دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر

کارشناسی ارشد
مهندسی نرم‌افزار

عنوان

متدولوژی‌های ایجاد نرم‌افزار

تمرین اول

نگارش

مهدی عیدی

استاد

دکتر رامان رامسین

پاییز ۱۴۰۳

چکیده

در این تمرین، متدولوژی‌های BON و USDP در یک قسمت و متدولوژی‌های ASD و EUP در قسمتی دیگر معرفی و مقایسه شدند. برای این مقایسه، ۱۱ معیار برای فرایند و ۲ معیار برای زبان مدل‌سازی معرفی گردید. ابتدا متدولوژی‌های BON و USDP بر اساس این معیارها مورد ارزیابی قرار گرفتند و سپس مقایسه‌ای مشابه برای متدولوژی‌های ASD و EUP انجام شد. این تحلیل به شفاف‌سازی ویژگی‌ها و نقاط قوت و ضعف هر یک از متدولوژی‌ها پرداخته که به انتخاب برای ایجاد سیستم‌های نرم‌افزاری کمک می‌کند.

کلیدواژه‌ها: مقایسه متدولوژی‌ها، معیارهای فرایند، معیارهای زبان مدل‌سازی، ارزیابی متدولوژی‌ها

فهرست مطالب

فهرست تصاویر

فصل ۱

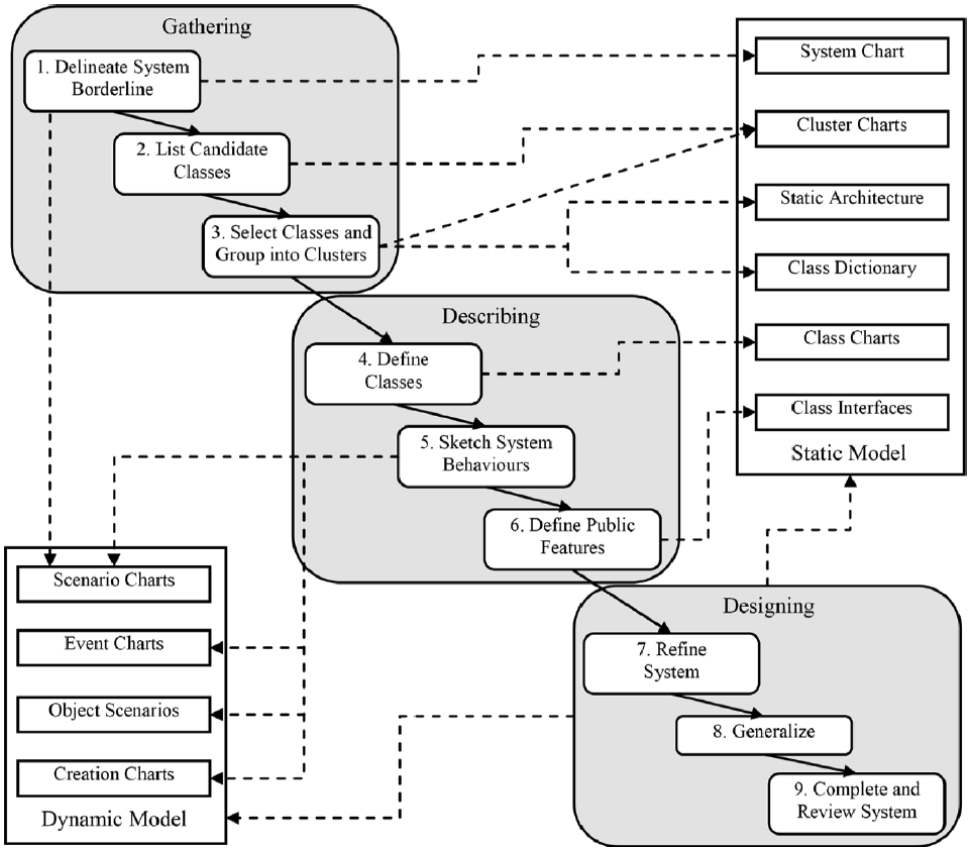
متدولوژی BON

متدولوژی BON نخستین بار در مقاله‌ای توسط نرسون[۱] در سال ۱۹۹۲ معرفی شد [۳] که نام آن مخفف "Better Object Notation" بود. نسخه بازبینی‌شده و بسیار مفصل‌تر این متدولوژی در سال ۱۹۹۵ ارائه شد که این بار مخفف "Business Object Notation" بود. BON صرفاً یک نمادگذاری ساده نیست، بلکه یک متدولوژی کامل است که فازهای تحلیل و طراحی در چرخه‌حیات ایجاد نرم‌افزار را پوشش می‌دهد. این متدولوژی تلاش می‌کند که مستقل از زبان برنامه‌نویسی باشد؛ با این حال، به شدت تحت تأثیر مکانیزم‌های ادعا[۲] در زبان Eiffel و مفهوم "طراحی بر اساس قرارداد"[۳] است.

فرایند BON شامل نُه قدم یا تکلیف[۴] است که در سه فاز تقسیم شده‌اند. تکالیف ۱ تا ۶ بر تحلیل تمرکز دارند و تکالیف ۷ تا ۹ به طراحی می‌پردازند (شکل۱-۱). ایجادکنندگان می‌توانند در صورت نیاز، ترتیب انجام تکالیف را تغییر دهند اگر این تغییر به دستیابی به اهداف پروژه کمک کند، اما ضروری است که در نهایت تمام مدل‌های لازم تولید شوند.

هر تکلیف در فرایند BON دارای مجموعه‌ای از منابع ورودی[۵] است، توسط معیارهای پذیرش[۶] کنترل می‌شود، و مجموعه‌ای از مصنوعات تحویل‌دادنی[۷] را تولید می‌کند. هدف فرایند BON این است که به تدریج مصنوعات را بسازد. این مصنوعات توصیفات ایستا و پویا از سیستم در حال ایجاد ارائه می‌دهند. توصیفات ایستا مدل ایستای سیستم را تشکیل می‌دهند. این مدل شامل توصیفات رسمی کلاس‌ها و گروه‌بندی آن‌ها در

[۱] Jean-Marc Nerson
[۲] assertion
[۳] design by contract
[۴] task
[۵] input sources
[۶] acceptance criteria
[۷] deliverable



شکل ۱-۱: فرایند BON: تکالیف و مصنوعات قابل تحویل آن‌ها [۱]

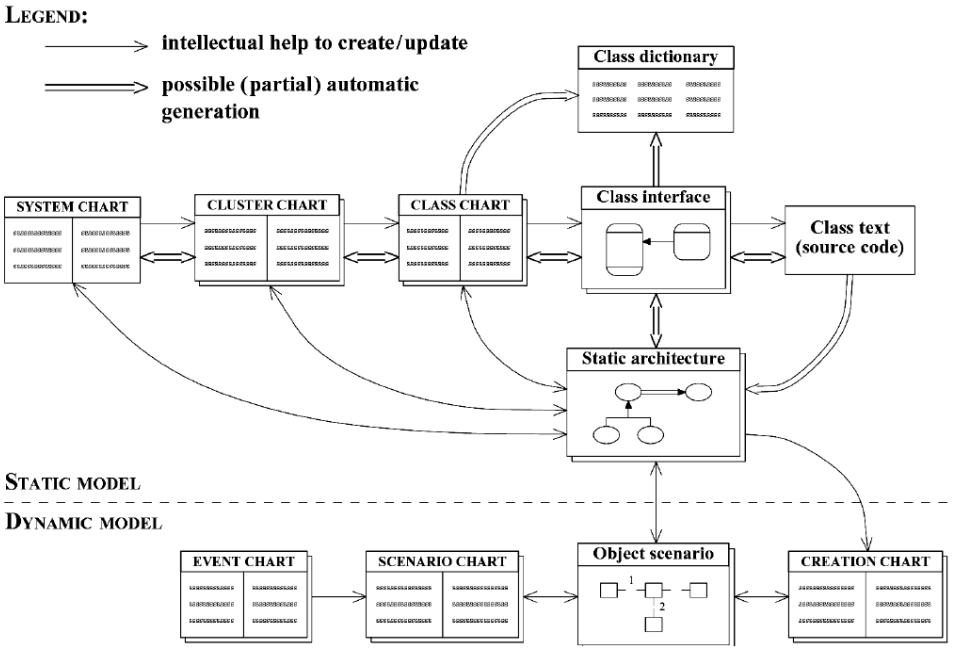
خوشه‌ها، همچنین روابط مشتری–سرویس‌دهنده[۸] و وراثت بین آن‌ها است و ساختار سیستم را نشان می‌دهد. از سوی دیگر، توصیفات پویا مدل پویای سیستم را تشکیل می‌دهند. این مدل رویدادهای سیستم، انواع اشیایی که مسئول ایجاد دیگر اشیاء هستند و سناریوهای اجرای سیستم را مشخص می‌کند. این سناریوها انواع انتخاب شده از استفاده از سیستم را با نمودارهایی که تبادل پیام بین اشیاء را نشان می‌دهند، به تصویر می‌کشند.

