مشخصی در فرآیند

آن دارای ابهام است و اشاره ای به آن نشده است. در نهایت این که تعریف فرآیند به صورت process-centered می باشد؛ چرا که محصول یا حتی نقشی در فرآیند مشخص نشده است اما فاز ها و زیرفاز ها به صورت انتزاعی آورده شده اند.

- متدولوژی EUP: متدولوژی EUP در تعریف بسیار خوب عمل می کند. مستندسازی آن تا حد بالایی دقیق و مفصل به همراه جزئیات واضح می باشد. تعاریف فاز ها و گام ها در آن روشن هست و تصریح های آن صحیح و دقت بوده و تناقضی در آن یافت نمی شود. چرخه ی عمر آن دارای ۴ فاز می باشد و برای هر فاز رویکردی افزایشی-تکرارشونده شامل ۷ دیسیپلین تکرار می شود. واحد های کاری مشخص هستند و زبان خود را UML معرفی می کند اما در آن تعصبی به خرج نداده و اجازه استفاده از نمودار های خارج از UML را نیز می دهد. قواعد و تکنیک ها درون هر واحدکاری تا حد بسیار بالایی تصریح شده اند(در حدی که برای هر کدام طول مدت و تعداد تکرار ها پیشنهاد می شود). در فرآیند، افراد درگیر معین می شوند(نقش های سازمانی) و همچنین بودن فعالیت های بسیار کنترلی-مدیریتی در آن، نشان از حضور فعالیت های چتری دارد. محصولات در حد خوبی، جزئیات دارند ولی رویکرد تعریف آن process-centered هست.

- متدولوژی Fusion: این متدلوژی ۳ فاز دارد که محدود به تحلیل و طراحی و پیاده سازی بوده و پوشایی خوبی از این نظر ندارد. در هر صورت Fusion بسیار خوب قادر هست تا محصولات خود را شرح داده و بگوید در نتیجه ی چه واحد کاری محصولات تولید خواهند شد. پس گام هایی با جزییات بالا در آن به چشم می خورد. این متدولوژی نمی تواند صحبتی از نقش افراد درگیر در فرآیند بیاورد و Fusion زبان مدلسازی خود را معرفی و با notation تعریفی خود ارائه می دهد و از این نظر محصولاتی با جزییات تقریبا بالایی به ارمغان می آورد. این که در Fusion فنون و تکنیک هر گام جداگانه ذکر می شود، حکایت از توجه

متدولوژی به تعریف تکنیک ها و قواعد دارد منتها نمی تواند توجهی به فعالیت های چتری بکند. متدولوژی Fusion به صورت process-centered می باشد.

- متدولوژی OOSE: این متدولوژی ۳ فاز را تعریف می کند که علاوه بر تعریف بسیار خوب و روشن آن به همراه ذکر جزییات، محصولات دقیقی را نیز تعریف می کند. ۳ فاز خود را در مجموع با ۵ مرحله تصریح می کند که همگی با یکدیگر به صورت سازگار تعریف شده اند. در کنار تعریف جامع و دقیق واحد های کاری، تکنیک هایی که بیان کننده ی چگونگی انجام کار هستند به علاوه ی زبان مدلسازی، دارای دقت هستند. نقش افراد اما در آن مغفول مانده است و به صورت process-centered تعریف شده است. متدولوژی OOSE راهکاری را برای فعالیت های چتری که ناظر بر کنترل و مدیریت هستند ارائه نمی کند.

- مقایسه OPM با EUP: متدولوژی EUP بسیار تعریف دقیقتری دارد و جزییات و مراحل عمده تری را در بر دارد. نقش افراد را می تواند پیشنهاد دهد و محصولات آن در قالب های متفاوت هست در حالی که OPM از یک مدل برای همه ی جنبه ها استفاده می کند. متدولوژی EUP فعالیت های چتری را به خوبی تصریح می کند ولی OPM این گونه نیست. هر دو متدولوژی در تعریف محصولات، قابل قبول و شفاف عمل کرده اند. تکنیک ها و قواعد داخل هر چرخه ی کاری را EUP بسیار بهتر تشریح می کند و از این نظر برتری دارد و در عین حال هر دو process-centered هستند.

- مقایسه OPM با Fusion: هیچ کدام از دو متدولوژی نمی توانند تعریفی برای نقش افراد درگیر تیم ایجاد ارائه دهند. محصولات در هر دو به خوبی تعریف شده اند اما Fusion زبان خاص خود را معرفی می کند در حالی که OPM از OPD استفاده می کند. تکنیک های بسیار خوبی در Fusion تعبیه شده که

حتی مراحل میانی در یک واحد کاری را خیلی خوب تشریح می کند. هیچ کدام توجهی به فعالیت های چتری ندارند و هر دو process-centered هستند. در این مقایسه، Fusion در تعریف فاز ها و مراحل خود عمیق تر عمل کرده و جزییات بیشتری را در اختیار می گذارد.

- مقایسه OPM با OOSE: در تعریف جزییات هر فاز و چیستی مراحل، متدولوژی OOSE برتری دارد. به علاوه این که متدولوژی OOSE زبان مدلسازی خود را که شبیه به UML هست ارائه می دهد و مانند OPM همه ی جنبه ها را در یک مدل نمی آورد. در عین حال فعالیت های چتری در هر دو مغفول مانده و توجهی به نقش افراد در متدولوژی نمی شود. محصولات در حد بسیار خوب و منطقی در هر دو روشن شده است و تکنیک در OOSE بهتر و مفصل تر به چشم می خورد. هر دوی آن ها در عین حال process-centered هستند.

۲.۱.۱ پوشش چرخه عمر عمومی ایجاد نرم افزار

معیار بیان می کند که یک متدولوژی در چه حدی چرخه ی عمومی ایجاد نرم افزار را پوشش می دهد. کما اینکه می دانیم چرخه ی عمومی ایجاد نرم افزار در سه بخش کلی تقسیم می شود: تعریف مساله[۱]، ایجاد[۲]، نگهداری[۳] که هر کدام ناظر به بخشی از lifecycle متدولوژی است و در صورت تحقق کامل همه ی آنها، متدولوژی مدنظر full-lifecycle خواهد بود. حال هر کدام را با شرح بیشتر بررسی خواهیم کرد.

- متدولوژی OPM: این متدولوژی به شکل کاملا پوشایی تعریف شده است، یعنی خود فرآیند جنریک ایجاد نرم افزار را برای یک روش مدلسازی تعریف کرده اند(هرچند فارغ از جزییات زیاد می باشد).کاوش قلمروی مساله، استخراج نیازمندی ها در فاز initiating و زیرفاز analyzing مطرح شده اما توجه زیادی به امکان سنجی ندارد(هرچند اقداماتی برای تشخیص یا تخصیص دادن منابع را در نظر می گیرد ولی این که مطمینا تصمیمی مبنی بر go یا no go برای ورود به فاز بعد را صادر کند ندارد). در عین حال در طراحی معماری، طراحی تفصیلی، برنامه نویسی و تست پوشش خوبی را در developing فراهم می کند. فاز آخر خود را نیز به فعالیت های پشتیبانی و نگهداری اختصاص داده به همراه اینکه توجه ویژه ای به مستقر سازی سیستم در محیط کاربر دارد و از این نظر بسیار کامل عمل می کند.

- متدولوژی EUP: متدولوژی EUP بسیار غنی از این جهت عمل می کند. به صورت ماهیت خود که افزایشی-تکرارشونده می باشد، هم با فاز inception و کمک دیسیپلین های خود مانند requirements به خوبی تحلیل امکان پذیری را معین می کند و در elaboration کاوش قلمروی مساله را مخصوصا

[۱] defenition
[۲] development
[۳] maintenance

در دیسیپلین تحلیل انجام داده و آن را مدلسازی می کند. استخراج نیازمندی ها تا ۸۰ درصد را این جا معین می کند و آن را در فاز بعد کامل می نماید. طراحی را به شکل عالی چه معماری و چه تفصیلی و همچنین برنامه نویسی را در construction اهتمام می کند و برنامه ای کامل برای پوشاندن تست و استقرار هم در این فاز و transition دارد. تنها مشکلی که نمی تواند به خوبی آن را پشتیبانی کند، نگهداری است که فعالیت خاصی برای آن تعبیه نشده است. صرفا مجبور به تعریف یک فرآیند سفارشی در فاز مثلا transition هستیم تا با هر درخواست، آن فعالیت را اجرا کنیم. اگر چنین نشود، مجبور به تکرار کل فرآیند برای هر درخواست هستیم که عملا ممکن و عقلانی نخواهد بود.

- متدولوژی Fusion: متدولوژی Fusion فعالیتی را برای تحلیل امکان پذیری ارائه نمی دهد. استخراج نیازمندی در آن تعبیه نشده و سند نیازمندی را ورودی متدولوژی می داند. کاوش قلمرو مساله اما در فاز اول آن که تحلیل باشد، با حساسیت بالایی انجام می گیرد. در این مرحله می توان سیستم و مرز آن را به طور دقیقی شناخت و فقط از مرز سیستم و نه داخل آن، مدل رفتاری-وظیفه ای ارائه داد. متدولوژی طراحی سطح بالای معماری و تفصیلی را از هم جدا نکرده و هر دو را در یک فاز موسم به طراحی انجام می دهد. قصدش آن است که در این فاز بتوان با افزودن جواب های قلمرو جواب و ارتباط آبجکت های آن، تحقق operation ها را بررسی نمود که در این راه با مدل هایی که ارائه می دهد می تواند نمود خوبی از طراحی را به نمایش بگذارد. این متدولوژی در آخرین فاز خود، برنامه نویسی را در بر می گیرد اما فکری برای تست، مستقرسازی و نگهداری نمی کند.

- متدولوژی OOSE: این متدولوژی در فاز اول خود که analysis نام دارد، با تعریف دو زیرفاز می تواند استخراج نیازمندی ها را به خوبی پشتیبانی کند. متدولوژی OOSE با تعریف تحلیل استحکام می تواند فضای قلمرو مساله را

بهتر نیز بکاود؛ چرا که سعی می کند در قالب آبجکت های متخصص و تعامل آن ها با یکدیگر نشان دهد نیازمندی ها چگونه محقق می شود. در همین راه در فاز construction خود با زیرفاز design سعی در معین نمودن طراحی سیستم دارد به طوری که بتواند طراحی تفصیلی را نیز نشان دهد؛ چرا که خود محیط پیاده سازی را تعریف می کند، معماری را توصیف می کند، دیاگرام تعامل در سطح قلمرو جواب ترسیم می کند، کلاس های طراحی موسوم به block را تدقیق می کند و در کل دیاگرام کلاس طراحی را که دارای روابط و جزییات داخلی هم هست را نشان می دهد. اما با وجود برنامه نویسی و تست، توجهی به تحلیل امکان پذیری ندارد و استقرار و نگهداری را در بر نمی گیرد.

- مقایسه OPM با EUP: در کل OPM می تواند نگهداری را به شکل خیلی بهتری عرضه کند در حالی که EUP به صورت نامطلوبی آن را ارائه می دهد و تصریحی برای آن ندارد. در عوض EUP در تحلیل امکان پذیری خیلی بهتر از OPM عمل کرده(دقیقا مرحله ای را با حضور خبرگان و مجربان تدارک دیده است که تحلیل می شود آیا ورود به فاز های بعدی صرفه دارد یا خیر و اگر دارد پلن کلی تقریبا خوب تا آخر پروژه چه چیز هایی خواهد بود) و در بقیه ی موارد هر دو پوشش خوبی داشته اند.

- مقایسه OPM با Fusion: در کل OPM برتری زیادی نسبت به Fusion دارد. توجه به پشتیبانی و نگهداری، مستقرسازی، تست و استخراج نیازمندی ها از مزایای آن به شمار می روند. هر دو متدولوژی توجه زیادی به امکان سنجی ندارند ولی در عین حال OPM کمی بهتر در شناسایی منابع در تحلیل مقدماتی خود موسوم به initiating رفتار می کند. در بقیه موارد نیز مانند هم عمل می کنند.

- مقایسه OPM با OOSE: متدولوژی OPM در اینجا هم بهتر عمل می کند و پوشایی خوبی نسبت به OOSE دارد. تصریح نگهداری و استقرار از برتری های

آن هست و البته هر دو امکان سنجی را تا حدودی ندید می گیرند. البته در این زمینه OPM برتری هایی دارد که در بالاتر نیز آمد.

۳.۱.۱ پشتیبانی از فعالیت های چتری

معیار پشتیبانی از فعالیت های چتری بیشتر ناظر بر اقدامات مدیریتی و سیاست های اتخاذ شده در سطح مدیریت برای متدولوژی ها می باشد. تعداد ۳ مبحث کلی باید در این چتر به آنها توجه شود که برای هر کدام تکنیک هایی موثر می تواند به کار گرفته شود:

- **مدیریت ریسک**

منظور از مدیریت کردن ریسک این است که فعالیت های مربوط به ارزیابی ریسک و کاهش آن چگونه در داخل فرآیند متدولوژی تعریف شده است. برای این مدیریت، تعدادی تکنیک موثر تعریف شده است که برای متدولوژی های زیر نگاهی به آن تکنیک های به کار گرفته شده خواهیم داشت.