مصنوعات BON به یکدیگر وابسته هستند؛ بین برخی از آن‌ها ارتباطات نزدیکی وجود دارد. اگرچه مدل‌های ایستا و پویا دو نوع توصیف کاملاً متفاوت از سیستم هستند، اما به شدت با یکدیگر مرتبط‌اند، زیرا اشیاء در حال تبادل پیام در مدل پویا دقیقاً با کلاس‌های موجود در معماری ایستا مطابقت دارند. نمای کلی از تمام مصنوعات BON و وابستگی‌های متقابل آن‌ها در شکل۱-۲ ارائه شده است.

۱-۱ فعالیت‌ها

همانطورکه در شکل ۱-۱ قابل مشاهده است، فعالیت‌های BON به سه فاز تقسیم شده و در مجموع نُه تکلیف دارد که در ادامه به بررسی مختصر هرکدام خواهیم پرداخت.

[۸] client-server



شکل ۱-۲: مصنوعات BON و وابستگی‌های متقابل آن‌ها [۲]

۱-۱-۱ مشخص کردن مرزهای سیستم

این تکلیف[۹] به دیدگاه[۱۰] اصلی از دنیایی که باید درک شود و سیستمی که قرار است مدل‌سازی شود، می‌پردازد. از طریق تکنیک‌های تثبیت شده گردآوری اطلاعات و تحلیل سیستم‌ها، محدوده سیستم[۱۱] و زیرسیستم‌های آن مشخص می‌شود، استعاره‌های کاربر[۱۲] جمع‌آوری می‌گردد و وظیفه‌مندی سیستم به صورت سناریوهای معمول استفاده تعریف می‌شوند. همچنین سیاست کلی استفاده مجدد در این تکلیف تعیین می‌شود، زیرا بر سایر تکالیف فرایند تأثیر می‌گذارد.

فعالیت اصلی در این تکلیف، تحلیل قلمرو مسئله و تصمیم‌گیری درباره بخش‌هایی است که به سیستم تعلق دارند. در BON، یک سیستم (یا حتی کل قلمرو مسئله) شامل یک یا چند خوشه است که هر کدام تعدادی کلاس و/یا زیرخوشه را در بر می‌گیرد. خوشه‌بندی به طور اساسی مکانیزمی برای گروه‌بندی کلاس‌ها است، اما برای نمایش زیرسیستم‌ها نیز استفاده می‌شود. در صورتی که سیستم بیش از حد پیچیده باشد، زیرسیستم‌های اصلی در اولین وظیفه فرایند BON شناسایی می‌شوند. هر زیرسیستم به عنوان یک خوشه سطح بالا مدل‌سازی می‌شود که در مراحل بعدی شامل کلاس‌هایی برای پیاده‌سازی ساختار و رفتار زیرسیستم خواهد بود. نمودار سیستم شامل توضیح مختصری از هر خوشه سطح بالا در سیستم است. استعاره‌های کاربر نیز شناسایی می‌شوند، که عمدتاً برای شناسایی کلاس‌ها در تکالیف بعدی استفاده می‌شوند، اما به تعریف مرزهای سیستم نیز کمک

[۹] delineate system borderline
[۱۰] view
[۱۱] scope
[۱۲] user metaphor

می‌کنند. ترکیب این استعاره‌ها با تحلیل ساختاری قلمرو مسئله و سیستم نشان می‌دهد کدام بخش‌های قلمرو مسئله از دیدگاه کاربران و متخصصان دامنه داخل سیستم قرار می‌گیرند و کدام بخش‌ها به دنیای بیرون تعلق دارند، بنابراین مرز سیستم را مشخص می‌کنند.

سایر فعالیت‌های این تکلیف بر سیستم و وظیفه‌مندی‌های آن از دیدگاه کاربران متمرکز است. جریان‌های اطلاعات ورودی و خروجی شناسایی می‌شوند، وظیفه‌مندی اصلی سیستم تعریف می‌شود و موارد استفاده معمول به عنوان سناریوهای سیستم تعیین و توصیف می‌شوند. یک سناریوی سیستم توصیفی از اجرای جزئی احتمالی سیستم است. این سناریو شامل دنباله‌ای از رویدادها است که توسط یک یا چند محرک داخلی یا خارجی آغاز می‌شوند و رویدادهای نتیجه را به ترتیب وقوع نشان می‌دهد. برخی سناریوهای جالب سیستم معمولاً برای نشان دادن جنبه‌های مهم رفتار کلی سیستم جمع‌آوری می‌شوند. توصیفی از این سناریوها که اقدامات انجام‌شده در هر کدام را به تصویر می‌کشد، به عنوان نمودارهای سناریو ترسیم می‌شوند.

۱-۱-۲ فهرست کردن کلاس‌های کاندید

این تکلیف[۱۳] عمدتاً به استخراج فهرستی از کلاس‌های کاندید از قلمرو مسئله می‌پردازد. این فهرست در جدول‌های خاصی به نام نمودارهای خوشه وارد می‌شود. اگرچه نمودارهای خوشه با یک فهرست از کلاس‌های کاندید شروع می‌شوند، در طول فرایند BON دقیق‌تر شده و تکمیل خواهند شد و در نهایت شامل توصیفاتی از کلاس‌ها و زیرخوشه‌ها در یک خوشه خواهند بود. تحلیل‌گران همچنین یک واژه‌نامه از اصطلاحات و مفاهیم فنی مورد استفاده در قلمرو مسئله تهیه خواهند کرد. تمام مصنوعات تولیدشده توسط کاربران نهایی و کارشناسان دامنه بازبینی و اعتبارسنجی[۱۴] می‌شوند.

۱-۱-۳ انتخاب کلاس‌ها و گروه‌بندی در خوشه‌ها

در این تکلیف[۱۵] با فهرست کاندیدهایی که در تکلیف اول تهیه شده‌اند شروع می‌کنیم. این مفاهیم به صورت کلاس مدل‌سازی می‌شوند و سپس در خوشه‌ها گروه‌بندی می‌گردند. این تکلیف همچنین شامل شناسایی روابط بین کلاس‌ها در یک خوشه و بین خوشه‌ها می‌شود؛ روابطی مانند وراثت، مشتری–سرویس‌دهنده، و تجمیع[۱۶]. مجموعه‌ای از نمودارها که به معماری ایستا معروف هستند و روابط بین کلاس‌ها و خوشه‌ها در سیستم را توصیف می‌کنند، به عنوان مصنوع اصلی این تکلیف تولید می‌شوند. همچنین یک واژه‌نامه کلاس تهیه می‌شود که شامل

[۱۳] listing candidate classes
[۱۴] validation
[۱۵] selecting classes and grouping into clusters
[۱۶] aggregation

فهرستی مرتب از کلاس‌ها به همراه توضیحات متنی آن‌ها است. نمودار سیستم و نمودارهای خوشه با نتایج این تکلیف به‌روزرسانی می‌شوند.

۱-۱-۴ تعریف کلاس‌ها

پس از انتخاب و گروه‌بندی مجموعه اولیه‌ای از کلاس‌ها، تکلیف بعدی[۱۷] تعریف هر کلاس از نظر وضعیت[۱۸] یعنی اطلاعاتی که می‌تواند ارائه دهد، رفتار یعنی عملیاتی که می‌تواند انجام دهد و قوانین کلی که باید توسط کلاس و مشتریان آن رعایت شوند، است. این کار شامل تکمیل نمودارهای کلاس با موارد زیر می‌شود:

- پرس و جوها[۱۹]: توابعی که اطلاعاتی درباره وضعیت سیستم ارائه می‌دهند بدون اینکه آن را تغییر دهند (معادل ویژگی‌ها[۲۰]).
- دستورات[۲۱]: عملیاتی که اطلاعاتی باز نمی‌گرداند اما ممکن است وضعیت را تغییر دهد (معادل عملیات‌ها).
- محدودیت‌ها[۲۲]: قوانین کلی کسب‌وکار و شرایط سازگاری مربوط به کلاس.

نتایج این تکلیف سپس توسط کاربر نهایی یا مشتری بازبینی و اعتبارسنجی می‌شوند.

۱-۱-۵ ترسیم رفتار سیستم

در این تکلیف[۲۳]، مدل پویا سیستم به طور کامل بسط داده می‌شود. نمودارهای سناریوی اولیه که مهم‌ترین انواع استفاده از سیستم را ثبت می‌کنند، به عنوان نتیجه تکلیف ۱ ساخته شده‌اند و ارزش زیادی برای یافتن کلاس‌های کاندید اولیه و انتخاب بین دیدگاه‌های مختلف از قلمرو مسئله دارند. با این حال، هدف اصلی این تکلیف ساخت یک مدل جامع‌تر و دقیق‌تر از استفاده‌های احتمالی سیستم است. رویدادهای خارجی (ورودی) که ارتباطات اشیاء را آغاز می‌کنند، و همچنین رویدادهای داخلی مهم (خروجی) که به صورت غیرمستقیم توسط رویدادهای ورودی تحریک می‌شوند، شناسایی شده و در نمودارهای رویداد فهرست می‌شوند. کلاس‌هایی که در طول اجرای سیستم نمونه‌سازی می‌شوند و همچنین کلاس‌هایی که آن‌ها را نمونه‌سازی می‌کنند، مشخص شده و در نمودارهای ایجاد ثبت می‌شوند. برای هر سناریوی سیستم که یک مورد استفاده[۲۴] معمول از سیستم را به

[۱۷] defining classes
[۱۸] state
[۱۹] query
[۲۰] attribute
[۲۱] command
[۲۲] constraint
[۲۳] sketching system behavior
[۲۴] use case

تصویر می‌کشد، توالی ارتباطات پیامی بین اشیاء که برای تحقق سناریو هدف‌گذاری شده‌اند، مشخص شده و در یک سناریوی شیء مدل‌سازی می‌شود. این معمولاً نیاز به تکمیل و دقیق‌تر سازی نمودارهای سناریو دارد. مدل پویایی که به این ترتیب ساخته می‌شود، از نظر سازگاری با مدل ایستا بررسی می‌گردد و در نهایت توسط کاربر نهایی یا مشتری بازبینی و اعتبارسنجی می‌شود.

۱-۱-۶ تعریف ویژگی‌های عمومی

در این تکلیف[۲۵]، توضیحات غیررسمی کلاس‌ها که در تکلیف ۴ در نمودارهای کلاس وارد شده‌اند، به رابط‌های رسمی کلاس یا همان ویژگی‌ها، همراه با قراردادهای نرم‌افزاری ترجمه می‌شوند. پرس و جوها به توابعی تبدیل می‌شوند که اطلاعاتی بازمی‌گردانند و معمولاً با ویژگی‌ها مطابقت دارند. دستورات به رویه‌هایی تبدیل می‌شوند که ممکن است وضعیت سیستم را تغییر دهند و معمولاً با عملیات‌ها تطابق دارند. این توابع و رویه‌ها به عنوان ویژگی‌ها شناخته می‌شوند. محدودیت‌ها به پیش‌شرط‌ها و پس‌شرط‌ها[۲۶] برای عملیات و ثابت‌های کلی برای کل کلاس ترجمه می‌شوند و قرارداد را شکل می‌دهند. امضای هر ویژگی عمومی (تابع یا رویه) نیز مشخص می‌شود. نتایج این تکلیف در رابط‌های کلاس نشان داده می‌شوند، که نمودارهایی با توضیحات دقیق، دارای نوع و رسمی از کلاس‌ها و روابط آن‌ها همراه با امضای ویژگی‌ها و قراردادهای تکمیل‌شده هستند. نوع ویژگی‌ها معمولاً به کشف روابط جدید مشتری–سرویس‌دهنده بین کلاس‌ها منجر می‌شوند که این روابط نیز در نمودارها مدل‌سازی می‌شوند. معماری ایستا برای بازتاب اصلاحات انجام‌شده در این وظیفه به‌روزرسانی می‌شود.

۱-۱-۷ دقیق‌تر سازی سیستم

این تکلیف[۲۷] بخش طراحی فرایند BON را آغاز می‌کند و بنابراین شامل تکرار بسیاری از فعالیت‌هایی است که قبلاً برای کلاس‌های تحلیل انجام شده‌اند، این بار بر روی کلاس‌های طراحی جدید اعمال می‌شوند. کلاس‌های موجود به‌ویژه ویژگی‌ها، قراردادها و روابط نیز برای انطباق با کلاس‌های طراحی و پیاده‌سازی تصمیمات طراحی، اصلاح و دقیق‌تر سازی می‌شوند. این تغییرات باعث اصلاح و دقیق‌تر سازی مدل پویا می‌شوند. نمودارها و جداول مرتبط، از جمله معماری ایستا، رابط‌های کلاس، نمودارهای رویداد، سناریوهای شیء و واژه‌نامه کلاس، به‌روزرسانی می‌شوند.

---

[۲۵] defining public features
[۲۶] pre- post-conditions
[۲۷] refining the system

۱-۱-۸ تعمیم

این تکلیف[۲۸] به بهبود سلسله‌مراتب وراثت کلاس‌ها با استخراج وضعیت و رفتار مشترک و قرار دادن آن‌ها در ابرکلاس‌های انتزاعی می‌پردازد. نمودارها و جداول مرتبط، از جمله معماری ایستا، رابط‌های کلاس و واژه‌نامه کلاس، به‌روزرسانی می‌شوند.

۱-۱-۹ تکمیل و بازبینی سیستم

در این تکلیف[۲۹] نهایی، مدل‌ها صیقل داده شده و تکمیل می‌شوند و سازگاری کلی سیستم بررسی می‌شود. این معمولاً شامل بازبینی و بهبود مدل‌های ایستا و پویا، تأیید نحوی[۳۰] کلاس‌ها و بررسی سازگاری ثابت‌های کلاس و پیش‌شرط‌ها و پس‌شرط‌های رویه‌ها است. نمودارها و جداول مرتبط، به‌ویژه معماری ایستا، رابط‌های کلاس، نمودارهای رویداد، سناریوهای شیء و واژه‌نامه کلاس، به‌روزرسانی می‌شوند.

۱-۲ مصنوعات

- **نمودار سیستم (System Chart)**: تعریف سیستم و فهرست خوشه‌های مرتبط. فقط یک نمودار سیستم برای هر پروژه وجود دارد؛ زیرسیستم‌ها از طریق نمودارهای خوشه مرتبط توضیح داده می‌شوند.
- **نمودارهای خوشه (Cluster Charts)**: تعریف خوشه‌ها و فهرست کلاس‌ها و زیرخوشه‌های مرتبط، در صورت وجود. یک خوشه ممکن است نمایانگر یک زیرسیستم کامل باشد یا فقط یک گروه از کلاس‌ها.
- **نمودارهای کلاس (Class Charts)**: تعریف کلاس‌های تحلیل از نظر دستورات، پرس و جوها و محدودیت‌ها، که برای کارشناسان دامنه و افراد غیر فنی قابل درک باشند.
- **واژه‌نامه کلاس (Class Dictionary)**: فهرستی مرتب شده به لحاظ الفبایی از تمام کلاس‌های موجود در سیستم، که خوشه هر کلاس و توضیح کوتاهی از آن را نشان می‌دهد. باید به صورت خودکار از نمودارهای کلاس/رابط‌ها تولید شود.
- **معماری ایستا (Static Architecture)**: مجموعه‌ای از نمودارهایی که خوشه‌ها، هدرهای کلاس و روابط آن‌ها را نشان می‌دهند. نمای کلی از سیستم که قابل بزرگنمایی است.

---

[۲۸] generalizing
[۲۹] completing and reviewing the system
[۳۰] syntactic verification

- **رابط‌های کلاس (Class Interfaces)**: تعاریف دارای نوع از کلاس‌ها با امضاهای ویژگی و قراردادهای رسمی. نمایی دقیق و جزئی از سیستم.
- **نمودارهای ایجاد (Creation Charts)**: فهرستی از کلاس‌هایی که مسئول ایجاد نمونه‌هایی از کلاس‌های دیگر هستند. معمولاً فقط یک نمودار برای هر سیستم وجود دارد، اما در صورت نیاز می‌تواند برای زیرسیستم‌ها تکرار شود.
- **نمودارهای رویداد (Event Charts)**: مجموعه‌ای از رویدادهای خارجی ورودی یا محرک‌ها که رفتارهای جالب سیستم را تحریک می‌کنند و مجموعه‌ای از رویدادهای داخلی خروجی که پاسخ‌های جالب سیستم را ایجاد می‌کنند. می‌تواند برای زیرسیستم‌ها تکرار شود.
- **نمودارهای سناریو (Scenario Charts)**: فهرستی از سناریوهای شیء که برای نشان دادن رفتار جالب سیستم استفاده می‌شود. زیرسیستم‌ها می‌توانند نمودارهای سناریوی محلی خود را داشته باشند.
- **سناریوهای شیء (Object Scenarios)**: نمودارهای پویا که ارتباطات مربوط به اشیاء را برای برخی یا تمام سناریوهای موجود در نمودار سناریو نشان می‌دهند.