متدولوژی OPM: متدولوژی OPM در سطح تحلیل مقدماتی خود که در ini- tiating انجام می شود در صدد مشخص نمودن scope می باشد که عمدتا با این کار، نیازمندی های وظیفه ای را به همراه کارهایی که در حیطه ی وظیفه ی سیسنم نیست در سطح بالایی تعریف تعریف می کنند و برای آن ها منابع لازم مشخص می شوند که بدیهی است سطحی از مدیریت ریسک را در این جا شاهد هستیم و درصدی از ریسک را توجه داریم؛ چرا که در حال کشف چیستی سیستم و تخصیص منابع به آن ها هستیم. یکی از مواردی که در این مرحله به آن توجه می شود معین کردن توجیه اقتصادی پروژه می باشد که باید در سطح سازمان انجام شود و معیار خوبی برای سنجش و کاهش آن می باشد. پس تحلیل مقدماتی که از تکنیک های مدیریت ریسک می باشد را در حد مطلوبی ولی نه کامل پوشش می دهد(چرا که دقت داریم تحلیل مقدماتی از جنبه های زیادی انجام نشده و نمی توان تصمیم قطعی برای ادامه دادن گرفت). توجهی به پروتوتایپینگ، طراحی برنامه ریزی ها بر حسب ریسک وظایف، فرآیند افزایشی_تکرارشونده، ترخیص

زودرس و بازنگری تکرارشونده در آن نمی شود و کاربر در حد پذیرش های نهایی دخالت دارد که خیلی نمی تواند مطلوب باشد؛ چرا که دخالت کاربر به صورت فعالانه در کنار تیم ایجاد می تواند خیلی موثرتر می باشد. اما در عین حال -val idation را می تواند حاصل کند و آن را تا حد کمی استمرار ببخشد که خود، سطحی از مدیریت ریسک را شامل می شود.

متدولوژی EUP: متدولوژی EUP از این نظر بسیار فوق العاده می کند و در طول ایجاد نرم افزار، ریسک های مختلفی را سنجیده و درصدد حل آنها بر می آید. مهم ترین آن رویکرد افزایشی-تکرارشونده ی خود می باشد. با این ماهیت، می توان تمرکز روی بخش های مختلف نرم افزار را جابجا کرد و بازنگری در محصول و فرآیند را ممکن می سازد. کامل شدن تدریجی می تواند امکان حجیم بودن واحدهای کاری را کاهش داده و کارها کم کم برداشته و تکمیل شوند. بدین طریق حتی می توان بر حسب ریسک حرکت کرد و برای وظایف ریسکی تر تدبیر کرد. در آخر هم تیم ایجاد را به ورود به قلمرو جواب مجبور کرده که دید خوبی از ریسک های مهم آنجا را نیز می دهد.می دانیم که EUP گسترش یافته RUP می باشد و می دانیم که در RUP پروتوتایپ های زیادی برای سنجش و مدیریت ریسک تولید می شود که در انواع دورریختنی و تکاملی می باشند و مهم ترین آن executable architecture baseline می باشد که در تکامل و توسعه نرم افزار مخصوصا در تحلیل و طراحی کمک شایانی به سنجش ریسک می کند. بدین طریق که کاربر بازخورد بسیار بهتری را می تواند ارائه دهد. بدین سبب دخالت فعال کاربر را نیز ممکن می سازد و یک validation مستمر دارد. تحلیل مقدماتی آن در inception که در نهایت آن تصمیمی برای ادامه دادن پروژه گرفته می شود درصد بالایی از ریسک های مهم را پوشش داده و نخبگان در تصمیم go یا no go می توانند موثر باشند و همچنین پلن کلی پروژه به صورت تقریب بالا و پلن دقیق تر فاز بعدی معین شده که نشان از سنجش انواع ریسک

ها به صورت دقیق بوده و کمکی برای مدیریت ریسک در آینده می باشد. البته EUP شاید ترخیص زودرس نداشته باشد اما چون در سطح کلان سازمانی است شاید توجه به این مهم از اهمیت خاصی برخوردار نباشد. تست های مستمر هم به دلیل آزمون گرفتن از پروتوتایپ ها از همان ابتدا موجود است و بازنگری های مکرر در آن به شدت عالی است. یعنی هر ۳ تا P در آن به صورت تکرارشونده ای بازنگری می شوند. این سه P عبارتند از process که قطعا در ورود به فاز ها نیاز هست، سپس product که به دلیل رویکرد افزایشی-تکرارشونده ی خود میسر می شود و درنهایت plan که هر فاز برای فاز بعد تدارک می بیند. همچنین EUP با دیسیپلین های مدیریتی در سطح سازمان، تنها به مدیریت ریسک ایجاد نرم افزار اکتفا نکرده و موارد سازمانی را نیز مدیریت ریسک می کند. بدین ترتیب EUP یکی از موفق ترین ها در مدیریت ریسک می باشد.

متدولوژی Fusion: متدولوژی Fusion در این زمینه جز ارائه چک لیست های خود نمی تواند ریسک را مدیریت کند. فعالیت Fusion نمی تواند پروتوتایپی ارائه دهد یا پلن خود را برحسب وظایفی ریسکی مدون کند. همچنین رویکرد افزایشی-تکرارشونده ای ندارد و به دخالت فعال کاربر بی توجه می باشد. ترخیص زودرس نباشد و بازنگری را در حد تکمیل مدل های یک فاز و گذار از یکی به دیگری در نظر می گیرد. در این بین اما Fusion راهکار مهمی ارائه داده است و آن ارائه دادن چک لیست هایی از فاز به فاز بعدی می باشد که برای بررسی سازگاری و کامل بودن مدل ها به کار می روند و بسیار هم دقیق هستند که به نوعی verification مستمر را مهیا می کند.

متدولوژی OOSE: این متدولوژی هم در تکمیل مدل های هر فاز خود، به شکل تکرارشونده ای عمل می کند اما در هر صورت فرآیندی افزایشی-تکرارشونده ندارد. با ارائه ی user interface design توجهی به دخالت فعال کاربر دارد که بازخورد را در گرفتن نیازمندی ها بگیرد و یک ساختار منطقی سطح بالا

می باشد که نمونه ای پروتوتایپ هرچند دورریختنی می باشد(در هر صورت در سنجش ریسک تاثیر خوبی می گذارد). تحلیلی برای تصمیم و طراحی پلن جهت ادامه دادن ندارد و همچنین تصریحی در اولویت بندی ریسک ها جهت برنامه ریزی آن ندارد. توجهی به ترخیص زودرس در شرح فرآیند آن یافت نمی شود. اما دقت خوبی در تست را تعریف می کند(همانطور که گفته شد) و در نهایت هم آزمون های خود را در سطوح مختلف ارائه می کند که validation را از همان اول می تواند ارائه دهد و تا آخر آن را مستمر ببخشد که کار مهمی در سنجش ریسک می باشد.

مقایسه OPM با EUP: در کل این زمینه EUP از همه جهت نقطه ی مثبت دارد. داشتن پروتوتایپ تکاملی، امکان سنجی بسیار دقیق تر با تحلیل مقدماتی، رویکرد افزایشی-تکرارشونده، بازنگری های تکراری، تست های مستمر بسیار موثرتر و منطقی تر و دخالت دادن کاربر از برتری های EUP هستند. البته دو متدولوژی ترخیص زودرس را توجه نکرده اند.

مقایسه OPM با Fusion: این دو متدولوژی پروتوتایپی را ارائه نمی کنند، تحلیل مقدماتی در OPM بیشتر به چشم می آید که نکته ی قوت آن به حساب می رود. هیچ کدام رویکرد افزایشی-تکرارشونده ای ندارند و به دخالت مستقیم فعالانه ی کاربر آن طور که نماینده ای سمت تیم ایجاد داشته باشند توجهی ندارند. در این بین اما OPM به پذیرش های نهایی کاربر اهمیت بیشتری می دهد.ترخیص زودرس و مرور و بازنگری فرآیند در آن ها کمتر به چشم می خورد. متدولوژی Fusion در verify کردن و OPM در validation اهتمام بیشتری دارند.

مقایسه OPM با OOSE: متدولوژی OOSE نقطه ی قوت خود را در ارائه اینترفیس کاربری در همان مراحل ابتدایی می داند که جهت کاهش ریسک پروتوتایپی مهیا می شود که در OPM چنین چیزی مشاهده نمی شود. هیچ کدام از متدولوژی ها رویکرد تحلیل مقدماتی درستی ندارند اما OPM در این زمینه بهتر

عمل می کند. متدولوژی ها رویکرد افزایشی-تکرارشونده ای نداشته و دخالت کاربر را فعال به عنوان عضوی از تیم ایجاد حساب نمی کنند، تنها OOSE با پروتوتایپ خود و OPM با چک لیست های خود، کمی از آن را پوشش می دهند و در نهایت ترخیص زودرس در هیچکدام تصریحی ندارد.

- **مدیریت پروژه**

برای مدیریت کردن پروژه، متدولوژی باید بتواند متذکر شود که فعالیت های planning و scheduling و همچنین controlling چه گونه تعبیه شده اند. به این معنا که آیا روابط پیش نیازی بین وظایف کاری پروژه مشخص شده باشد و برای هر تسکی زودترین و دیرترین زمان شروع و پایان دیده شده باشد. توجه به تقویم در محور زمان و تخصیص منابع مشخص باشد و در نهایت نظاره گر بر پیشبرد آن بود.

متدولوژی OPM: هیچ نشانه ای از مدیریت پروژه مانند طراحی و برنامه ریزی در OPM یافت نمی شود جز آن که در مراحل ابتدایی منابع مشخص می شوند. به تبع آن کارهای مدیریتی خاصی مربوط به کنترل کردن و مانیتور کردن هم دیده نمی شود.

متدولوژی EUP: از بهترین متدولوژی ها برای مدیریت پروژه، متدولوژی EUP می باشد. این متدولوژی با تعریف دیسیپلین های چتری ناظر بر فعالیت های مدیریتی خود مخصوصا project management گام بلندی در راستای مدیریت پروژه بر می دارد که این ویژگی خود را RUP حفظ کرده است. دقیقا کار های مدیریت پروژه که در بالا آمد در همین دیسپلین طی فاز ها حرکت نموده و کل چرخه ی عمر را می پوشاند. مشخص نمودن تسک های کاری به شکل پیش نیازی و زمانبندی آن ها، نگاشت آن ها به محور زمان(تقویم) و تخصیص منابع و در نهایت پایش پروژه براساس آن در این دیسیپلین انجام می شود.

متدولوژی Fusion: متدولوژی Fusion فاقد هر گونه مدیریت پروژه می باشد و تصریحی در آن انجام نگرفته است.

متدولوژی OOSE: متدولوژی OOSE هم نمی تواند به صورت دقیقی از آنچه که برای مدیریت پروژه در تعریف مد نظر ما هست، ارائه بدهد و در آن اشاره به زمان تصریح نشده است.

مقایسه OPM با EUP: متدولوژی EUP از هر نظر برتر است، چرا که تمام کار های مدیریت پروژه را دقیقا تعبیه کرده و هیچ کم و کاستی نمی گذارد در حالی که OPM جز مشخص کردن منابع، کار دیگری برای مدیریت ندارد.

مقایسه OPM با Fusion: این دو متدولوژی فعالیت خاصی برای مدیریت پروژه در نظر نگرفته اند جز اینکه OPM در مراحل ابتدایی خود به شناسایی منابع می پردازد.

مقایسه OPM با OOSE: این دو متدولوژی فعالیت خاصی برای مدیریت پروژه به گونه ای که با تعریف مدنظر همخوانی داشته باشد، در نظر نگرفته اند جز اینکه OPM در مراحل ابتدایی خود به شناسایی منابع می پردازد.

- **تضمین کیفیت**

برای ضمانت کردن کیفیت نرم افزار لازم هست تا فعالیت های مخصوص سنجش کیفیت و ارتقای کیفیت داخل متدولوژی تعبیه شده باشد. تکنیک هایی مطرح می شود تا به این نیازمندی متدولوژی فائق آمد که در زیر آن ها را بررسی خواهیم کرد.

متدولوژی OPM: البته OPM بازنگری فنی مکرری را در خود نمی بیند اما از تک مدل بودن خود به خوبی استفاده می کند تا رهگیری به نیازمندی ها در آن بسیار آسان باشد که از تکنیک های تضمین کیفیت می باشد. در نهایت این که از شرط های طراحی براساس قرارداد نمی تواند استفاده کند. البته OPM در آخرین

فاز خود در تلاش است تا با ارزیابی عملکرد، جلوی افت کیفیت را بگیرد اما در روند ایجاد نرم افزار کاربردی ندارد.

متدولوژی EUP: متدولوژی EUP از همه ی تکنیک ها برای برقراری تضمین کیفیت استفاده می کند. با برگزار کردن دوره ای مرور های فنی، چه ناظر بر محصول، چه پلن و چه فرآیند کمک زیادی به تضمین نهایی کیفیت می کند که این را هم مدیون رویکرد افزایشی-تکرارشونده ی خود می باشد. در مدل هایی که به کار می گیرد می تواند طراحی را بر اساس قرارداد انجام دهد و برای آن پیش شرط و پس شرط در نظر بگیرد. در نهایت آن که هرچند الزامی به usecase ندارد اما چه با استفاده از آن و چه با هر مدل نیازمندی دیگر در حد خوبی می تواند به نیازمندی ها رهگیری کند. چرا که مدل های توالی یا فعالیت به خوبی می توانند نشان دهند هر قطعه در کد مثلا چگونه نیازمندی را محقق می کنند. در نتیجه قطعات مختلف نرم افزار توجیه شده و کیفیت محصول تضمین می شود.

متدولوژی Fusion: متدلوژی Fusion از دو راه کمک گرفتن از روش های صوری و رسیدن به پایداری در مدل ها سعی در بالا بردن ضمانت کیفیت دارد. این متدولوژی در کل برای تضمین کیفیت خیلی راهکاری عملی نمی کند و تصریحی ندارد.

متدولوژی OOSE: متدولوژی OOSE به دلیل usecase-driven بودن خود به شکل عالی، رهگیری به نیازمندی را محقق می کند و در مدل های خود طراحی بر اساس قرارداد را پیشنهاد می دهد. هنگامی که متدولوژی به شکل پیشرفت براساس usecase باشد می تواند در هر جا فقط به نیازمندی نگاه کرده و در صدد تحقق آن با کمک مدل های قبلی بربیاید که به این شکل حتی کد و بخش های آن را می توان به نیازمندی ها trace کرد و متوجه شویم که نیازمندی ها درست محقق شده اند یا خیر. بدین شکل وجود هر چیزی توجیه داشته و کیفیت محصول به خوبی تضمین می شود.