فصل ۲

متدولوژی USDP

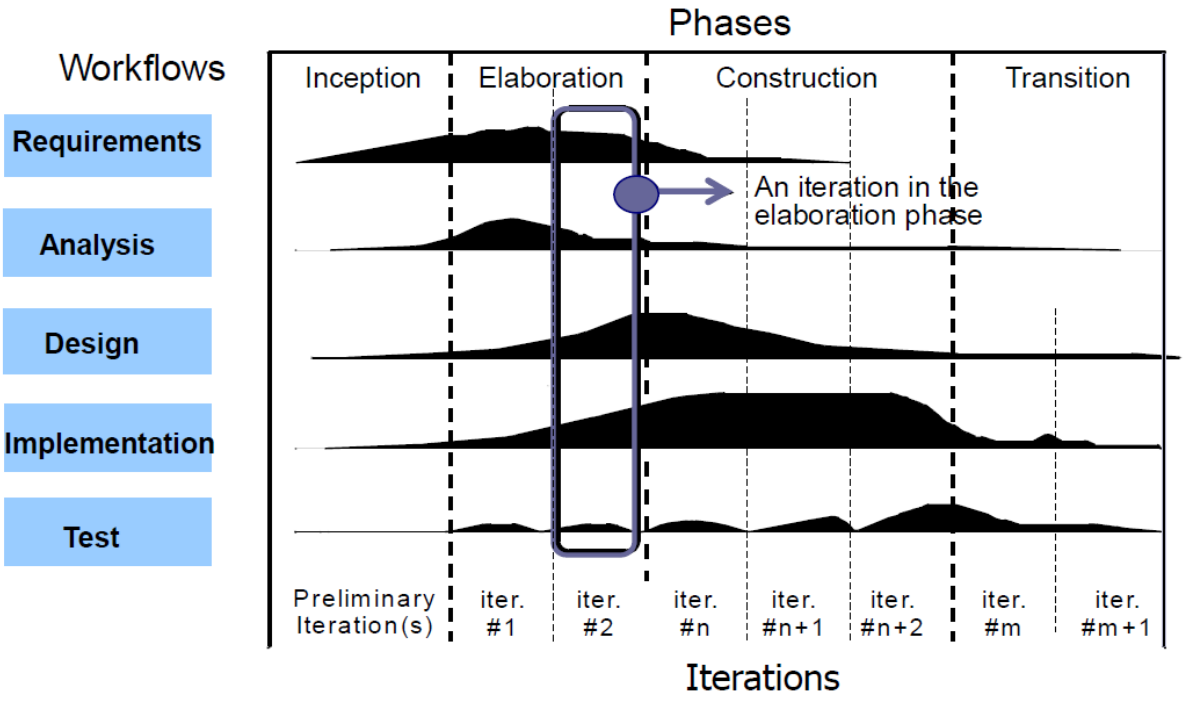
متدولوژی USDP[۱] که با نام UP نیز شناخته می‌شود، در سال ۱۹۹۹ توسط رامبا[۲]، جیکابسن[۳] و بوچ[۴] ساخته شد که نسخه ساده شده و غیر اختصاصی متدولوژی RUP[۵] است. این متدولوژی بر پایه UML[۶] بوده، و مبتنی بر Use-Case است. همچنین به صورت فرایند تکراری–افزایشی[۷] اجرا می‌شود. در این متدولوژی چرخه‌حیات نرم‌افزار بر روی زمان به چهار فاز ترتیبی تقسیم می‌شود که در شکل۲–۱ می‌توان دید. به ترتیب فاز آغاز[۸]، فاز تفصیل[۹]، فاز ساخت[۱۰] و فاز انتقال[۱۱]، چرخه‌حیات در UP را تشکیل می‌دهند[۴].

۲–۱ چرخه‌حیات نرم‌افزار در USDP

در شکل۲–۱ برای هر فاز می‌توانیم تعدادی تکرار را مشاهده کنیم که هرکدام شامل چهار جریان‌کاری[۱۲] هستند که عبارتند از نیازمندی‌ها، تحلیل، طراحی، پیاده‌سازی و تست. در ادامه به توضیح فازها و فعالیت‌های آن‌ها می‌پردازیم.

---

۱ unified software development process
۲ James Rumbaugh
۳ Ivar Jacobson
۴ Grady Booch
۵ rational unified process
۶ unified modeling language
۷ iterative-incremental
۸ inception
۹ elaboration
۱۰ construction
۱۱ transition
۱۲ workflow



شکل ۲-۱: مدل چرخه حیات USDP

۲-۱-۱ فاز آغاز

این فاز نقطه عطف چشم‌انداز[۱۳] پروژه است که در آن چشم‌انداز، حوزه[۱۴] پروژه و توجیه اقتصادی[۱۵] تعریف می‌شوند.

این فاز نهایت چهار هفته طول کشیده و در طی یک یا دو تکرار انجام می‌شود. در این فاز برای امکان‌سنجی[۱۶]، نمونه‌های اولیه[۱۷] فنی برای اعتبار سنجی تصمیمات تکنولوژی و همچنین نمونه‌های اولیه اثبات مفهوم[۱۸] برای اعتبارسنجی نیازمندی‌های کسب و کار، ساخته می‌شوند. یک توجیه اقتصادی ساخته می‌شود تا نشان‌داده شود که پروژه منافع قابل اندازه‌گیری تحویل خواهد داد. در این فاز نیازمندی‌های اساسی ثبت می‌شوند تا کمک کنند حوزه سیستم مشخص شود. همچنین یکی از مهم‌ترین کارها در این مرحله، تشخیص ریسک‌های حیاتی است. این فاز یک مرحله مقدماتی است که در آن میخواهیم به مواردی پاسخ دهیم از قبیل این که هدف پروژه چیست و آیا ارزش تلاش دارد؟ آیا پروژه از لحاظ تکنولوژی، اقتصادی و ... انجام‌پذیر است؟ آیا بهتر است یک سیستم موجود را بخریم یا بسازیم؟ تخمین هزینه‌ها و ریسک‌ها چیست؟ و در نهایت تصمیم بر این که آیا روند پروژه را ادامه دهیم یا نه؟

---

[۱۳] vision
[۱۴] scope
[۱۵] business case
[۱۶] feasibility
[۱۷] prototype
[۱۸] proof of concept

همچنین در این فاز درگیر برنامه‌ریزی و مدیریت پروژه نیز می‌شویم.

۲-۱-۱-۱ پس‌شرط‌ها و مصنوعات Inception

مصنوعات قابل تحویل در این فاز را می‌توان در شکل۲-۲ مشاهده کرد. مهم‌ترین آن‌ها یک سند چشم‌انداز[۱۹] است که در آن نیازمندی‌های اصلی، قابلیت‌ها و محدودیت‌ها بیان شده است. همچنین در این فاز باید یک معماری برای پروژه متصور شد وگرنه پروژه در امکان‌سنجی رد شده و پروژه نباید ادامه پیدا کند.

Conditions of satisfaction	Deliverable
The stakeholders have agreed on the project objectives	A vision document that states the project's main requirements, features, and constraints
System scope has been defined and agreed on with the stakeholders	An initial use case model (only about 10% to 20% complete)
Key requirements have been captured and agreed on with the stakeholders	A project glossary
Cost and schedule estimates have been agreed on with the stakeholders	An initial project plan
A business case has been raised by the project manager	A business case
The project manager has performed a risk assessment	A risk assessment document or database
Feasibility has been confirmed through technical studies and/or prototyping	One or more throwaway prototypes
An architecture has been outlined	An initial architecture document

شکل ۲-۲: مصنوعات قابل تحویل در فاز inception

به دلیل این که امکان‌پذیری پروژه و ادامه آن قبل از آغاز مشخص نیست، منطقی است که این فاز کوتاه باشد تا اگر پروژه رد شود، هزینه و زمان زیادی هدر نرفته است.

[۱۹] vision document

۲-۱-۲ فاز تفصیل

این فاز نقطه عطف معماری پروژه است که در آن تعریف پروژه دقیق‌تر می‌شود و یک خط‌پایه معماری قابل‌اجرا[۲۰] ساخته می‌شود و همچنین برای ایجاد و استقرار[۲۱] پروژه برنامه‌ریزی دقیق‌تر انجام می‌شود. هدف فاز قبل فهم چیستی مسئله است درحالی‌که فاز تفصیل، قلمرو جواب را می‌کاود.

در این فاز، ارزیابی ریسک دقیق‌تر می‌شود، صفات کیفیت تعریف می‌شود، نیازمندی‌های وظیفه‌ای[۲۲] تا ۸۰% ثبت می‌شوند، یک برنامه‌ریزی مفصل برای فاز بعدی ایجاد می‌شود و در آخر یک مناقصه[۲۳] شامل منابع، زمان، تجهیزات، افراد و هزینه تنظیم و ارائه می‌شود.

در این فاز تمرکز روی جریان‌های کاری به این شکل است که در نیازمندی، حوزه سیستم و نیازمندی‌ها دقیق‌تر می‌شود، در تحلیل، چیزی که قرار است ساخته شود دایر می‌شود، در طراحی یک معماری پایدار ساخته می‌شود، در پیاده‌سازی یک خط‌پایه معماری قابل اجرا ساخته می‌شود و در تست، خط‌پایه ساخته شده تست می‌شود.

بعد از فاز تفصیل، ریسک‌های پروژه باید از بین رفته باشند، معماری و واسطکاربری باید توسط مشتری و مدیرها تایید شده باشد، مولفه‌هایی که از لحاظ فنی دشوار هستند باید پیاده‌سازی شده باشند، تخمین هزینه باید نهایی شده باشد، مستندات راهنمای کاربر مقدماتی باید ساخته و تحلیل شده باشند.

۱-۲-۱-۲ پس‌شرط‌ها و مصنوعات Elaboration

در این فاز، یک خط‌پایه معماری قابل اجرا ساخته شده و همچنین با ذی‌نفعان مبنی بر ادامه پروژه توافق شده و سند مربوطه ایجاد می‌شود. سایر اصلاحات و مصنوعات در شکل۲-۳ قابل مشاهده است.

۳-۱-۲ فاز ساخت

این فاز نقطه‌عطف قابلیت‌های عملیاتی اولیه[۲۴] است که در آن محصول تا حدی که بتوان برای اولین بار به کاربر نهایی ارائه داد، ساخته می‌شود.

هدف این فاز این است که به صورت تکراری مصنوعاتی که قبلا ساخته شده‌اند را تکامل داده و به سیستم هدف برسانیم. تمام نیازمندی‌ها (۲۰% باقی مانده)، تحلیل و طراحی تکمیل می‌شوند. خط‌پایه معماری قابل اجرا که در فاز قبل ساخته شد، تکامل پیدا کرده و به سیستم نهایی تبدیل می‌شود.