مقایسه OPM با EUP: در این مقایسه EUP بسیار بهتر عمل می کند. چرا که از هر راه و تکنیکی استفاده می کند اما در مقابل OPM فقط می تواند نیازمندی خود را رهگیری کند که EUP نیز این کار را می کند اما الزامی به استفاده از usecase ندارد. خود OPM هم به دلیل تک مدل بودن چنین ویژگی ای دارد و الا مبتنی بر نیازمندی پیش نمی رود.

مقایسه OPM با Fusion: در این مقایسه OPM آنجایی که رهگیری به نیازمندی دارد برتر است ولی در استفاده از روش های فرمال، متدولوژی Fusion برتری دارد. هیچ کدام اما به بازنگری فنی مکرر توجهی ندارند.

مقایسه OPM با OOSE: در این بین متدولوژی OOSE به دلیل پیشبرد فعالیت مبتنی بر usecase می تواند به نیازمندی ها رهگیری کند ولی OPM به دلیل استفاده از یک مدل چنین قابلیتی دارد اما OOSE از روش های صوری نیز بهره می برد که قدرت آن را در برابر OPM نشان می دهد.

۴.۱.۱ بی درزی و هموار بودن انتقال

بی درزی[۴] و هموار بودن انتقال[۵] از معیار هایی هست متدولوژی را مجبور می کند که شیفت پارادایمی نداشته باشد و اطلاعات از یک مرحله به مرحله ی دیگر از بین نرود و مغفول نماند تا ناسازگاری بوجود نیاید. حال بعضی وقت ها درز و گذارنشدن هموار می تواند در فعالیت ها رخ بدهد و گاهی هم در زنجیره مدلسازی نمود پیدا کند. برای جلوگیری از رخداد هر کدام فنون مختلفی پیشنهاد شده است که به آن ها خواهیم پرداخت.

- **بی درزی**

متدولوژی OPM: در این متدولوژی بی درزی به شکل مطلوبی یافت می شود. دلیل آن هم تک‌ـمدله بودن فرآیند می باشد که تنها از OPD برای مدلسازی همه جنبه ها استفاده می شود و تنها محصول نموداری و غیر نرم افزاری می باشد. البته باید دقت داشت که قدرت زیادی نداشته و مدلی شلوغ و به هم تنیده ارائه می دهد که می تواند درز نداشتن را خدشه دار کند، چرا که بخش رفتاری از usecase به کد به درستی گذار ندارد و درز دیده می شود که مشکل بی درزی را کمی با خلل روبرو می کند. در این منظور تیم ایجاد از یک عنصر خالص غیر شیءگرا به عنصر کاملا شیءگرا پریده و عناصر میانی را ندارد. پس چون مدل های رفتاری ضعیفی دارد پس اندکی مخصوصا در تعاملات بین شیءای و حتی درون شیءای از خود ضعف نشان می دهد. در نهایت این که آن بخشی از درز که مربوط به رفتار می باشد و در آخر باید تعامل آبجکت ها را بتوان از آن معین کرد، مدلسازی ای خاصی ندارد و این خدشه را برای بی درزی بوجود خواهد آورد.

متدولوژی EUP: این متدولوژی دقیقا مانند RUP عمل کرده و درست هست که بی درزی را نشان می دهد اما به اصطلاح سکسکه دارد. این متدولوژی در

[۴] seamlessness
[۵] smoothness of transition

کل شیءگرا فکر می کند اما در جایی که نوعا از usecase به توالی پل می زند سکسکه را به همراه دارد. بدین منظور برای پوشاندن درز از activity کمک گرفته می شود که با ایجاد swim lane پلی درست شود تا بی درز بهبود پیدا کند. متدولوژی با بیان مفهوم مشترک شیء و یوزکیس به خوبی می تواند خود را بی درز جلوه دهد و در این میان سعی در پوشش انتقال از usecase به شیء دارد. اگر هم از usecase استفاده نکند، به هر صورت انتقال از دید dfd مثلا به دید شیء گرا یک درز می باشد که باعث مغفول ماندن اطلاعات می شود و در این زمینه نمی تواند بی درز باشد.

متدولوژی Fusion: متدولوژی Fusion نیز در این زمینه از خود بی درزی نشان می دهد چرا که انتقال از فاز ها و فعالیت ها در آن بدون پارادایم شیفت بوده و تنها ممکن است در انتقال فاز تحلیل به طراحی دچار از دست دادن اطلاعات شود که در نهایت نمی تواند گذار همواری را ممکن بسازد. بدین منظور Fusion با مطرح کردن مفهوم operation و مبتنی بودن بر آن در دید بین شیءای سعی در بی درز جلوه کردن دارد. می دانیم که operation معادل همان usecase ها بوده و صرفا از usecase ریزدانه تر می باشد. درنتیجه Fusion درصد بالایی از بی درزی را از خود نشان می دهد.

متدولوژی OOSE: متدولوژی OOSE هم با usecase-driven بودن خود، خود را بی درز نشان می دهد. بدین طریق با مطرح کردن مفهوم شیء، سعی دارد که پارادایم شیفت را حذف کند و تفاوتی در الگو های ادراک نگذارد. هر چند که درز را باید در انتقال از usecase که عنصری غیر شیءگرا می باشد به مفهوم شیء بپوشاند وگرنه در این زمینه ضعف دارد.

مقایسه OPM با EUP: در این مقایسه، متدولوژی OPM آن جا که تک_مدل هست برتری دارد و متدولوژی EUP آن جا که مختار است از usecase استفاده کند و برای رسیدن به مفهوم شیء، پل بزنند. اما ضعف OPM در آن جا می باشد که

مدلسازی رفتاری را به خوبی نمی تواند پشتیبانی کند و گذار درستی از نیازمندی به کد شیءگرا وجود ندارد و ضعف EUP زمانی است که از dfd استفاده شود و یا از مدل های میانی برای رسیدن به مدل توالی و رفتاری استفاده نشود.

مقایسه OPM با Fusion: متدولوژی Fusion مشکل OPM را ندارد(مدلسازی های رفتاری خوبی دارد) اما فقط ضعف خود را جایی نشان می دهد که می خواهد آبجکت_مدلی که در تحلیل به دست آورده را در فاز بعدی به کار بگیرد. در این راه، نمی تواند آن را کامل کند بلکه سعی دارد اطلاعات آن را پخش کرده و صرفا استفاده کند. این البته ارتباط مستقیمی با درز ندارد ولی از آنجا که بالقوه باعث پیش آمدن ناسازگاری می شود ممکن است بی درزی Fusion هم بالقوه در خطر باشد.

مقایسه OPM با OOSE: متدولوژی OOSE هم مانند EUP در خطر بی درزی هست اما به دلیل قراردادن usecase در مسند همه فازهای خود، سعی در از دست ندادن اطلاعات دارد و در انتقال از فاز به فاز با کمک گرفتن از مدل های قبلی در مدل های شیءگرایی می کوشد تا با نیازمندی مطابقت داشته باشد وبی درز باشد. این نقطه ی قوتی در برابر OPM هست که مدلسازی رفتاری را برنامه ای ندارد.

- **هموار بودن انتقال**

متدولوژی OPM: همانطور که بیان شد، گذار درستی در بین محصولات OPM ممکن است وجود نداشته باشد؛ چرا که از تک_مدل خود به کد شیءگرا جهش می کند و محصولی کاملا جدید بدون درنظر گرفتن مراحل میانی و مدل های رفتاری تولید می کند که باعث ایجاد ناهمواری هست. متدولوژی OPM البته فازها و زیرفازهای پیوسته ای دارد که درصد بالایی از هموار بودن را برای تیم اجرایی به همراه داشته و از این سبب پرش بزرگی در گام ها مشاهده نمی شود

اما به دلیل ساخت کد شیءگرا که محصولی جدید می باشد از روی نموداری با مدلسازی رفتاری ضعیف، گذار هموار را تضعیف می کند.

متدولوژی EUP: متدولوژی EUP بسیار هموار و خوب عمل می کند. دلیل آن هم خاصیت افزایشی-تکرارشونده می باشد که گذار را بین فاز ها و واحد های کاری هموار کرده تا محصول جدیدی به یکباره شکل نگیرد و کار ها کوچک کوچک برداشته شوند تا پیشرفت در کار ها پیوسته انجام شود. البته نکته ی منفی در این متدولوژی بیش از حد پیچیده بودن آن است که در صورت عدم تسلط سبب آسیب رسیدن به تعدد انواع مدل ها می شود که خاصیت منفی بالقوه ای است.

متدولوژی Fusion: از مهم ترین نقاط قوت Fusion گذار تقریبی هموار از فاز به فاز و گام به گام می باشد؛ چرا که ساخت مدل های خیلی سنگین جدیدی در آن برای هر فاز وجود ندارد اما مدل های جدیدی در فاز طراحی به چشم می خورد که نمی تواند ارتباط درستی با مدل های تحلیل در آن یافت و این گذار هموار را کمی دچار مشکل می کند. البته بی درزی خوب آن تقریبا می تواند این مساله را پوشش دهد و این که توالی منطقی در نمودار های آن موجود است اما در هر صورت گذار کاملا هموار در بین فاز ها دیده نمی شود.

متدولوژی OOSE: متدولوژی OOSE بسیار خوب می تواند از خود گذار هموار را نشان دهد. این متدولوژی مدل ها و محصولات خود را با کمک گرفتن از محصولات قبلی و تکامل آن ها براساس usecase می بیند که باعث می شود محصولات جدید سنگین با مفهومی متفاوت تولید نشوند و در این میان بدین طریق متوجه هستیم که متدولوژی OOSE مجموعه محصولات خود را در فاز ها به فرمت کاملا جدیدی عوض نمی کند.

مقایسه OPM با EUP: متدولوژی OPM در گذار هموار از خود ضعف نشان داده و نمی تواند گذار درستی را از مدلسازی به پیاده سازی داشته باشد. در عین

حال EUP مشکل خود را در استفاده ی احتمالی از dfd می بیند در غیر این صورت با به کارگیری از خط شنا می تواند انتقال هموار را از مفهوم یوزکیس به شیء فراهم کند.

مقایسه OPM با Fusion: متدولوژی Fusion در کل گذار هموار از نقاط مثبت آن می باشد. مدل های آن سنگین نبوده اما نقطه ی منفی در کاملا جدید بودن آن ها در فاز طراحی است. در کل به دلیل بی درزی، از فاز تحلیل به طراحی گذار را می توان در حد خوبی هموار فرض کرد. در این مقایسه OPM نمی تواند چنین گذار همواری را مهیا کند و صرفا به خاطر تک مدل خود بی درز هست ولی نمی تواند گذار به implementing را هموار جلوه دهد، چرا که فرآیند نمی تواند مدلسازی رفتاری درستی را ارائه دهد.

مقایسه OPM با OOSE: متدولوژی OOSE در اینجا برتری خوبی دارد؛ از این جهت که مدل ها محصولات(نرم افزاری و غیر نرم افزاری) خود را به تدریج کامل کرده و در گذار از فاز ها از نیازمندی هم بهره می گیرد و توجه خود را به آن معطوف می کند.

۵.۱.۱ مبتنی بر نیازمندی وظیفه ای و غیر وظیفه ای

برای تحقق به این نیازمندی که در سال های اخیر برای یک متدولوژی بدیهی شده است باید ویژگی های زیر در متدولوژی تعبیه شده باشد:

اولا که نیازمندی ها در ابتدای فرآیند و شروع متدولوژی باید ثبت و استخراج شوند.

دوما به تنهایی و به صورت مستقل مدل بشوند و این مستندسازی باید جداگانه و به صورت خاصی اتفاق بیافتد؛ نباید به گونه ای باشد که نیازمندی ها را مجبور باشیم از نموداری که ساختار سیستم را مثلا نشان می دهد برداشت کنیم. نیازمندی باید زود و به صورت خاص مدل شود.

سوما که از آنها برای مبنا در هر واحد کاری بعدی اسفاده کنیم.

حال با این تفاسیر، متدولوژی های زیر را از جنبه این معیار بررسی می کنیم:

- متدولوژی OPM: این متدولوژی فقط از یک مدل بهره می برد که آن هم OPD است و می دانیم که هیچ گونه مستندی را برای ثبت نیازمندی ها به صورت جداگانه ندارد. یعنی مشخص شدن scope سیستم در سطح بالا و نیازمندی ها، ثبت می شوند اما نمی تواند در یکی از مدل هایی که متدولوژی معرفی می کند به طور خاص ذخیره شوند. در نهایت آن که نیازمندی های بدست آمده را استفاده می کند و از روی آن ها مدل OPD را تکمیل می دهد ولی نمی تواند رهگیری درستی به کد داشته باشد. در کل مبنای OPD آن نیازمندی است ولی usecase-driven نمی باشد. توجه به نیازمندی های غیروظیفه ای در این متدولوژی بالا می باشد و نگهداری به عنوان مهم ترین نیاز غیر وظیفه ای را پشتیبانی می کند.

- متدولوژی EUP: متدولوژی EUP نیازمندی ها را در حد بسیار بالا و دقیقی در می آورد به طوری که در فاز دوم آن تا ۸۰ درصد و سپس در فاز construction خود تا ۱۰۰ درصد آن را معین می کند و رویکرد افزایشی ـ تکرار شونده ی آن این

امکان را می دهد که نیازمندی را به خوبی ثبت و ضبط کنیم. متدولوژی EUP اصراری به استفاده از usecase در مدلسازی نیازمندی ها ندارد و بنابراین نمی توان آن را لزوما usecase-driven دانست. اما نکته ی مثبت آن این هست که در هر صورت نیازمندی را به طریقی در یک مدل جدا به صورت خاص می آورد و در نهایت این که آن را برای فعالیت های بعدی مبنا قرار می دهد. متدولوژی EUP نسبت به نیازمندی نگهداری کم توجه بوده است.

- متدولوژی Fusion: متدولوژی Fusion هیچ گونه اهتمامی برای اینکه در ابتدای فرآیند و شروع متدولوژی، نیازمندی را استخراج کند ندارد و آن را ورودی خود می داند. این گونه فرض می کند که قبلا نیازمندی ها را گروه دیگری طی یک روشی استخراج کرده اند و حالا به عنوان ورودی برای متدولوژی در نظر گرفته می شوند. پس هیچ مدلسازی خاصی هم برای آن نمی توان متصور شد و فقط می توان آن را مبنای تعریف operation ها در تحلیل دانست. هیچ نیازمندی غیروظیفه ای را Fusion نمی تواند مبنا قرار دهد.

- متدولوژی OOSE: متدولوژی OOSE هم در ابتدای کار فرآیند، تحلیل نیازمندی ها را انجام داده و آن ها را می شناسد و در usecase مدلسازی می کند. در نهایت آن که usecase را مبنای هر مدلی در هر فاز و مرحله ای می داند و از آن برای کنترل بهتر و رهگیری به نیازمندی ها استفاده می کند. اینگونه OOSE دقیقا این معیار را پاس می کند و هیچ نقصی ندارد. متدولوژی OOSE با انجام انواع تست ها در سطح های مختلف سعی در برآورده سازی نیازمندی های غیروظیفه ای دارد و برای نگهداری البته تمهیدی نیاندیشیده است.

- مقایسه OPM با EUP: در این زمینه متدولوژی EUP نقص خاصی در برابر OPM ندارد جز آنکه مبتنی بر نیازمندی نگه داری نیست و OPM مشکلش آن هست که مدل خاصی را به صورت جدا برای نیازمندی ها ارائه نمی دهد در حالی که دست

EUP در این قسمت باز هست. سرانجام آن که هر دو نیازمندی ها را کشف می کنند ولی می توان گفت EUP به خاطر بهره مندی از ویژگی افزایشی ـ تکرارشونده ی خود خیلی بهتر در این زمینه می تواند نیازمندی ها را کشف کند و هر دوی آن ها از نیازمندی وظیفه ای برای ادامه کار استفاده خواهند کرد.

- مقایسه OPM با Fusion: متدولوژی OPM از این جهت برتر است که شناسایی scope سیستم را تصریح می کند؛ اما Fusion آن را پیشفرض ورودی به شروع متدولوژی می داند. البته هیچکدام مدلسازی خاصی را برای ثبت نیازمندی ها تصریح نکرده اند اما هر دو آن را مبنای تحلیل و مدلسازی بعدی می دانند.

- مقایسه OPM با OOSE: متدولوژی OOSE وجه مثبتی که دارد این هست که مفهوم usecase را معرفی و نیازمندی های خود را در آن به شکل جداگانه ثبت می کند. سپس به صورت usecase-driven عمل کرده و آن را مبنای هر چیزی قرار می دهد. هر دو متدولوژی ها البته به تحلیل مقدماتی و کشف قلمرو سیستم با استخراج نیازمندی ها در ابتدای کار اهتمام می ورزند.

۶.۱.۱ قابل آزمون، ملموس و قابل رهگیری به نیازمندی ها

معیار ششمی که به آن می پردازیم، شامل بر ۳ ویژگی می باشد به عنوان های: آزمون پذیری متدولوژی[۶]، ملموس بودن متدولوژی[۷] و قابلیت رهگیری به نیازمندی ها[۸] که هر یک به مفاهیم جداگانه ای اشاره دارند اما عمدتا ناظر بر محصولات یک متدولوژی هستند.

- **آزمون پذیری**

به طور خلاصه متدولوژی ای آزمون پذیر است که به سهولت بتوان متدولوژی و با تکیه ی بر محصولات آن، بدون ابهام آزمون را طراحی و اجرا کرد.

نکته در این هست که برای تحقق چنین هدفی، متدولوژی بهتر است تا پیچیدگی محصولات خود را کاهش دهد. تعداد آنان حتی الامکان کم بوده و ساده به طوری که پیچیدگی فهم در حداقل ممکن بوده تا قابل فهم باشد. وابستگی های بین محصولات باید کم بوده و باید وابستگی های حداقلی بین آن ها به خوبی تعریف شده باشد. به عنوان مثال بهتر است تا وابستگی شدید دو نمودار را که یکی با نمودار دیگر، رابطه ی isomorphism دارند جایی در متدولوژی نداشته باشد؛ چرا که در غیر این صورت یکپارچگی و جامع بودن محصولات به خطر افتاده و تناقض می تواند به آسانی پیدا شود. تغییر در یکی، باعث انتشار تغییر می شود که این موارد به خودی خود، قابلیت آزمون پذیری را کاهش می دهد.

متدولوژی OPM: با توجه به تعریف بالا، می دانیم که OPM یکی از پیچیده ترین مدل ها را دارد، پس مورد آزمون قرار گرفتن محصول آن بسیار سخت می نماید. زبان مدلسازی OPD سعی دارد که همه ی جنبه های موجود در یک سیستم را اعم از رفتاری، ساختاری و وظیفه مندی تنها داخل یک نمودار بیان

[۶] Testability
[۷] Tangibility
[۸] Traceability to requirements

کند که این خود نوعا باعث پیچیدگی است. اما از آن سو به علت وجود تنها یک مدل در OPM تعداد نمودارهای متدولوژی کم و وابستگی بین محصولات در حداقل ممکن است و احتمال بروز تناقض بسیار سخت خواهد بود.

متدولوژی EUP: متدولوژی EUP سعی دارد که از UML استفاده کند اما در این راه اصرار یا اجباری ندارد. محصولات آن به نسبت زیاد بوده و وابستگی هایی زیادتری دارند. دقت داریم که با توجه به نگاه سازمانی در آن، انتشار تغییرات در آن گسترده بوده و ممکن است تصمیم گیری های کلان بر روی مدل ها و محصولات داخل پروژه اثرگذار باشد که همه ی اینها سبب کاهش آزمون پذیری می باشد. این متدولوژی حتی با ارائه مدل های سازمانی بر پیچیدگی محصولات RUP می افزاید.

متدولوژی Fusion: متدولوژی Fusion وابستگی بین محصولات زیادی دارد؛ هر چند که فرآیند به خوبی می تواند وابستگی ها را به نسبت کنترل کند. در Fu-sion تعداد مدل ها تقریبا بالاست(طوری که تعداد حداقل ۷ نوع مدل نموداری در آن دیده می شود). در نهایت اما چیزی که باعث آزمون پذیری بهتر آن می شود، ساده بودن محصولات آن است که درک و فهم آن ها سخت نبوده و به خوبی شرح داده شده اند.

متدولوژی OOSE: در این متدولوژی از مدل های بسیار شبیه UML استفاده می شود و تعداد آنها تقریبا بالا می باشد. منتها فهم سختی ندارند و در حد زیادی پیچیده نیستند. بدیهی است که وابستگی بین آن ها موجود بوده و بالقوه دارای پتانسیل تناقض می باشد اما تشریح وابستگی بین مدل ها به مانند UML خوب انجام شده است.

مقایسه OPM با EUP: برتری OPM در استفاده از تک مدل می باشد و وابستگی حداقلی بین محصولات دارد که آزمون پذیری را بهتر جلوه می دهد اما OPM در مقابل EUP محصول بسیار پیچیده تری دارد. پس از این نظر آزمون پذیری

ضعیف تری دارد؛ چرا که سعی دارد همه جنبه ها را در یک مدل ببینید که درنتیجه تمرکز و تست را دچار خدشه می کند.

مقایسه OPM با Fusion: البته Fusion نمی تواند مانند OPM تعداد مدل های کمی بدهد اما پیچیدگی مدل های آن زیاد نیست و البته به خوبی می تواند وابستگی ها را شرح دهد و از ظهور غیرسازگاری جلوگیری کند. بدین ترتیب Fusion را می توان از نظر آزمون پذیری برتر از OPM دانست.

مقایسه OPM با OOSE: متدولوژی OOSE تعداد مدل های زیادی دارد و از این بابت در برابر OPM ضعیف عمل می کند اما پیچیدگی بسیار کمی در فهم مدل های آن وجود دارد و تمرکز ها در نمودار های مختلف به خوبی پخش شده اند. این می تواند آزمون پذیری را برای OOSE بالا ببرد. البته متدولوژی OPM از وابستگی حداقلی بین محصولات خود برخودار است که نکته ی مثبتی در برابر OOSE می باشد.

- **ملموس بودن**

ملموس بودن برای یک متدولوژی آن است که محصولات متدولوژی از دید استفاده کننده ی محصول، قابل فهم و موجه باشند. طبق تعریف بدیهی است که تا حد زیادی به بیننده بستگی دارد. حال بیننده می تواند کاربر باشد و یا توسعه دهنده که در هر دو صورت محصولات خاصی متناظر می شوند. اگر از دید یک بیننده، محصولی ملموس نباشد، دلیل وجود آن مشخص و موجه نیست، لذا در ساخت آن ها دقتی نشده و استفاده ی بعدی نیز صورت نمی پذیرد که به نوعی اتلاف منابع به حساب می آید. مثلا برای کاربر، محصولات قابل اجرا مانند پروتوتایپ ها و یا محصولاتی که برای او معنی داشته باشند و طبق سواد او ساخته می شود مثل usecase ها، ملموس بودن خوبی را در محصولات متدولوژی می رساند. برای توسعه دهنده نیز مصنوعاتی که خیلی واضح در چرخه عمر و واحد

های کاری فرآیند، کاربرد دارند و حضور آن ها موجه هست، از ملموس بودن محصولات متدولوژی خبر می دهد.

متدولوژی OPM: متدولوژی OPM که از OPD برای مدلسازی خود استفاده می کند، توصیه دارد که از OPL استفاده شود که در واقع مستند متنی همان نمودار می باشد که از این بابت محصولی ساده است که به زبان عادی نوشته شده است و محصولی است که می تواند برای کاربر معنا داشته باشد و در سطح سواد او باشد. ملموس بودن آن به خاطر این نیز هست که پیچیدگی نمودار را کاهش داد و از این بابت برای ایجاد کننده ی محصول هم قابل توجیه هست(هرچند اعتراف بدی است اما حضور آن به خوبی می تواند مساله را شفاف کند). متدولوژی در جای دیگری برای ایجاد کنندگان محصولی درنظر نگرفته است.

متدولوژی EUP: برای کاربر، محصولات قابل اجرا ساخته شده که می تواند پروتوتایپ های تکاملی و یا دورریختنی باشد و در این نظر، توجیه بسیار خوبی برای کاربر دارد و درجه بالای ملموس بودن خود را نشان می دهد. از طرفی با بهره گیری از مواردی مثل usecase می تواند مدل های نیازمندی را تا حد خوبی برای سطح دانش کاربر تعریف کند و از این بابت ملموس عمل کند. محصولی که برای کاربر و تیم ایجاد بسیار شفاف هست همان user-manual هایی است که EUP به ساخت آن حتی در فاز های اولیه تاکید می کند. با وجود چنین محصولی، کاربر می تواند در سطح دانش خود نظر خود را اعمال کرده و بازخوردی به تیم ایجاد انتقال دهند و همچنین تیم ایجاد می تواند برای آینده ی خود مثل فاز construction و بعدی ها، از آن بهره ببرد؛ پس حضور بسیار واضحی دارد و برای طرفین ملموس است. مصنوعات دیگری هم که در طول چرخه ی عمر ساخته می شوند، اگر از dfd استفاده نشود، به طور کلی همان UML های RUP بوده که حضور هر یک قابل توجیه هست. نمودار dfd اما برای استفاده نیاز به برقراری ارتباط قوی با بقیه ی محصولات دارد(تا بتوان از آن

استفاده کرد و اتلاف منابع نباشد) که در این صورت می تواند همچنان ملموس بوده و حضور پررنگی در فرآیند ایجاد نرم افزار داشته باشد.

متدولوژی Fusion: متدولوژی Fusion محصولاتش را خیلی خوب ملموس جلوه می دهد. هر نوع محصولی که ارائه می دهد برای تیم ایجاد، حضوری قابل توجیه دارد ولی در این راه از دید کاربر نمی تواند محصولی را ارائه کند که خدشه ای در ملموس بودن بقیه ی محصولات آن ندارد.

متدولوژی OOSE: در متدولوژی OOSE وجود محصولی مانند usecase بسیار خوب می تواند از دید کاربر شفافیت محصول را نشان دهد؛ چرا که در سطح دانش کاربر بوده و او به راحتی می تواند نظر خود را ارائه دهد. بدین ترتیب OOSE در سطح بالای خود که هنوز درگیر کشف چیستی سیستم می باشد، با رابط کاربری لوجیکالی که فراهم می کند می تواند محصولی قابل ملموس به کاربر ارائه دهد که حضور پررنگی در کشف نیازمندی های سیستم دارد. سپس باقی محصولاتی که می سازد همگی برای تیم ایجاد قابل توجیه بوده و بسیار شبیه به UML عمل می کند و نمی توان محصولی را تصور کرد که از آن نتوان استفاده کرد.

مقایسه OPM با EUP: در این مقایسه برتری EUP در استفاده از پروتوتایپ ها و همچنین استفاده ی احتمالی از usecase می باشد که محصولات توجیه پذیری برای کاربر می باشند. متدولوژی EUP همچنین با ساختن user-manual ها از OPM جلوتر می باشد.