---

[۲۰] executable architectural baseline
[۲۱] deployment
[۲۲] functional requirements
[۲۳] bid
[۲۴] initial operational capability milestone

Conditions of satisfaction	Deliverable
A resilient, robust executable architectural baseline has been created	The executable architectural baseline
The executable architectural baseline demonstrates that important risks have been identified and resolved	UML static model UML dynamic model UML use case model
The vision of the product has stabilized	Vision document
The risk assessment has been revised	Updated risk assessment
The business case has been revised and agreed with the stakeholders	Updated business case
A project plan has been created in sufficient detail to enable a realistic bid to be formulated for time, money, and resources in the next phases	Updated project plan
The stakeholders agree to the project plan	
The business case has been verified against the project plan	Business case
Agreement is reached with the stakeholders to continue the project	Sign-off document

شکل ۲-۳: مصنوعات قابل تحویل در فاز elaboration

۲-۱-۳-۱ پس‌شرط‌ها و مصنوعات Construction

محصولاتی که در این فاز ساخته یا تکمیل می‌شوند در شکل۲-۴ قابل مشاهده‌اند.

Conditions of satisfaction	Deliverable
The software product is sufficiently stable and of sufficient quality to be deployed in the user community	The software product The UML model Test suite
The stakeholders have agreed and are ready for the transition of the software to their environment	User manuals Description of this release
The actual expenditures vs. the planned expenditures are acceptable	Project plan

شکل ۲-۴: مصنوعات قابل تحویل در فاز construction

۲-۱-۴ فاز انتقال

این فاز نقطه‌عطف انتشار[۲۵] محصول است که در آن، محصول ساخته شده به محیط کاربر منتقل می‌شود که این شامل ساخت، تحویل، آموزش و برنامه‌ریزی برای پشتیبانی و نگهداری محصول می‌باشد.

[۲۵] release

هدف این فاز، استقرار نهایی محصول نرم‌افزاری هست که در انتهای فاز قبل ساخته‌شده است. در این فاز تست بتا و پذیرش[۲۶] انجام‌شده و ایرادها برطرف می‌شوند. محیط کاربر برای نرم‌افزار جدید آماده می‌شود. تغییرات لازم در نرم‌افزار انجام می‌شود تا در محیط کاربر عملیاتی شود و مشکلات پیش‌بینی نشده استقرار حل شوند. مستندات مختلف مانند راهنمای کاربر ساخته می‌شوند. به کاربر مشاوره ارائه می‌شود و همچنین بازبینی پساپروژه[۲۷] انجام می‌شود.

در این پروژه جریان‌کاری نیازمندی معنی ندارد، مدل تحلیل در صورت نیاز بروزرسانی می‌شود، مدل طراحی در صورت بروز مشکلات بروز می‌شود، در پیاده‌سازی تغییرات نرم‌افزار برای محیط کاربر انجام می‌شود و همچنین تست بتا و تست پذیرش در محیط کاربر انجام می‌شوند.

۲-۱-۴-۱ پس‌شرط‌ها و مصنوعات Transition

در این فاز محصول نرم‌افزاری ارائه شده و سایر موارد در شکل۲-۵ قابل مشاهده‌اند.

Conditions of satisfaction	Deliverable
Beta testing is completed, necessary changes have been made, and the users agree that the system has been successfully deployed The user community is actively using the product	The software product
Product support strategies have been agreed on with the users and implemented	User support plan Updated user manuals

شکل ۲-۵: مصنوعات قابل تحویل در فاز transition

۲-۲ نقاط قوت و ضعف

نقاط قوت

- فرایند تکراری-افزایشی.
- فرایند به خوبی مستندسازی شده.

---

[۲۶] acceptance
[۲۷] post-project

- براساس مدل‌سازی وظیفه‌ای[۲۸]، رفتاری[۲۹] و ساختاری[۳۰] قلمرو مسئله و سیستم.
- قابلیت ردیابی[۳۱] از طریق Use-Case پشتیبانی می‌شود.
- تا حد خوبی بی‌درز (البته سکسکه‌هایی وجود دارد برای مثال در تبدیل هاuse-case به نمودارهای توالی[۳۲]).
- فرایند معماری–محور که توصیف زودهنگام یک طرح معماری را ایجاب می‌کند.
- به قابلیت سفارشی‌سازی پرداخته‌شده است.
- ایجاد مبتنی بر ریسک، با هدف کاهش ریسک‌ها قبل از انجام تکالیف.
- پشتیبانی از انواع مدل‌سازی در تمام سطوح (از قلمرو مسئله تا اشیا و منطقی تا فیزیکی).
- زبان مدل‌سازی غنی به ویژه در مدل‌سازی ساختاری و رفتاری.
- تاحدی از روش‌های صوری پشتیبانی میکند (از طریق OCL).
- پشتیبانی قوی ابزاری.
- نسبت به RUP بسیار ساده‌تر است.
- برخلاف RUP، تحلیل و طراحی جریان‌های کاری جدا از هم هستند که هرکدام واحدهای کاری و محصولات بخصوص دارند.

نقاط ضعف

- فرایند پیچیده (البته نسبت به RUP بسیار ساده‌تر است).
- فرایند برای افراد می‌تواند گیج‌کننده باشد و طبیعت تکراری–افزایشی بودن آن نیز بیشتر پیچیده می‌کند.
- پیکربندی آن دشوار است.
- فاز مراقبت و نگهداری[۳۳] ندارد.
- تعداد زیاد مدل.
- تعصب به استفاده از UML. به خصوص این که UML بی‌نقص نیست و می‌تواند مشکل ناسازگاری مدل را تشدید کند.

---

[۲۸] functional
[۲۹] behavioural
[۳۰] structural
[۳۱] traceability
[۳۲] sequence diagram
[۳۳] maintenance

- فعالیت‌های مدل‌سازی کسب و کار، استقرار و مدیریت، نقش‌های محوری خود را در قالب جریان‌های کاری از دست داده‌اند.

فصل ۳

معیارهای ارزیابی

یک متدولوژی از دو بخش اصلی زبان مدل‌سازی[۱] و فرایند[۲] تشکیل می‌شود. برای ارزیابی متدولوژی‌ها با رویکرد مبتنی بر معیار[۳]، باید برای دو بخش تشکیل دهنده متدولوژی، معیارهایی تعریف کرد که در ادامه فهرست می‌شوند [۵].

۳-۱ معیارهای فرایند

بخش فرایند متدولوژی شامل ۱۱ معیار است.

- تعریف[۴]
- پوشش چرخه‌حیات عمومی ایجاد نرم‌افزار
- پشتیبانی از فعالیت‌های چتری[۵]
- بی‌درزی[۶] و گذر هموار[۷]
- مبتنی بودن بر نیازمندی‌ها (وظیفه‌ای و غیر وظیفه‌ای)
- تست‌پذیر[۸] و ملموس[۹] بودن مصنوعات و قابلیت رهگیری به نیازمندی‌ها

[۱] modeling language
[۲] process
[۳] criteria-based
[۴] definition
[۵] umbrella activities
[۶] seamlessness
[۷] smoothness of transition
[۸] testable
[۹] tangible

- تشویق مشارکت فعال کاربر
- قابل استفاده بودن[۱۰] و قابل استفاده بودن به صورت موثر و کارا[۱۱]
- قابل مدیریت بودن پیچیدگی
- قابلیت‌های گسترش، مقیاس‌پذیری، پیکربندی و انعطاف
- حوزه کاربرد

۳-۱-۱ تعریف

یک متدولوژی باید به خوبی تعریف و مستندسازی شده باشد و مشخصا باید ویژگی‌های زیر در توصیف یک متدولوژی باشند:

- جامع و پوشا
- روشن و واضح
- منطقی
- صحیح و دقیق
- با جزئیات
- سازگار و بدون وجود تناقض در قسمت‌های مختلف

همچنین توصیف یک متدولوژی باید شامل موارد زیر باشد:

- چرخه‌حیات و واحدهای کاری[۱۲] داخل چرخه باید مشخص شده باشند.
- تولید کننده‌ها[۱۳]
- زبان مدل‌سازی
- محصولات
- تکنیک‌ها و قواعد. تکنیک‌ها بیان کننده چگونگی اجرای واحدهای کاری هستند.
- موارد مربوط به فعالیت‌های چتری[۱۴]

---

[۱۰] practicable
[۱۱] practical
[۱۲] work-units
[۱۳] producers
[۱۴] فعالیت های چتری در متدولوژی های ایجاد نرم افزار، مربوط به مرحله خاصی نیستند و کل چرخه‌حیات را پوشش می‌دهند.

توصیف مطلوب متدولوژی به صورت فرایند–محور[۱۵] است که در آن چرخه‌حیات، فازها و فعالیت‌های ذیل فازها یعنی مرحله‌ها[۱۶] و تکالیف[۱۷] گفته می‌شود و نقش‌ها و محصولات ذیل فعالیت‌ها بیان می‌شوند. یعنی داخل هر فعالیت مشخص می‌شود که چه محصولاتی تولید می‌شوند و چه نقش‌هایی درگیر هستند و این کار چگونه انجام می‌شود.

دو نحوه توصیف دیگر یعنی محصول–محور[۱۸] و نقش–محور[۱۹] می‌توانند به عنوان مکمل در کنار توصیف فرایند–محور وجود داشته باشند. برای مثال ساختار کتاب USDP این چنین است. ولی اگر صرفا یکی از توصیفات محصول–محور یا نقش–محور وجود داشته باشد، این نشان از ضعف در توصیف است.

۳-۱-۲ پوشش چرخه‌حیات عمومی ایجاد نرم‌افزار

ترجیح این است که یک متدولوژی تمام چرخه‌حیات را پوشش دهد[۲۰]. پوشش بهتر چرخه‌حیات، استفاده مهندس نرم‌افزار را آسان‌تر می‌سازد.
چرخه‌حیات عمومی ایجاد نرم‌افزار:

- تعریف[۲۱]
  - کاوش قلمرو مسئله و مدل‌سازی آن
  - استخراج نیازمندی‌ها[۲۲]
  - تحلیل امکان‌پذیری[۲۳]
- ایجاد[۲۴]
  - طراحی معماری
  - طراحی تفصیلی: اجزا سیستم و رفتار آن‌ها با جزئیات مشخص می‌شود و چگونگی تعامل اشیا با یکدیگر برای تحقق نیازمندی‌ها نشان داده می شود.
  - برنامه‌نویسی
  - تست

---

[۱۵] process-centered
[۱۶] stage
[۱۷] task
[۱۸] product-centered
[۱۹] role-centered
[۲۰] full lifecycle
[۲۱] definition
[۲۲] requirements elicitation
[۲۳] feasibility analysis
[۲۴] development

- استقرار

- مراقبت و نگهداری[۲۵]
  - انواع آن مانند نگهداری تصحیحی، تکمیلی، تطبیقی و اجتنابی[۲۶]

۳-۱-۳ پشتیبانی از فعالیت‌های چتری

در معیار اول بیان شد که متدولوژی باید در توصیفش به موارد مربوط به فعالیت‌های چتری بپردازد و همچنین خود متدلوژی نیز باید فعالیت‌های چتری را پشتیبانی کند. ممکن است یک متدلوژی پشتیبانی خوبی از فعالیت‌های چتری داشته باشد ولی در توصیف به آن موارد اشاره نکند که این یعنی در معیار اول مشکل دارد.

فعالیت‌های چتری شامل موارد زیر می‌شود:

- مدیریت ریسک
  - فعالیت‌های مربوط به ارزیابی ریسک و کاهش ریسک باید در فرایند دیده شوند.
  - تکنیک‌های مفید: تحلیل مقدماتی، ساخت نمونه‌اولیه، برنامه‌ریزی براساس ریسک، فرایند تکراری- افزایشی، دخالت فعالانه کاربر، ترخیص زودهنگام، صحت‌سنجی و اعتبارسنجی پیوسته، بازبینی تکراری، یکپارچه‌سازی مداوم[۲۷].
- مدیریت پروژه
  - باید فعالیت‌های برنامه‌ریزی، زمان‌بندی و نظارت و کنترل در متدولوژی تعبیه شوند.
  - برنامه‌ها و زمان‌بندی را باید به صورت مرتب مورد بازنگری قرار دهیم، اصلاح و بروز کنیم.
  - مدیریت پروژه حتما باید مدیریت تیم را هم لحاظ کند.
- تضمین کیفیت
  - بیشتر روش‌هایی که در ریسک را مدیریت می‌کنند، توجه به کیفیت نیز دارند.
  - باید به صورت صریح روش‌های سنجش و ارتقا کیفیت در متدولوژی تعبیه شده باشد.
  - تکنیک‌های مفید: بازبینی فنی مکرر، تکنیک طراحی مبتنی بر قرارداد، صحت‌سنجی و اعتبارسنجی پیوسته و تکنیک‌ها و استراتژی‌های که برای بهبود ردیابی به نیازمندی‌ها استفاده می‌شوند.

---

[۲۵] maintenance
[۲۶] corrective, perfective, adaptive, and preventive maintenance
[۲۷] continuous integration

۳-۱-۴ بی‌درزی و همواری انتقال

درز[۲۸] به این معنی است که از یک پارادایم به پارایم دیگر رفته و شکاف پارادایمی یا تغییر پارادایم[۲۹] داشته باشیم. پارادایم‌ها و الگوهای تصور و ادراک، متفاوت‌اند. بنابراین وقتی از یک پارادایم به پارادایم دیگر میرویم، این کار خطاخیز است و احتمال بالای از بین رفتن اطلاعات وجود دارد. بی‌درزی به این معنی است که در زنجیره مدل‌سازی و فعالیت‌ها تغییر پارادایم نداشته باشیم.
همواری انتقال[۳۰] یعنی فعالیت‌ها باید ادامه طبیعی یکدیگر باشند و این‌گونه نباشد که در فعالیت بعدی تلاش عمده انجام شود تا نتیجه فعالیت قبل را به دیگری تبدیل کنیم و یکدفعه مجموعه‌ای از مدل‌های جدید بسازیم.

۳-۱-۵ مبتنی بودن بر نیازمندی‌ها (وظیفه‌ای و غیر وظیفه‌ای)

در متدولوژی، فعالیت‌ها باید مبتنی بر نیازمندی‌ها انجام شوند. نیازمندی‌های وظیفه‌ای و غیر وظیفه‌ای باید این ۳ ویژگی را داشته باشند:

- اوایل فرایند یعنی در ابتدای پروژه و شروع اجرای متدولوژی باید استخراج و ثبت شوند.
- باید به تنهایی و به صورت مستقل مدل شوند.
- از آن‌ها به عنوان مبنا استفاده کنیم برای کارهای بعدی مثل تحلیل، طراحی، پیاده‌سازی و تست.

امروزه در برخی متدولوژی‌ها براساس نیازمندی‌ها تحلیل انجام می‌شود و براساس نیازمندی‌ها طراحی انجام شده و براساس نیازمندی‌ها پیاده‌سازی انجام می‌شود. در هرکدام از مراحل به نتایج مراحل قبلی توجه می‌کنند ولی مبنای کار آن‌ها نیازمندی است که به این متدولوژی‌ها Requirements-Driven گفته می‌شود. در این متدولوژی‌ها رهگیری به نیازمندی‌ها به صورت مستقیم وجود دارد.
همچنین بیان می‌شود که نیازمندی چیز ثابتی نبوده و بروز یابنده است و باید اجازه دهیم به مرور تکامل پیدا کند.

۳-۱-۶ آزمون‌پذیری، ملموس بودن و قابلیت رهگیری به نیازمندی‌ها

آزمون‌پذیری[۳۱] محصولات به پیچیدگی آن‌ها وابسته است. محصولات باید ساده، به تعداد کم و قابل فهم باشند و روابط بین آن‌ها حداقلی باشد. محصولات باید مکمل یکدیگر در زمینه فرایند باشند و صرفا یکدیگر را تزئین

[۲۸] seam
[۲۹] paradigm shift
[۳۰] smoothness of transition
[۳۱] testability

نکنند.

ملموس بودن[۳۲] محصولات به مخاطب آن‌ها بستگی دارد. از دید کاربر محصولات باید قابل اجرا بوده و نحو و معنای آن‌ها برای کاربر قابل درک باشند. از دید ایجادکننده نرم‌افزار، محصولاتی که ساخته می‌شوند باید به صورت واضح در فرایند مفید باشند.

محصولات باید از طریق تحقق مستقیم یا غیرمستقیم نیازمندی‌ها قابل رهگیری[۳۳] باشند یا از طریق سناریوهای ارزیابی مبتنی بر نیازمندی این رهگیری به عمل آید.

۳-۱-۷ تشویق مشارکت فعال کاربر

این که یک متدولوژی مشارکت فعال کاربران[۳۴] را تشویق کند، برای مدیریت ریسک و تضمین کیفیت حیاتی است. از تکنیک‌های مفید در این زمینه می‌توان به حضور نماینده کاربر یا کاربر سفیر[۳۵] در تیم و همچنین جلسات برنامه‌ریزی و مرور با حضور کاربر اشاره کرد.

۳-۱-۸ قابلیت اجرا و قابلیت اجرا به صورت کارا

قابلیت اجرا[۳۶] یعنی قابلیت به‌کارگیری و اجرای فرایند که می‌تواند وابسته به پیچیدگی فرایند یا ماهیت پروژه هدف وابسته باشد.

قابلیت اجرا به صورت کارا[۳۷] یعنی فرایند قابل استفاده به صورت موثر و بهینه باشد که موارد زیر می‌تواند روی کارا بودن تاثیر بگذارد:

- پیچیدگی واحدهای فرایند.
- تکالیفی که ایجادکنندگان را از فعالیت‌های اصلی منحرف می‌سازند یا آن‌ها را با جزئیات غیرضروری درگیر می‌سازند. تمرکز ایجادکنندگان با تکنیک‌هایی همچون جلسات مدیریت تیم، مدل‌های مبتنی بر نیازمندی‌ها و معماری سیستم می‌تواند محقق شود.
- وابستگی به تکنیک‌ها و استراتژی‌های مستعد خطا.
- وابستگی به ابزارها و فناوری‌های بخصوص.
- کمبود استراتژی مدیریت پروژه کافی.