مقایسه OPM با Fusion: البته متدولوژی Fusion در ملموس بودن محصولات خود کامل عمل می کند ولی OPM با در نظر گرفتن OPL می تواند محصولی شفاف را برای کاربر مهیا می کند درحالی که متدولوژی Fusion از دید کاربر محصولی شفاف ندارد.

مقایسه OPM با OOSE: مندولوژی OOSE هم با استفاده از usecase و پروتوتایپ ها و محصولات نموداری منطقی(که شباهت زیادی با UML دارند)، در طول فرآیند تولید نرم افزار هم از دید کاربر و هم از دید تیم ایجاد، محصولات ملموسی تولید می کند اما OPM تنها می تواند با OPL ملموس بودن محصول را برای کاربر داشته باشد. خود نمودار OPD هم اغلب پیچیده شده و نمی توان خیلی سرراست از آن در تولید کد استفاده کرد.

- **قابل رهگیری به نیازمندی ها**

در نهایت قابل رهگیر بودن به نیازمندی ها به این معنا می باشد که محصولات متدولوژی را به یک نیازمندی خاص یا مجموعه ای از آن ها رهگیری کرد.
بدین ترتیب مصنوعات باید مستقیم یا غیرمستقیم، پیاده سازی نیازمندی ها باشد و یا از طریق سناریوهای ارزیابی مبتنی بر نیازمندی بشود این تناظر را نشان داد. بدین معنا که از سناریوهایی استفاده می شود تا تحقق نیازمندی را بسنجند و سپس سناریو را روی مصنوعات متدولوژی اجرا می کنند و مشخص می شود که آیا نیازمندی محقق شده است یا خیر و اگر شده است کدام مصنوع درگیر در تحقق نیازمندی بوده است و اینگونه تناظر را بدست می آورند.

متدولوژی OPM: متدولوژی OPM چون تک_مدل هست، در کل می تواند به نیازمندی ها رهگیری کند، چرا که نیاز به مراجعات غیرمستقیم از قطعه ها به نیازمندی ها نیست و ارتباط قطعه های مختلف سیستم نرم افزاری با نیازمندی ها بلافصل می باشد. اما قدرت OPD به اندازه کافی نیست و مدلسازی رفتاری در آن ضعیف می باشد؛ در نتیجه رهگیری به نیازمندی ها دچار مشکل می شود. اگر دقت داشته باشیم متوجه هستیم که هنگامی که نمی توان بخشی از رفتار را که نهایتا با تعامل آبجکت ها مشخص کرد، رهگیری به نیازمندی ها خدشه دار می شود. متدولوژی OPM همچنین سناریو های تحقق نیازمندی را در خود پوشش نمی دهد.

متدولوژی EUP: در اینجا رهگیری به نیازمندی ها به صورت خوبی انجام می شود؛ چرا که خود را مبتنی بر نیازمندی نشان می دهد و رویکرد افزایشی- تکرارشونده ی آن کمک می کند تا بازنگری مکرر فنی مصنوعات به صورت مستقیم یا حتی غیرمستقیم انجام شده و چک شود که پیاده سازی نیازمندی ها چگونه محقق شده است. البته EUP صحبتی از سناریو های قابل اجرا جهت رهگیری به نیازمندی ها نمی کند اما می تواند از آن با نمودار های UML بهره مند شود.

متدولوژی Fusion: در متدولوژی Fusion رهگیری به نیازمندی ها از طریق سناریو های استفاده از سیستم در transaction scenario تعریف شده است که این سناریوها مبنای تعریف نیازمندی هستند. بدین مضمون، زمانی که op- eration ها محقق می شوند بررسی می شود که آیا transaction scenario بدرستی اجرا خواهند شد یا خیر.

متدولوژی OOSE: متدولوژی OOSE صددرصد رهگیری به نیازمندی ها را دارد. چرا که usecase-driven می باشد و در هر مرحله ای از اجرای فرآیند نگاهی به نیازمندی ها دارد که آن را مبنای همه چیز می داند. پس این ویژگی باعث می شود بتوان هر قسمی از سیستم را به صورت مستقیم به نیازمندی ها نگاشت داد.

مقایسه OPM با EUP: متدولوژی OPM آنجایی که تک-مدل هست می تواند رهگیری را به خوبی نشان دهد اما زمانی که می بیند نمی تواند مدلسازی رفتاری درستی از خودش نشان دهد، در مقابل متدولوژی EUP ضعیف عمل می کند. متدولوژی EUP مبتنی بر نیازمندی بوده و به دلیل افزایشی-تکرارشونده می تواند رهگیری را نیازمندی ها بهبود ببخشید. چرا که این امکان را دارد تا به مرور نیازمندی را کامل کرده و بخش های فنی را مرور می کند.

مقایسه OPM با Fusion: متدولوژی Fusion می تواند با تعریف سناریوهای

تراکنشی در راستای رهگیری به نیازمندی ها عمل کند ولی OPM چنین مدلسازی رفتاری را از خود نشان می دهد که برتری Fusion را نشان می دهد. از آن طرف OPM با تنها مدل خود رهگیری به نیازمندی ها را بهبود می بخشد اما Fusion قادر به ارائه چنین چیزی نیست.

مقایسه OPM با OOSE: متدولوژی OOSE در بهترین شکل رهگیری قرار دارد. نقطه ی قوت آن تعریف usecase و usecase-driven بودن آن می باشد که OPM از آن ناتوان هست.

۷.۱.۱ دخالت فعال کاربر

هفتمین معیار، یکی از تکنینک های کمک کننده به مدیریت ریسک بود و باعث خلق اعتبارسنجی مستمر نیز می شد که تاثیر مستقیم مثبتی بر تضمین کیفیت داشت و حال تبدیل به یک معیار برای ارزیابی و مقایسه ی متدولوژی ها شده است. چرا که به قدری مهم است که عدم رضایت مشتری را تا حد مطلوبی از بین برده و طرز نگرش در شکاف "ما" و "آنها" را از بین می برد و تهدیدی به نام مشتری را تبدیل به فرصت می کند.

دو روشی که موثر شناخته شده است تا این معیار پوشش داده شود این است که:

۱ـ نماینده ی کاربر، عضوی از تیم ایجاد باشد.

۲ـ جلسات برنامه ریزی و مرور و بازنگری با شرکت کاربران و نخبگان حوزه به صورت دوره ای برگزار شود.

با این حال می خواهیم این معیار را برای متدولوژی های زیر بررسی کرده و آن ها را با یکدیگر مقایسه کنیم:

- متدولوژی OPM: در این متدولوژی هیچ تصریحی برای دخالت فعال کاربر ذکر نشده است. نه حضوری از نماینده ی کاربر در تیم ایجاد می باشد و نه تدبیری برای جلسات برنامه ریزی و مرور و بازنگری با شرکت کاربران و نخبگان حوزه اندیشیده است. تنها در تست نهایی توجه به پذیرش کاربر شده است که نمی تواند به طور کلی این معیار را پوشش دهد.

- متدولوژی EUP: متدولوژی EUP در همان فاز یا فازهای اول مانند RUP عمل کرده و با ایجاد پروتوتایپ های چه دورریختنی و چه تکاملی سعی در بازخورد گرفتن از کاربر دارد و او را به صورت فعال دخالت می دهد. یکی از مهم ترین پروتوتایپ های تکاملی هم executable architecture می باشد که دقیقا یکی از کاربرد های آن برای نمایش به کاربر می باشد. در گام های وسط(یا حتی

اول) می توان به فکر ساخت user-manual ها برای سیستم نرم افزاری بود که این خود می تواند با تعامل با کاربر بوده و نظر اون نیز تعیین کننده باشد. مهم تر از همه ی این ها آن که می تواند ویژگی خود را از RUP حفظ کرده و قرارداد دومی با کاربر در انتهای فاز elaboration بسته شود که این نشان از مدیریت کلان پروژه و مشخص کردن بسیار دقیق طرح می باشد و در این راستا، کاربر در اواسط کار، درگیر شده و از اخبار و جریانات آتی با خبر می شود و می تواند نظر خود را اعمال کرده و یا تصمیمات دیگری بگیرد که این خود نوعی تشویق کاربر به دخالت فعال است. متدولوژی EUP از دومین روش موثر دخالت کاربر نیز استفاده می کند که بدین سبب بخشی از کار خود را به جلسات مکرر بازنگری و مدیریت اختصاص داده و از نظرات متخصصین می تواند بهره مند شود.

- متدولوژی Fusion: متدولوژی Fusion هیچ تصریحی در این زمینه ندارد و نمی تواند نقشی را برای کاربر در هنگام ایجاد سیستم متصور شود و هیچ کدام از دو روش موثر بالا را توجه نکرده است.

- متدولوژی OOSE: متدولوژی OOSE به هنگامی که در تحلیل نیازمندی های خود در فاز اول می باشد مهم ترین کاری که در جریان شناسایی scope سیستم می کند آن است که نظر کاربر را بر روی یک اینترفیس کاربری در سطح بالای منطق می پرسد و از آن بازخورد دریافت می کند. بدین جهت کاربر به نوعی در روند توسعه سهیم می باشد. البته وجود جلسات مکرری که با حضور متخصصین حوزه یا کاربران باشد در آن دیده نشده است اما با ساخت پروتوتایپی که غیروابسته به پیاده سازی است، برای دید بهتر کاربر از سیستم، کاربر را دخالت داده و نظر او را جویا می شود.

- مقایسه OPM با EUP: متدولوژی EUP از این جهت برتری دارد که هم می تواند با روال افزایشی-تکرارشونده ی خود، جلسات مکرر مرور و مدیریت را اعمال

کند و هم این که در جای جای مختلف فازها و مراحل خود(کما این که در بالا آمد) نظر و بازخورد کاربر را بداند و او را جزوی از تیم ایجاد بداند. متدولوژی OPM اما هیچ توجهی جز جویا شدن نظر کاربر در مورد تست نهایی به این موارد ندارد.

- مقایسه OPM با Fusion: متدولوژی OPM برتری خود را در گرفتن نظر کاربر به هنگام تست پذیرش سیستم می بیند و Fusion اقدام موثری انجام نداده است.
- مقایسه OPM با OOSE: متدولوژی OOSE بسیار خوب می تواند در دخالت دادن کاربر در مرحله اولیه، به کشف بهتر سیستم بپردازد و با تهیه پروتوتایپ ریسک عدم رضایت او را کاهش دهد. این متدولوژی هم تست های بسیار خوبی در خود می بینید که در سطوح مختلفی انجام می شود و در آن می تواند از کاربر هم نظر خواهی کند. اما OPM در acceptance testing خود دقیقا در محیط کاربر حاضر شده و تست در آنجا انجام می شود و نظر کاربر تعیین کننده خواهد بود که خیلی البته نمی تواند برآورده کننده معیار باشد.

۸.۱.۱ قابل اجرا بودن و استفاده کارآ

قابل اجرا بودن متدولوژی[۹] و به شکل کارآ استفاده کردن یا کارآیی متدولوژی[۱۰] در هشتمین معیار برای ارزیابی متدولوژی بیان می شود. این دو ویژگی تا حدی مهم هستندکه به صورت معیار جداگانه ای برای متدولوژی درآمده که تعریف های مطابق زیر دارد:

می گوییم متدولوژی قابل اجرا می باشد زمانی که بتوان آن را اعمال کرد؛ یعنی بتوان فرآیندی را که ذکر می کند، به کار بست. این صحبت برای محقق شدن به ۲ ویژگی وابستگی دارد. اولا آن که فرآیند خیلی پیچیده نباشد؛ چرا که پیچیدگی فرآیند نهایتا قابل اجرا نخواهد بود و یا به تمامیت قابل اعمال نیستند. مانند RUP که در حال حاضر بر همگان مشخص است که فرآیند پیچیده ی آن، قابلیت اجرایی آن را مختل می کند. سنگین بودن RUP در نهایت به ظهور RMC منجر شد.

دوما که به ماهیت و طبیعت پروژه هدف خیلی بستگی دارد که چه چیزی را هدف نهایی قرار داده است. بدیهی است که متدولوژی ای که برای موارد پروژه ای ساده تعریف شده است، نمی تواند در فرآیندی در خور اعمال برای پروژه های سنگین باشد و بالعکس. این مورد را نمی توان از ذات یک متدولوژی تحقق بخشید.

اما می گوییم یک متدولوژی به شکل کارآ استفاده می شود و کارآیی دارد زمانی که ابتدا قابل اجرا بر روی پروژه باشد و سپس چه میزان کارآیی را با خود به همراه می آورد تا هدف پروژه بدست آورده شود. بدین معنا که متدولوژی در حوزه کاربرد خودش آیا می توان به شکل موثر از آن استفاده کرد یا خیر که برای احقاق آن ۵ ویژگی ثبت می شود.

اولا که متدولوژی بدیهی است اگر اجزا پیچیده ای داشته باشد، کارآیی کاهش می یابد. سه تا P و زبان مدلسازی باید در بهینه ترین پیچیدگی ممکن شکل بگیرند.

دوما فعالیت هایی که حواس ایجادکنندگان نرم افزار را از فعالیت اصلی دور می

---

[۹] practicability

[۱۰] practicality

کند و یا جزییات بدون مورد و غیرضروری را منتقل می کند، حذف کرد و توجهات را به نکات اصلی معطوف کرد تا کارآیی بالاتر برود. این کار با جلسات مدیریت تیم، مدلهایی مبتنی بر نیازمندی ها و ایجاد معماری سیستم امکان پذیر خواهد بود.

سوما از وابستگی به تکنیک های خطا مثلا لزوم استفاده از همه ی نمودار های UML و یا پارادایم شیفت ناشی از حرکت های درز دار پرهیز کرد تا کارآیی افت نکند.