---

[۳۲] tangibility
[۳۳] traceability to requirements
[۳۴] encouragement of active user involvement
[۳۵] ambassador user
[۳۶] practicability
[۳۷] practical

۳-۱-۹ قابلیت مدیریت پیچیدگی

پیچیدگی واحدهای کاری باید قابل مدیریت باشند. دو تکنیک که برای این منظور استفاده می‌شوند:

- جزء بندی[۳۸]
- لایه‌بندی[۳۹]

۳-۱-۱۰ قابلیت‌های گسترش، مقیاس‌پذیری، پیکربندی و انعطاف

قابلیت گسترش[۴۰] یعنی فرایند یک متدولوژی باید به صورت یک هسته قابل توسعه باشد که نقاط گسترش[۴۱] و مکانیزم‌ها به صورت صریح بیان شده باشند.
مقیاس‌پذیری[۴۲] یعنی فرایند باید بر روی پروژه‌ها با سطوح بحرانیت و اندازه‌های مختلف قابل اجرا باشد. منظور از اندازه صرفا تعداد خطوط کد نیست و منظور تعداد افرادی که به صورت همزمان در پروژه دخیل هستند و پروژه‌های با اندازه بالا نیاز به مدیریت پروژه و مدیریت تیم خوب دارند. همچنین منظور از سطوح بحرانیت سطوح C D E L[۴۳] است. در کل متدولوژی‌هایی که در آن‌ها مدل‌سازی جدی است، مقایس‌پذیری بالایی نیز دارند.
قابلیت پیکربندی[۴۴] یعنی بتوان فرایند را در ابتدای پروژه پیکربندی کرد تا متناسب با موقعیت پروژه هدف باشد.
انعطاف‌پذیری[۴۵] یعنی فرایند باید در حین اجرا قابل تنظیم باشد. تکنیک‌های مفید برای این منظور عبارتند از جلسات بازبینی فرایند تکراری و بازنگری‌های مبتنی بر بازخورد.

۳-۱-۱۱ حوزه کاربرد

حوزه کاربرد یک فرایند به زمینه استفاده موردنظر آن بستگی دارد. البته به عنوان معیار حداقلی، باید بتواند برای حوزه سیستم‌های اطلاعاتی استفاده شود.

---

[۳۸] partitioning
[۳۹] layering
[۴۰] extensibility
[۴۱] extension points
[۴۲] scalability
[۴۳] life, essential money, discretionary money, comfort critical
[۴۴] configurability
[۴۵] flexibility

۳-۲ معیارهای زبان مدل‌سازی

بخش زبان مدل‌سازی شامل دو معیار است.

- پشتیبانی از مدل‌سازی شی‌گرای سازگار، دقیق و بدون ابهام.
- تامین استراتژی‌ها و تکنیک‌ها برای مقابله با ناسازگاری‌ها و مدیریت پیچیدگی مدل.

۳-۲-۱ پشتیبانی از مدل‌سازی شی‌گرای سازگار، دقیق و بدون ابهام

دیدگاه‌های مدل‌سازی متنوع باید پوشش داده شود:

- ساختاری[۴۶]: بیان‌کننده اجزاء سیستم.
- وظیفه‌ای[۴۷]: بیان‌کننده کارهای سیستم بدون در نظر گرفتن ترتیب.
- رفتاری[۴۸]: بیان‌کننده کارهای سیستم با درنظر گرفتن ترتیب.

مدل‌سازی منطقی تا فیزیکی می‌تواند باشد یعنی قلمرو فرایند کسب و کار یا مسئله تا قلمرو جواب و تا قلمرو پیاده‌سازی.

سطوح درشت‌دانگی و انتزاع متعدد مانند سازمان، سیستم، زیرسیستم، بین‌اشیا و داخل‌شی.

امکانات مدل‌سازی صوری و غیرصوری.

۳-۲-۲ ارائه راهبردها و تکنیک‌هایی برای رفع ناسازگاری‌ها و مدیریت پیچیدگی

زبان‌های مدل‌سازی می‌توانند با ارائه معانی رسمی[۴۹] که وابستگی‌ها و محدودیت‌های مدل را تعریف می‌کنند، فرایند بررسی سازگاری را تسهیل کنند.

زبان‌های مدل‌سازی باید شامل سازوکارهایی برای مدیریت پیچیدگی باشند. مانند بسته‌ها[۵۰] و مولفه‌ها[۵۱] در UML.

---

[۴۶] structural
[۴۷] functional
[۴۸] behavioural
[۴۹] semantic
[۵۰] package
[۵۱] component

فصل ۴

مقایسه متدولوژی‌های BON و USDP

در این قسمت، دو متدولوژی BON و USDP که در فصول قبل معرفی شدند، با رویکرد مبتنی بر معیار[۱] ارزیابی و مقایسه می‌شوند.

۴-۱ فرایند

در این بخش، قسمت فرایندی متدولوژی‌ها مبتنی بر معیارهایی که در فصل گذشته معرفی شدند، مقایسه و ارزیابی می‌شوند.

۴-۱-۱ تعریف

به صورت کلی تعریف و مستندسازی متدولوژی USDP بسیار مفصل‌تر و کامل‌تر از متدولوژی BON است که در سال ۱۹۹۲ در یک مقاله توصیف شده است (اگرچه در سال ۱۹۹۵ توصیف این متدولوژی در قالب یک کتاب بسیار مفصل‌تر شد).

۴-۱-۱-۱ چرخه‌حیات و واحدهای کاری

در **متدولوژی BON** همان‌طور که در فصل ۱ توصیف شد، چرخه‌حیات و واحدهای کاری به صورت قابل‌قبولی توصیف شده‌اند. به صورت خلاصه چرخه‌حیات آن از ۹ تکلیف که در ۳ فاز تقسیم‌بندی شده‌اند، تشکیل شده

[۱] criteria-based

است که ۶ تکلیف اول مربوط به فعالیت‌های تحلیل و ۳ تکلیف مربوط به فعالیت‌های طراحی هستند. همان‌طور که بیان شد، ترتیب انجام تکالیف می‌تواند متغیر باشد با این شرط که تمام آن‌ها باید انجام شوند. جزئیات مربوط به هر تکلیف نیز مشخص شده است.

در توصیف **متدولوژی USDP** بیان شده است که چرخه‌حیات آن از ۴ فاز ترتیبی آغاز، تفصیل، ساخت و انتقال تشکیل شده است که هرکدام از آن‌ها شامل تکرارهایی می‌شوند. هر تکرار از ۵ جریان‌کاری نیازمندی‌ها، تحلیل، طراحی، پیاده‌سازی و تست تشکیل شده است که تاکید روی هرکدام بستگی به هدف فاز دارد. در فاز آغاز مواردی مانند چشم‌انداز، حوزه و توجیه اقتصادی پروژه تعریف می‌شود. در فاز تفصیل، شناخت محصول تکمیل‌تر شده، معماری مبنای قابل‌اجرا ساخته شده و نیازمندی‌های دارای ریسک پیاده‌سازی می‌شوند. در فاز ساخت، نیازمندی‌ها پیاده‌سازی شده و محصول تا حدی که قابل ارائه به مشتری باشد، ساخته می‌شود. در فاز انتقال، محصول به محیط کاربر انتقال پیدا کرده و تحویل داده می‌شود همچنین تست و آموزش و برنامه‌ریزی برای پشتیبانی و نگهداری انجام می‌شود. تمام جزئیات چرخه‌حیات و واحدهای کاری در توصیف بیان شده است.

هردو متدولوژی مورد بحث، چرخه‌حیات و واحدهای کاری را در توصیف لحاظ کرده‌اند.

۴-۱-۱-۲ تولیدکنندگان (نقش‌ها)

در **BON** صراحتا درباره نقش‌های دخیل در فعالیت‌ها و تولیدکننده‌های محصولات هر فعالیت مطلبی بیان نشده است.

در توصیف **USDP** به عنوان مکمل توصیف فرایند–محور، توصیف نقش–محور نیز دیده می‌شود که در آن نقش‌ها و افراد دخیل در فعالیت‌ها و مصنوعات، به خوبی توصیف شده‌اند.

بنابراین USDP در این معیار بسیار بهتر عمل می‌کند.

۴-۱-۱-۳ زبان مدل‌سازی

BON زبان مدل‌سازی بخصوصی را معرفی کرده است و انواع محصولات موجود در متدولوژی با آن زبان ایجاد می‌شوند که شامل جداول، نمودارها و مستندات می‌شوند. درباره زبان برنامه‌سازی که خود جزوی از زبان مدل‌سازی است، هرچند BON اجباری برای استفاده از زبان بخصوصی نکرده است ولی رابطه‌ای با زبان Eiffel دارد و برخی موارد در آن زبان ساده‌تر پیاده‌سازی می‌شوند. موارد مربوط به نمادگان استفاده شده در مصنوعات، در توصیف آن مشخص شده‌اند.

در **USDP** از UML به عنوان زبان مدل‌سازی استفاده می‌شود و استفاده از زبان‌های دیگر مجاز نیست. برای برنامه‌سازی نیز زبان بخصوصی اجبار نشده است. موارد مربوط به زبان مدل‌سازی در این متدولوژی کاملا شرح داده شده‌اند.

هر دو متدولوژی در توصیف خود، موارد زبان مدل‌سازی را بیان کرده‌اند.