چهارما آنکه متدولوژی اگر به ابزار یا تکنولوژی خاصی وابسته باشد (مانند XP که برای اعمال نیاز به ابزار خاص خود دارد) تاثیر مستقیم روی کاهش کارآیی دارد.

پنجما هم اینکه اگر در متدولوژی، استراتژی ها فعالیت های مدیریتی برای پروژه پشتیبانی و پیش بینی نشده باشد و ناکافی باشد، تاثیر منفی بر روی کارآیی خواهد داشت.

حال با این تعریف اولیه به سراغ مقایسه متدولوژی های زیر می رویم:

- متدولوژی OPM: متدولوژی OPM پیچیدگی فرآیندی خود را آن جا نشان می دهد که برای هر یک از گام های خود، جزییات زیادی را مطرح نمی کند(پیچیدگی نه در تعریف، برای به کارگیری و اجرا) و تیم اجراکننده ی متدولوژی را سردرگم می گذارد. بدین ترتیب شاید نتوان به تمامیت از آن چه که متدولوژی و واضع آن مدنظرش بوده است استفاده کرد؛ چرا که برای پیاده سازی مراحل فازها مقدار زیادی از جزئیات موجود نیست. اما برای استفاده ی کارآ توجه داریم که همین سطح بالا بودن اجزای فرآیند و عدم پیچیدگی در تعریف، آن را قابل استفاده ی کارآ جلوه می دهد. متدولوژی OPM با فعالیت های طراحی معماری سیستم در designing سعی در جمع حواس تیم ایجاد کننده دارد که کمک کننده به کارآیی است. وابستگی به استراتژی های خطاخیز زیادی در آن یافت نمی شود(که البته بدلیل ذکر نکردن جزییات گام ها برای اعمال بدیهی است) و وابسته به ابزار خاصی هم نمی باشد. در نهایت OPM نمی تواند شیوه ای ارائه دهد تا فعالیت های مدیریت پروژه ای پشتیبانی شود که ضعفی در کارآیی خواهد بود.
- متدولوژی EUP: متدولوژی EUP بسیار زیاد پیچیده می باشد. یعنی آنقدر

پیچیدگی فرآیندی آن در تعریف و اجرا زیاد هست که اعمال آن فقط برای سازمان های کلان(یا ماهیت خاصی از پروژه های مثلا (life-crytical صرفه و توجیه دارد. اجرای آن بر روی پروژه ها و تیم های ایجاد کوچک تر بسیار بی معنی است؛ به گونه ای که هرگز نمی توان آن را به تمامیت استفاده کرد. پس برای بسیاری از تیم های ایجاد اصلا قابل اجرا و اعمال نیست. متدولوژی با پیچیدگی اجزای فرآیندی که دارد از practicality هم می افتد؛ به گونه ای که برای افرادی که تازه به سازمان وارده شده اند یا تجربه ی کمی دارند بسیار غیرکارآ می نماید. البته نقطه ی قوت آن استفاده از جلسات مدیریت تیم، دقت بالا به معماری سیستم و مدلهای مبتنی بر نیازمندی هست که توجهات را به نکات مهم معطوف می کند و سعی در ایجاد کارآیی دارد. در نهایت این که ممکن است برای پیکربندی آن به ابزار متوسل شویم اما لزوما وابسته به ابزار نیستیم؛ چرا که خودش بسیار دقیق توضیح داده شده است. در نهایت اینکه به دلیل پیشنهاد استفاده از dfd متدولوژی مستعد استفاده از تکنیک های خطا خیز بوده اما فعالیت های مدیریت پروژه ای در آن سعی در کارآتر کردن EUP دارند.

- متدولوژی Fusion: متدولوژی Fusion فرآیند پیچیده ای ندارد. استفاده از آن بسیار راحت بوده و در تعریف فاز ها و مراحل و وظایف بسیار دقیق عمل کرده و در به کار گرفته شدن، پیچیدگی ای را از خود نشان نمی دهد. پس این متدولوژی تا حد خوبی قابل اعمال می باشد(البته برای طبیعت پروژه های سنگین یا حساس نه). در نهایت این که متدولوژی کارآیی خود را با داشتن اجزای ساده فرآیند، دور کردن حواس تیم ایجاد از جزییات غیرضرور با توجه کردن به معماری سیستم، عدم وابستگی به تکنیک های خطاخیز و عدم وابستگی به ابزار خاص برای پیاده سازی نشان می دهد. البته متدولوژی در تعریف فعالیت های مدیریتی پروژه ناتوان هست که کمی از کارآیی آن می کاهد.

- متدولوژی OOSE: متدولوژی OOSE در سطح بسیار قابل قبولی می تواند قابلیت

اجرا را از خود نشان دهد؛ زیرا پیچیدگی فرآیند در آن حداقلی است و جزییات تا حد خوبی برای هر مرحله بیان شده اند. به تمامیت قابل اجرا بوده و به راحتی روی درصد کثیری از پروژه ها قابل اجرا می باشد. از نظر practicality به خاطر تعریف اجزای فرآیند به صورت ساده و توجه به مدل های مبتنی بر نیازمندی در همه ی سطوح و جلب تمرکز تیم ایجاد بسیار خوب عمل می کند. همچنین عدم وابستگی به ابزار خاصی برای پیاده سازی، آن را کارآتر جلوه می دهد. البته OOSE در پشتیبانی از فعالیت های مدیریتی در طول پروژه ناتوان عمل کرده و استراتژی خاصی برای آن ندارد.

- مقایسه OPM با EUP: متدولوژی EUP در جلب توجه تیم ایجاد به جزییات اصلی و مهم بسیار عالی عمل می کند که برتری آن مقابل OPM می باشد. این متدولوژی البته نمی تواند به صورت کامل قابل اجرا باشد(جز در حالت های خاص). این سنگینی را OPM نداشته اما در تعریف فرآیند خود ساده تر(سطح بالاتر) عمل می کند. هر دو وابستگی به ابزار ندارند اما EUP در خطر بالقوه از تکنیک خطاخیز رسیدن به نمودار توالی از dfd قرار دارد. لازم به ذکر است که OPM فعالیت مدیریت پروژه ای را در خود تعبیه نکرده اما EUP در حد بالایی از آن بهره می برد که آن را کارآتر جلوه می دهد.

- مقایسه OPM با Fusion: متدولوژی Fusion در تعریف فرآیند خود به خوبی عمل کرده و جزییات آن را برای اجرا بیان می کند در حالی که OPM جزییاتی را برای اعمال درست بیان نمی کتد. متدولوژی Fusion اجزای سبکتری دارد پس کارآتر بوده ولی توجه به معماری سیستم در هر دو متدولوژی خوب بوده که باعث تمرکز تیم ایجاد می شود. هیچکدام از متدولوژی ها به فعالیت مدیریتی پروژه توجهی نداشته، وابسته به ابزار برای به کار گیری نبوده و از تکنیک های خطا خیز رنج نمی برند.

- مقایسه OPM با OOSE: با مقایسه متوجه هستیم که OOSE در عمل جزییات بهتری را برای فازها و زیرفاز های خود در اختیار می گذارد که سبب ساده شدن فرآیندش می شود، به گونه ای که بتوان از آن به راحتی در تمامیت استفاده کرد که چنین چیزی در OPM یافت نمی شود. متدولوژی OOSE با مبتنی بودن همه ی مدلهایش بر نیازمندی، تمرکز خوبی برای تیم ایجاد کرده که نقطه ی برتری آن است. هر دو، توجهی به تکنیک های خطاخیز ندارند، وابستگی به ابزار ندارند و توجهی به پشتیبانی از فعالیت های مدیریت پروژه ای ندارند.

۹.۱.۱ قابلیت مدیریت پیچیدگی

قابلیت مدیریت پیچیدگی به عنوان معیار بعدی بیشتر روی مدیریت پیچیدگی فرآیند توجه دارد. این که واحد های کاری در چرخه عمر متدولوژی باید بتوانند قابل مدیریت باشند. این مدیریت از دو طریق البته قابل دسترسی است.

اولا به کمک تقسیم بندی[۱۱] که اجازه ی تجزیه ی یک کل پیچیده را به قسمت های تشکیل دهنده ش می دهد و به این ترتیب اجزای ساده تری را بدست می آوریم و درک و اجرای کار، تسهیل می شود. چرا که با شکستن، قابلیت فهم بالا تر رفته و از سطح انتزاع[۱۲] کلی گویی دوری می کنیم. بدین ترتیب، متدولوژی می تواند فرآیند پیچیده ی خود را قابل مدیریت سازد.

دوما به کمک لایه بندی[۱۳] که اجازه می دهد تا از سطح بالا به سطح جزئی[۱۴] به صورت لایه به لایه حرکت کنیم و در این پیشروی، مفاهیم را به ترتیب درک کنیم تا درک ساده تر شود. با افزوده شدن جزییات در عمق لایه، پیچیدگی کار کاهش یافته و مدیریت می شود.

از جنبه ی مطرح شده می خواهیم متدولوژی های زیر را مقایسه کنیم و بررسی کنیم که هر یک آیا از شیوه های بالا استفاده کرده اند یا خیر.

- متدولوژی OPM: متدولوژی OPM با داشتن تعداد بالای فاز ها و مراحل ایجاد نرم افزار سعی در تقسیم پیچیدگی فرآیند دارد. البته OPM در فرآیند خود با مشخص کردن scope سیستم در ابتدای کار، سعی در پیشبرد مساله به صورت لایه لایه خواهد داشت که در این راه موفق هست. یعنی برای کم شدن پیچیدگی فرآیند، آن ها را به سطوح مرحله های داخلی شکسته و پایین می رویم. از این نظر(هم تجزیه ی کل پیچیده و هم پیشروی لایه بندی شده) OPM می تواند ساده

[۱۱] partitioning
[۱۲] abstract
[۱۳] layering
[۱۴] concrete

فهمیده و اجرا شود(دارای ۳ فاز و در کل ۹ زیرفاز).

- متدولوژی EUP: متدولوژی EUP در وسعت زیادی از partitioning کمک می گیرد و سعی دارد قابلیت فهم را با شکستن فرآیند فراهم کند. متدولوژی EUP به کمک دیسیپلین ها و فاز های تکرارشونده ی خود بسیار عالی می تواند پیچیدگی فرآیند خود را مدیریت کند؛ به این شکل که از abstract به concrete پیش روی می کند و همین گونه در پایین ترین سطح کار ها نشان داده شده اند(دارای ۶ فاز و ۱۷ دیسیپلین). متدولوژی EUP امکان بازنگری در فرآیند را نیز باز می گذارد.

- متدولوژی Fusion: این متدولوژی با شکستن فرآیند خود به ۳ فاز سعی در ساده سازی فرآیند خود دارد که در عمل از partitioning بهره می برد. سپس درون فاز های خود به شکل ساده ای وظایف را مشخص می کند(یک سطح در لایه).

- متدولوژی OOSE: متدولوژی OOSE دارای ۳ فاز که در ۲ فاز اولیه به خوبی دو زیرفاز را تشریح می کند و به خوبی می تواند از شکستن کل پیچیده به قسمت های مختلف بهره ببرد. این متدولوژی قابل مدیریت در پیچیدگی است و گام های داخلی تشکیل دهنده ی آن به خوبی مشخص هستند(یک سطح در لایه).

- مقایسه OPM با EUP: متدولوژی EUP می تواند در مدیریت پیچیدگی از -par titioning و layering بهره بگیرد و فرآیند پیچیده ی خود را به فاز های زیادی تقسیم کرده و با دیسیپلین های تکرار شونده سعی در پیشروی عمیق در هر فاز دارد. در عین حال OPM نیز کل پیچیده ی خود را به فاز های درستی تقسیم کرده اما لایه بندی عمیقی ندارد. متدولوژی EUP امکان بررسی پلن و فرآیند را در خود تعبیه کرده که می تواند جهت مدیریت پیچیدگی کارآ باشد.

- مقایسه OPM با Fusion: این دو متدولوژی به خوبی تقسیم بندی را انجام

داده اند اما OPM با داشتن تعداد بیشتری از گام ها، دقت بیشتری در مدیریت پیچیدگی دارد.

- مقایسه OPM با OOSE: این دو متدولوژی تقریبا به مانند هم عمل کرده و با داشتن ۳ فاز و گام های نزدیک، پیچیدگی خود را شکسته و دارای مدیریت در فرآیند هستند.

۱۰.۱.۱ قابل گسترش، مقیاس پذیر، قابل پیکربندی، انعطاف پذیر

نهمین معیار درگیر در چهار تا خصوصیت تحت عنوان های گسترش پذیری[۱۵]، مقیاس پذیری[۱۶]، قابلیت پیکربندی[۱۷]، انعطاف پذیری[۱۸] می باشد که می خواهیم هر کدام را در ادامه بسط دهیم.

گسترش پذیری آن است که بتوان به متدولوژی قابلیت هایی را اضافه کرد. بدین معنا که برای پروژه های خاص بتوان نیز متدولوژی را خاص کرد. اگر بتوان متدولوژی را به شکل هسته ی قابل گسترش[۱۹] تعریف کرد بهترین شکل قابلیت گسترش خواهد بود؛ چرا که در زمان اجرا و اعمال متدولوژی بتوان آن را به شکل دلخواه متناسب با پروژه، گسترش داد.

نکته ی اساسی آن است که اگر قابلیت گسترش را برای متدولوژی باقی می گذاریم، باید نقاط گسترش[۲۰] را نیز مستند کرده و گفته شود که از چه قسمی و با چه مکانیزمی گسترش می تواند انجام پذیرد.

مقیاس پذیری را می توان این گونه تعریف کرد که فرآیند متدولوژی تا چه حد می تواند روی پروژه های با سایز های مختلف اعمال شود. پس متدولوژی با مقیاس پذیری بزرگ می تواند در حوزه کاربرد خودش، هم روی پروژه های کوچک(تعداد افراد درگیر در هر لحظه کم) و هم روی پروژه های بزرگ (تعداد افراد درگیر در هر لحظه زیاد) اجرا شود و بدیهی است که هر چه گسنره سایز بیشتر باشد، متدولوژی از این نظر وضع بهتری دارد. البته جز سایز، میزان بحرانیت[۲۱] نیز اهمیت دارد که می توانند به جای هم گذاشته شوند و مانند سایز می توان با آن برخورد داشت. نکته ی

---

[۱۵] Extensibility
[۱۶] Scalability
[۱۷] Configurability
[۱۸] Flexibility
[۱۹] extensible core
[۲۰] extension points
[۲۱] criticality

آخر آن که برای رسیدن به مقیاس پذیری بالا، می توان از مدلسازی غنی کمک گرفت.

قابلیت پیکربندی به این معنی که فرآیند متدولوژی را در ابتدای پروژه پیکربندی کنیم تا با وضعیت پروژه ای تطابق پیدا کند. لذا اگر متدولوژی را بخواهیم از نظر قابلیت پیکربندی سنجید باید دید که برای فاکتور های خاص موقعیت پروژه ای، می توان پیکربندی را برای متدولوژی ارائه داد یا خیر.

در نهایت انعطاف پذیری است که به سادگی یعنی آیا می توان فرآیند متدولوژی را در حین اجرا مورد بازنگری قرار داد یا خیر. برای رسیدن به چنین ویژگی ای، متدولوژی ها می توانند از دو تکنیک استفاده کنند، اول آنکه جلسات مرور فرآیند به صورت تکرار شونده ای شکل بگیرد و دوم آنکه از افراد درگیر پروژه، بازخورد ها مرتبا گرفته شده و براساس آن بازنگری مدیریتی انجام می شود. امروزه از هر دو روش به صورت ترکیبی می توان استفاده کرد. یعنی در طول اجرای متدولوژی، بازخورد ها ثبت شده و سپس در جلسات از آن ها برای بازنگری فرآیند استفاده می کنیم.

حال با تعاریف بالا می خواهیم بین متدولوژی های زیر از منظر ۴ تا ویژگی مقایسه داشته باشیم:

- **گسترش پذیری**

متدولوژی OPM: متدولوژی OPM در تعریف گسترش پذیری بالا نمی گنجد. چرا که هسته ی آن صرفا در سطح abstract تعریف شده و جزییات کمی دارد. این چنین متدولوژی نمی تواند گسترش پذیر باشد و در اجرا نمی توان تصور کرد که برای موقعیت های خاص پروژه ای، قابلیتی به آن افزود. به علاوه اینکه حتی نقطه یا نقاطی به همراه مکانیزم های افزایش در آن تصریح نشده است.

متدولوژی EUP: متدولوژی EUP نیز گسترش پذیر نمی باشد. این متدولوژی هر چند که بسیار عظیم هست و جزییات بسیار زیادی را جهت تدقیق در گام ها بیان می کند اما نمی تواند بیش از این گسترش پذیر تر باشد.تنها کاری که در آن فقط پیشنهاد شده است و به عنوان یک توصیه ی بد شناخته می شود این است که

جهت نگهداری محصول، در فاز مثلا transition یک فعالیت کوچک برای نگهداری تعریف کرد تا بتوان maintenance را نیز محقق کرد. پس درصد کمی از گسترش پذیری را می توان برای آن قائل شد.

متدولوژی Fusion: متدولوژی Fusion اصلا نمی تواند در تعریف بالا از گسترش پذیری صدق کند. هیچ نقطه ای برای افزایش قابلیت گسترش ندارد.

متدولوژی OOSE: متدولوژی OOSE هم نمی تواند گسترش پذیر باشد. در این تعریف متدولوژی OOSE تمام فازها و زیرفازهای خود را کامل می داند و نقطه ای را برای افزودن قابلیت معرفی نمی کند.

مقایسه OPM با EUP: در این مقایسه می توان گفت هر دو متدولوژی به شکل های هسته های قابل گسترشی تعریق نشده اند. متدولوژی OPM که بسیار سطح بالا گام های خود را در فرآیند معرفی می کند و جز OPL که برای OPD ضروری می باشد، پیشنهاد دیگری برای پروژه های خاص ندارد و EUP که در گفتن ریز جزییات در هر قسمت کوتاهی نکرده و فقط می تواند نگهداری را اضافه کند. البته مکانیزم آن را تا حدودی مشخص کرده است که چه زمان این اتفاق می تواند بیافتد ولی نقطه ی دقیقی را بیان نمی کند و فقط در حد پیشنهاد هست.

مقایسه OPM با Fusion: متدولوژی Fusion که فاقد گسترش پذیری است که در این راه بسیار شبیه OPM عمل می کند. البته OPM به دلیل فارغ از جزییات بیان شدن خود سعی دارد OPL را در مواقع خاص وارد کار کند و این گونه فرآیند و مدل را گسترش دهد.

مقایسه OPM با OOSE: در متدولوژی OOSE نقطه ی خاصی برای گسترش پذیری تعبیه نشده است که از این حیث مانند OPM رفتار می کند.

- **مقیاس پذیری**

متدولوژی OPM: اینکه تا چه میزان می توان OPM را مقیاس پذیر دانست بستگی

به موارد مهمی دارد. دقت داریم که OPM هرگز نمی تواند در مدلسازی خود یعنی تک_مدل OPD از جنبه ی رفتاری به خوبی نگاه کند، نمی تواند طراحی براساس قرارداد داشته باشد و پیش شرط و پس شرط را محقق کند و در نهایت این که پیچیدگی زیادی را دخیل در کار می کند. متدولوژی حتی قادر به تصریح کنترل و مدیریت نبوده و خیلی سطح بالا تعریف شده است که پیاده سازی و اجرای آن را دچار مشکل می کند. همه ی این ها باعث می شودکه برای مقیاس های بالای پروژه ای که یا سنگین بوده و یا حیاتی، نتوان به خوبی به OPM تکیه کرد؛ چرا که روش های صوری را در مدلسازی خود نمی بیند و درعین حال پیچیدگی بالایی را در کار می آورد. تنها نکته ی مثبتی که در OPM به چشم می خورد پوشش دادن فعالیت نگهداری است که درصدی از ضریب اطمینان را به همراه می آورد.

متدولوژی EUP: متدولوژی EUP از قابلیت مقیاس پذیری تقریبا خوبی برخودار است اما استفاده از آن می تواند هزینه بر باشد؛ چرا که در صورت دقت ناکافی و اجرای آن برای پروژه های با حساسیت پایین و یا سبک، طبعات سنگینی برای تیم ایجاد خواهد داشت. در واقع EUP به خاطر تنوع مدل های خود، فعالیت های عالی مدیریتی خود، توجه به سطح کلان سازمانی خود، استفاده ی حداکثری از روش های صوری جهت تست پس شرط ها یا پیش شرط ها، توجه و اعمال تست های سنگین و در نهایت رویکرد افزایشی_تکرارشونده ی خود، برای پروژه های حساس و یا سنگین در سازمان های کلان بسیار مفید می باشد. گستره ی آن البته کما اینکه بیان شد خیلی زیاد نبوده و برای پروژه های با سایز کوچک خیلی کارآیی نخواهد داشت و حوزه ی کاربرد آن برای پروژه های کوچکتر تعریف نشده می باشد. نداشتن دقت کافی البته برای EUP می تواند آسیب زننده باشد که برای آن البته پیشنهادات و فعالیت هایی توصیه شده است که پیشتر به آن پرداخته شد اما در کل EUP متدولوژی می باشد که بتواند تعداد افراد زیادی را مدیریت کند.

متدولوژی Fusion: متدولوژی Fusion در مدلسازی قوی عمل کرده و می تواند در طرح هایی که برای operation می سازد از روش های صوری بهره بگیرد و بتواند از این طریق مدلسازی غنی تری را ارائه دهد که باعث افزایش گستره ی آن بر روی پروژه های با سایز های مختلف و حتی پروژه های با criticality بیشتر می شود. البته Fusion نگهداری و آزمون را پوشش نمی دهد که نقطه ی ضعفی برای آن برای سیستم های بزرگ به حساب می آید و گستره ی حوزه ی کاربرد آن را کاهش می دهد.

متدولوژی OOSE: متدولوژی OOSE با مدلسازی غنی خود و ارائه ی use-case و مدل های مبتنی بر آن به همراه به کارگیری از formalism کنترل خوبی بر روی پروژه های حساس تر یا سنگین تر دارد. البته از تصریح فعالیت های مدیریتی عاجز است و همچنین نداشتن نگهداری می تواند گستره ی مقیاس پذیری آن را کاهش دهد.

مقایسه OPM با EUP: در این مقایسه، OPM تنها به دلیل توجه بیشتر به نگهداری برتر است؛ در غیر این صورت در ۱- تنوع مدل ها، ۲- فعالیت های عالی مدیریتی، ۳- توجه به سطح کلان سازمانی، ۴- استفاده ی حداکثری از روش های صوری جهت تست پس شرط ها یا پیش شرط ها، ۵- توجه و اعمال تست های سنگین، ۶- رویکرد افزایشی-تکرارشونده ی برتری با EUP است.

مقایسه OPM با Fusion: متدولوژی Fusion در داشتن و به کارگیری از روش های فرمال و شکستن پیچیدگی مدل ها در انواع مدل ها و داشتن مدلسازی رفتاری خوب، برتر است. اما از طرفی دیگر OPM به دلیل ذکر نگهداری و اعمال تست ها می تواند بهتر عمل کند.

مقایسه OPM با OOSE: متدولوژی OOSE تنوع مدل های زیادی دارد و در این راستا به خوبی در برابر ضعف OPM عمل کرده و مدلسازی رفتاری و وظیفه ای عالی را مخصوصا در مدل usecase نشان می دهد. متدولوژی OOSE البته

نمی تواند در داشتن فاز نگهداری از خود برتری نشان می دهد اما به دلیل استفاده از طراحی بر اساس قرارداد می تواند بهتر عمل کند.

- **قابلیت پیکربندی**

متدولوژی OPM: هیچ تصریحی در جریان متدولوژی OPM جهت پیکربندی آن برای اعمال در پروژه ذکر نشده است و نمی توان با وضعیت پروژه ای آن را تطبیق داد.

متدولوژی EUP: در این متدولوژی به دلیل حجیم بودن بیش از آن به شخصی سازی تاکید شده است به گونه ای که پیشنهاد customize کردن آن در ابتدای شروع پروژه معمولا انجام می شود تا پیکربندی آن با موقعیت پروژه ای با فاکتورهای خاص خودش تطبیق پیدا کند. البته دقت داریم که کار هرس کردن متدولوژی به این حجم تا چه حد می تواند سخت باشد.

متدولوژی Fusion: براساس تعریف بالا از قابلیت پیکربندی در متدولوژی Fu - sion هیچ تصریحی درباره ی قابلیت پیکربندی دیده نشده است. پس نمی توان حالتی را تصور کرد که Fusion خاص یک پروژه با فاکتور های مخصوص به خود شده است.

متدولوژی OOSE: در متدولوژی OOSE نیز هیچ تصریحی درباره قابلیت پیکربندی نشده است.

مقایسه OPM با EUP: متدولوژی EUP از این نظر بر OPM برتری دارد و در ابتدا می توان پیکربندی کلی EUP را شخصی سازی کرد.

مقایسه OPM با Fusion: این دو متدولوژی مانند هم عمل کرده و قابل پیکربندی نیستند.

مقایسه OPM با OOSE: این دو متدولوژی مانند هم عمل کرده و قابل پیکربندی نیستند.

- **انعطاف پذیری**

متدولوژی OPM: متدولوژی OPM نمی تواند برای پیکربندی فرآیند خود در طول اجرای آن نیز تصریحی داشته باشد و فعالیتی برای مرور فرآیند در آن در نظر گرفته نشده است.

متدولوژی EUP: متدولوژی EUP که گسترش یافته ی RUP می باشد به خاطر ماهیت افزایشی-تکرارشونده ی خود از انعطاف پذیری بهره می برد و به کمک دیسیپلین های مدیریتی خود همیشه فرآیند را مورد پایش و نگرش قرار داده و در ابتدای هر فاز نیز سعی دارد تا محیط را برای ایجاد فراهم کند(که این کار در دیسیپلین environment انجام شده و فرآیند را به گونه ای مهیا می کنند که بتوان در محیط ایجاد از آن استفاده کرد). بدین ترتیب EUP در جریان اعمال، به صورت مداوم فرآیند را بررسی کرده و در صورت نیاز آن را دوباره پیکربندی می کند.

متدولوژی Fusion: متدولوژی Fusion نمی تواند قابلیت انعطاف پذیری را در تعریف بالا پشتیبانی کند.

متدولوژی OOSE: متدولوژی OOSE نیز نمی تواند قابلیت انعطاف پذیری را در تعریف بالا پشتیبانی کند.

مقایسه OPM با EUP: از آنجا که در بالا آمد EUP به خوبی از این معیار پشتیبانی می کند که نقطه ی برتری آن دربرابر OPM می باشد.

مقایسه OPM با Fusion: این دو متدولوژی هیچکدام نمی توانند پاسخی برای این معیار بدهند.

مقایسه OPM با OOSE: این دو متدولوژی هیچکدام نمی توانند پاسخی برای این معیار بدهند.

۱۱.۱.۱ حوزه کاربرد

حوزه کاربرد به عنوان آخرین معیار فرآیندی بیان می کند که یک متدولوژی در چه حوزه ای کاربرد دارد و برای چه تیپ از پروژه هایی استفاده نخواهد شد. خب البته بدیهی است که این معیار به context کار خیلی وابسته است و به عنوان حداقل باید متدولوژی های ایجاد سیستم نرم افزاری، سیستم های اطلاعاتی مخصوصا از نوع data-intensive را پشتیبانی کنند که حدی از مدلسازی را خواهد طلبید.

- متدولوژی OPM: متدولوژی OPM می تواند پوشش کاملی را برای چرخه ی عمر جنریک ارائه دهد، پس اگر تعریف دقیقی برای گام های آن ذکر شود می توان آن را برای پروژه های سیستم های اطلاعاتی استفاده کرد، چرا که با داشتن فاز نگهداری می تواند به خوبی از انواع مراقبت نگهداری پشتیبانی کند که نیاز هر سیستم اطلاعاتی مخصوصا حساس به داده است.

- متدولوژی EUP: متدولوژی EUP حوزه ی کاربرد خود را در پروژه های با حساسیت بالا(بحرانی) و پروژه های با سایز بزرگی که در هر لحظه تعداد افراد زیادی در آن درگیر هستند می داند(البته توجه کافی برای نگهداری ارائه نمی دهد که خود تیم ایجاد بنا بر توصیه می توانند آن را تعبیه کنند). معمولا چنین پروژه هایی در سطح سازمان های کلان نیز تعریف و پیاده می شود. برای این متدولوژی توصیه می شود که در حوزه کاربرد های با حساسیت ساده تر و سبک تر مورد استفاده قرار نگیرد؛ چرا که روند طولانی آن ممکن است شکست پروژه را رقم بزند.

- متدولوژی Fusion: متدولوژی Fusion کاربرد خود را می تواند در ساخت سیستم های اطلاعاتی حساس به داده ببیند. این متدولوژی، بسیار تعریف ساده و روشنی دارد و با بی درزی و تحلیل شیءگرای خود می تواند حداقل حوزه ی کاربرد را فراهم کند؛ چرا که Fusion به صحت خیلی پایبند است و طراحی

خود را بر اساس قرارداد انجام می دهد. البته ضعف Fusion در نداشتن آزمون و نگهداری است.

- متدولوژی OOSE: متدولوژی OOSE نیز می تواند به خوبی سیستم های کمی بزرگتر را مدیریت کند. چرا که با معرفی usecase و داشتن پروتوتایپ های کنترلی به همراه رویکرد بنای همه چیز بر نیازمندی، سطح خوبی از مدیریت پروژه ی پیچیده را فراهم می کند. اما با وجود داشتن انواع سطوح تست در نگهداری مشکل دارد.

- مقایسه OPM با EUP: متدولوژی EUP در سطح کلان سازمانی می تواند به خوبی پیاده سازی و اجرا شود در حالی که اجرای آن روی پروژه های ساده ی اطلاعاتی، بسیار ناکارآمد خواهد بود. در آن طرف OPM نمی تواند تفکر سازمانی داشته باشد و حوزه ی کاربرد آن برای سیستم های اطلاعاتی حساس به داده، منطقی است.

- مقایسه OPM با Fusion: متدولوژی Fusion به دلیل نداشتن آزمون و نگهداری نمی تواند در برابر برتری OPM که این ها را دارا می باشد، حوزه ی کاربرد وسیع تری داشته باشد.

- مقایسه OPM با OOSE: متدولوژی OOSE اما به خاطر استفاده از مکانیزم های صوری و داشتن پروتوتایپ و usecase و رویکرد usecase-driven از این نظر در حوزه های پیچیده تر استفاده شود اما OPM از جهت داشتن نگهداری برتری دارد.

۲.۱ معیارهای زبان مدلسازی

۱.۲.۱ مدلسازی شیءگرای سازگار، دقیق و بدون ابهام

اولین معیار برای سنجش متدولوژی از نظر زبان مدلسازی، تحت عنوان پشتیبانی از مدلسازی شیءگرایی سازگار(بدون تناقض)، صحیح و دقیق و بدون ابهام بیان می شود.

بدین منظور مدلسازی باید سعی خود را بکند که اولا مدل های ایزمورف هم ندهد تا تغییر در یکی باعث انتشار تغییر در مستندات دیگر نشود(هرچند که با تعریف فرآیند می توان این مورد را کنترل کرد).

دوما مدلسازی، جزییات را به اندازه کافی که ابهام از بین برود ارائه دهد، یعنی از سطح انتزاع بالا و مفهوم به سطح پایین ساده ی قابل فهم جزئی بتوان مدلسازی کرد(مدلسازی منطقی به فیزیکی).

سوما زبان مدلسازی باید بتواند هر سه دیدگاه مختلف رفتاری، وظیفه مندی و ساختاری را پشتیبانی کند و از این نظر نقصی نداشته باشد تا دست متدولوژی باز باشد.

چهارم آن که بتواند برای راههای صوری و غیر صوری جوابی داشته باشد تا بتوان از هر کدام که متدولوژی خواستار است، در اختیار بگذارد.

و در نهایت آن که بتواند سطوح درشت دانگی را به خوبی پشتیبانی کند. از مدل enterprise تا مدل های intra-Object که برای متدولوژی ضروری می نماید.

- متدولوژی OPM: متدولوژی OPM با معرفی OPD که تنها مدل آن می باشد، از انتشار تغییر جلوگیری می کند. جزییات در آن بسیار خلاقانه بوده و می تواند ابهامات را ازبین ببرد. منتها مشکل آن در مدلسازی جنبه ی مهم رفتار است که در آن غفلت می کند و نمودار نمی تواند در حد مطلوبی آن را پوشش دهد. مشکل بعدی زمانی ظهور می کند که OPD نمی تواند از نظر قواعد صوری از خود انعطاف نشان دهد. در این نظر، OPD فقط سطوح داخل شیءای و بین شیءای را می تواند مدل کند و خبری از مشخص نمودن زیرسیستم و سیستم و

سازمان وجود ندارد که ضعف در عدم وجود سطوح مختلف درشت دانگی است.

- متدولوژی EUP: این متدولوژی با پیشنهاد UML انعطاف بالایی از اعمال روش های صوری و غیرصوری را گذاشته ولی عدم اصرار به UML باعث می شود که مدل سطوح مختلف ریزدانگی از سامان تا اجزای داخل شیءای وجود داشته باشد. همچنین مدلسازی خود را تا حد زیادی شیءگرا معرفی می کند(البته در این راه برای ثبت نیازمندی از usecase و dfd استفاده می کند). باید سعی کند که از مدل های خطاخیزی که سبب انتشار تغییر می شوند جلوگیری کند(مانند نمودار توالی و interaction که ایزومورف هم هستند). در نهایت این که EUP می تواند در مدلسازی های خود همه ی جنبه های رفتاری را پوشش دهد.

- متدولوژی Fusion: متدولوژی Fusion در مدلسازی، شیءگرا عمل کرده و مدلهای سازگار بدون تناقض ارائه می دهد. جزییات در آن در حد بالایی تدقیق شده اند و جنبه ی ساختاری و رفتاری را به خوبی نشان می دهد. جنبه ی وظیفه را هم سعی دارد در life-cycle model حفظ کند. این متدولوژی در مدلسازی خود، ارائه مدلی از سازمان و زیرسیستم ها ناتوان عمل می کند ولی از روش های صوری بهره می گیرد.

- متدولوژی OOSE: این متدولوژی در مدلسازی بسیار شبیه UML عمل کرده و مانند آن شیءگرا بوده، از جزییات زیادی برخودار است، هر سه جنبه ی مدلسازی را پوشش می دهد و راهکاری برای استفاده از روش های صوری پیشنهاد می کند. تنها نکته ی آن عدم توانایی در مدل کردن سازمان می باشد که سطح بالای درشت دانگی را ندارد.

- مقایسه OPM با EUP: متدولوژی OPM نمی تواند مانند EUP همه ی جنبه های مدلسازی را پوشش داده و وجود سطوح ریزدانگی تا درشت دانگی در EUP غنی تر است که برتری های EUP را در مدلسازی می رساند. البته OPM مدلسازی

کاملا شیءگرای سازگاری را با قطعیت ارائه می دهد اما این ضعف به صورت بالقوه در EUP موجود است. متدولوژی EUP در اعمال فرمالیزم، انعطاف دارد که برتری آن نسبت به OPM است و در نهایت این که در هر دو متدولوژی جزییات مدلسازی به قدری بالا می باشد که ابهامات برطرف می شوند.

- مقایسه OPM با Fusion: متدولوژی Fusion در داشتن فرمالیزم منعطف عمل کرده و هر سه جنبه ی مدلسازی را پشتیبانی می کند در حالی که در OPM چنین نیست. هر دو متدولوژی زبان مدلسازی خود را به خوبی توضیح داده اند و دارای جزییات کافی هستد تا از بتوان از سطح بالا به سطح پایین جزیی، پیاده سازی را انجام داد. در این راه، دو مدلسازی نتوانسته اند سطوح مختلف ریزدانگی تا درشت دانگی را به طور کامل پوشش دهند و در این راه هر دو شیءگرا عمل کرده و خود را دقیق و بدون تناقض نشان می دهند.

- مقایسه OPM با OOSE: آنجایی زبان OOSE برتری خود را نشان می دهد که می تواند برای طراحی براساس قرارداد راهکار ارائه کرده و سیستم و زیر سیستم ها را مشخص کند و هر سه جنبه ی مدلسازی را نشان دهد. البته OOSE باید بتواند مشکلات سازگاری بالقوه ای که ممکن است به دلیل تعدد نمودار ها بوجود می آید را برطرف نماید؛ مساله ای که OPM از آن رنج نمی برد.

۲.۲.۱ مدیریت تناقض ها و پیچیدگی ها

معیار آخر ناظر بر مدیریت تناقض ها و پیچیدگی ها می باشد. به این منظور که زبان مدلسازی به اندازه کافی راهبردها و سازوکارهایی در اختیار بگذارد که بشود جلوی ناسازگاری ها را گرفت تا تناقضی بوجود نیاید. از طرفی پیچیدگی مدلها را مدیریت کرد.

دو راه مرسوم برای کمک به زبان مدلسازی وجود دارد. اول آن که قواعد معنایی باید کامل باشد. قواعد معنایی[۲۲] کامل مدل ها به تعریف مفاهیم مختلف در مدل ها و ارتباط دهنده ی انواع مدل ها کمک شایانی می کند. اگر قواعد به درستی بیان نشده باشد ریسک ظهور تناقض در نمودار ها زیاد می شود.

دوم آن که زبان مدلسازی باید ابزار های کنترل پیچیدگی را بدهد. مثل نمودار پکیج در UML که بهترین مثال برای این مضمون هست.

- متدولوژی OPM: این متدولوژی با تک مدل خود، آن قدر در حد بسیار مطلوبی قواعد معنایی را در مدل نموداری خود شرح داده که امروزه نیز برای مدل کردن بعضی از موارد در فیلد های خاص استفاده می شود(هر چند با ظهور UML انتظار فراموش شدن آن می رفت). لذا در مدیریت تناقض بسیار خوب عمل می کند(به دلیل یک مدل خوب تشریح شده) اما نمی تواند پیچیدگی خود را مدیریت کند؛ چرا که سعی دارد همه ی جزییات را در خود پوشش دهد. متدولوژی OPM برای کنترل پیچیدگی، راه حل خود را در تنظیم سند متنی-روایی می بیند که موسوم به OPL است و می تواند راه حل بدی تلقی شود.

- متدولوژی EUP: متدولوژی EUP که از RUP گسترش یافته است، مانند آن می تواند برای مدیریت پیچیدگی از نمودار های پکیج استفاده کند. متدولوژی EUP می تواند به کمک UML که پیش از این قواعد معنایی را به صورت دقیق بیان کرده است، از مزایای مدیریت تناقض آن بهره ببرد و یا برای dfd می تواند از لایه

[۲۲] semantic

بندی تعریف شده در آن استفاده کرده تا پیچیدگی کنترل شود.

- متدولوژی Fusion: متدولوژی Fusion در مدلسازی های خود، در حد بسیار خوبی تمام مدل های خود را تشریح کرده و قواعد معنایی آن را ناپخته نمی گذارد. به این ترتیب ریسک ظهور تناقض کم شده و سازگاری بالایی در مدل ها بوجود می آید. البته Fusion نمی تواند در مدل خود راهکاری را برای ارائه مدلی که زیرسیستم ها و کامپوننت های سیستم را مشخص کرده و یا آن ها را در لایه های مختلف نشان دهد که از این نظر در مدیریت پیچیدگی کارآ نیست.

- متدولوژی OOSE: متدولوژی OOSE هم در مدلسازی خود با تعریف مفاهیم مختلف در مدل ها، سازگاری را بوجود آورده و با مشخص نمودن زیرسیستم ها در مدل های تحلیل خود، گامی برای مدیریت پیچیدگی بر می دارد.

- مقایسه OPM با EUP: متدولوژی EUP بسیار پخته عمل می کند و برتری خود را در مدیریت پیچیدگی می بیند. اما OPM در سازگاری بهتر عمل کرده و با داشتن تنها یک مدل، وابستگی بین مدل ها را پایین آورده و با تعریف غنی قواعد معنایی در همان تک مدل، ریسک ظهور تناقض را کم می کند.

- مقایسه OPM با Fusion: متدولوژی Fusion و OPM در این زمینه هر دو مدل های خود را به خوبی تشریح می کنند و در این راه OPM به خاطر تک_مدله بودن، کمتر در معرض افشای ناسازگاری است. در این بین، دو متدولوژی راهکار خاصی را برای مدل کردن پیچیدگی ندارند.

- مقایسه OPM با OOSE: متدولوژی OOSE و OPM نیز مدل های خود را به خوبی تشریح می کنند و در این راه نیز OPM به دلیل وجود یک مدل در آن، کمتر در معرض ظهور ناسازگاری است. اما OOSE با مدل هایی که تعریف می کند می تواند زیرسیستم ها را تشخیص داده و آن ها را مدلسازی کند که نوعی از

مدیریت پیچیدگی است.