۴-۱-۱-۴ محصولات

در **BON** تمام محصولاتی که در فعالیت‌ها ساخته می‌شوند تحت عنوان مصنوعات قابل‌تحویل[۲] به صراحت بیان شده‌اند که در فصل ۱ اشاره شد. از جمله آن‌ها می‌توان به نمودار سیستم، نمودار خوشه و نمودار کلاس اشاره کرد.

در **USDP** همان‌طور که در فصل ۲ اشاره شد، محصولات متعددی طی فعالیت‌ها ساخته می‌شوند که به خوبی توصیف شده‌اند.

- فاز آغاز
  - سند چشم‌انداز که شامل نیازمندی‌های وظیفه‌ای و غیروظیفه‌ای و محدودیت های کاربر می‌باشد.
  - مدل Use-Case که نیازمندی‌های هسته را شامل می‌شود.
  - فرهنگ لغت پروژه که مفاهیم قلمرو مسئله را مشخص می‌کند.
  - برنامه‌ریزی پروژه اولیه که بر اساس آن امکان‌سنجی زمانی انجام می‌شود.
  - سند توجیه اقتصادی که امکان‌سنجی مالی و نتایج آن را مشخص می‌کند.
  - سند ارزیابی ریسک.
  - تعدادی نمونه‌اولیه[۳] از نوع دور ریختنی با هدف امکان‌سنجی و مدیریت ریسک.
  - سند معماری اولیه که شامل معماری سطح بالای سیستم است.
- فاز تفصیل
  - یک معماری مبنای قابل‌اجرا[۴] که یک نمونه‌اولیه تکاملی است و باید با کیفیت مناسبی ساخته شود. همچنین در کنار آن مدل‌های ایستا، پویا و مدل Use-Case تکمیل شوند.
  - سند چشم‌انداز کامل می‌شود.
  - ارزیابی ریسک باید کامل شده و نهایی شده باشد.

---

[۲] deliverable
[۳] prototype
[۴] executable architectural baseline

- توجیه اقتصادی باید بازبینی شده و با ذی‌نفع‌ها توافق شده باشد.
- برنامه‌ریزی پروژه با دقت کافی ایجاد می‌شود تا مناقصه واقع‌بینانه ارائه شود.
- سند Sign-Off که بیان توافق ذی‌نفع‌ها برای ادامه پروژه است.

- فاز ساخت

- محصول نرم‌افزاری که به صورت پیکربندی بوده و شامل تمام مدل‌ها و مجموعه تست‌ها می‌شود.
- مستندات راهنمای کاربر و توصیفات انتشار[۵] فعلی.
- برنامه‌ریزی پروژه.

- فاز انتقال

- محصول نرم‌افزاری به صورت پیکربندی کامل.
- برنامه‌ریزی پشتیبانی کاربر.
- مستندات راهنمای کاربر بروز شده.

بنابراین هردو متدولوژی در توصیف خود به محصولات اشاره خوبی کرده‌اند.

۴-۱-۱-۵ تکنیک‌ها و قواعد

در **BON** برای ۹ تکلیفی که بیان شده است، تا حدی جزئیات و چگونگی انجام فعالیت‌ها و قواعد مشخص شده‌اند ولی این توصیف در کتاب BON که بعدها چاپ شد دقیق‌تر و مفصل‌تر شده است.

USDP توصیف مفصلی دارد که تکنیک‌ها و رویه‌های عملی مناسب برای فعالیت‌های مختلف به خوبی بیان شده‌اند. برای مثال استخراج نیازمندی‌ها، یافتن کلاس‌های طراحی، مدیریت ریسک وغیره به خوبی توضیح داده شده‌اند.

هردو چگونگی و قواعد انجام کارها را بیان کرده‌اند ولی USDP توصیف مفصل‌تری دارد.

۴-۱-۱-۶ موارد مربوط به فعالیت‌های چتری

در مقاله **BON** به صراحت اشاره‌ای به فعالیت‌های چتری نشده است ولی بررسی نظر کاربر نهایی در معیار پذیرش برخی تکالیف دیده می‌شود که می‌توان آن را در جهت مدیریت ریسک در نظر گرفت. همچنین اشاره شده است با این که تکالیف به صورت ترتیبی بیان شده‌اند، در عمل به صورت تکراری اجرا می‌شوند. درباره مدیریت

[۵] release

تیم نیز مطلبی بیان نشده است. در مورد تضمین کیفیت، به دلیل استفاده از تکنیک طراحی با قرارداد[۶]، قابلیت اطمینان افزایش پیدا می‌کند که در کیفیت تاثیر می‌گذارد.

در **USDP** برای مدیریت ریسک، در فاز آغاز، فعالیت‌ها با هدف امکان‌سنجی انجام می‌شوند و سند ارزیابی ریسک و توجیه اقتصادی ساخته می‌شوند. نمونه‌های اولیه برای تحلیل نیازمندی‌ها و امکان‌سنجی ساخته می‌شوند، مشارکت فعال کاربر، اعتبارسنجی مداوم و تکراری–افزایشی بودن فرایند همگی برای مدیریت ریسک در توصیف متدولوژی بیان شده‌اند. برای مدیریت پروژه، در هر فاز برای تکرارهای آینده نزدیک برنامه‌ریزی دقیق و برای کلیت پروژه برنامه‌ریزی تخمینی انجام می‌شود و مکرر مورد بازبینی قرار می‌گیرد. برای تضمین کیفیت، صحت‌سنجی و اعتبارسنجی[۷] مستمر انجام شده، به دلیل مبتنی بودن بر Use-Case، ردیابی به نیازمندی‌ها به خوبی شکل می‌گیرد. معیارهای کیفیت مانند نرخ کشف خطا و تراکم خطای قابل‌قبول تعریف می‌شوند و تست نیز همراه با پیاده‌سازی انجام می‌شود و به این موارد در توصیف به خوبی اشاره شده‌اند.

بنابراین توصیفات مربوط به فعالیت‌های چتری در متدولوژی USDP بهتر عنوان شده‌اند.

۴-۱-۱-۷ چگونگی تعریف

متدولوژی USDP توصیف اصلی یعنی فرایند–محور را شامل می‌شود و البته توصیفات مکمل محصول و نقش–محور برای تبیین جزئیات از دیدگاهی متفاوت برای تکمیل دیدگاه اصلی، بکار برده شده‌اند بنابراین در این معیار USDP بهتر عمل می‌کند.

۴-۱-۲ پوشش چرخه‌حیات عمومی ایجاد نرم‌افزار

۴-۱-۲-۱ کاوش و مدل‌سازی قلمرو مسئله

در متدولوژی **BON** فاز جمع‌آوری اطلاعات تحلیل که از سه تکلیف مشخص کردن مرز سیستم، فهرست کردن کلاس‌های کاندید، انتخاب و گروه‌بندی کلاس‌ها در خوشه‌ها تشکیل شده، به کاوش قلمرو مسئله و مدل‌سازی می‌پردازد.

در **USDP** یکی از اهداف فاز آغاز، شناخت چیستی مسئله است. بنابراین ذیل این فاز در تکرارهای متعدد، جریان‌های کاری نیازمندی‌ها و تحلیل پررنگ‌تر هستند و قلمرو مسئله تا حدی مدل‌سازی می‌شود. در فاز تفصیل، قلمرو مسئله بیشتر کاوش شده و مدل‌سازی دقیق‌تر می‌شود و شامل انواع مدل ساختاری، رفتاری و وظیفه‌ای می‌شود. برای مثال نمودار کلاس، نمودار شی، نمودار بسته‌بندی، نمودار فعالیت و غیره.

[۶] design by contract
[۷] verification & validation

۴-۱-۲-۲ استخراج نیازمندی‌ها

در **BON** به صراحت به فعالیت مجزا برای استخراج نیازمندی‌ها اشاره نشده است.

در **USDP** در فاز آغاز، نیازمندی‌های اساسی که ۲۰% کل نیازمندی‌ها را تشکیل می‌دهند، استخراج می‌شوند و حوزه سیستم را مشخص می‌کنند. در فاز تفصیل، تا ۸۰% نیازمندی‌ها استخراج می‌شود و مابقی نیازمندی‌ها طی فاز ساخت استخارج می‌شوند. در طی تکرارها نیز جریان کاری بخصوص نیازمندی‌ها وجود دارد و تمام کارها به صورت مبتنی بر Use-Case انجام می‌شوند.

۴-۱-۲-۳ تحلیل امکان‌پذیری

در **BON** فعالیتی برای تحلیل امکان‌پذیری و بررسی ریسک‌های جدی دیده نمی‌شود.

در **USDP** یکی از اصلی‌ترین اهداف فاز آغاز، تحلیل امکان‌پذیری و شناسایی ریسک‌های مهم است. در این راستا، فرد خبره همراه با تیم خود در طی مدت کوتاه حداکثر ۴ هفته، با تحلیل چیستی سیستم و تکنیک‌هایی مانند ساخت نمونه‌اولیه[۸] هم برای امکان‌سنجی فنی و هم برای ارزیابی درک خود از نیازمندی‌ها، انجام شده و سعی می‌کنند یک معماری برای پروژه متصور شوند و در انتها تصمیم بر ادامه دادن یا ندادن پروژه گرفته می‌شود. در انتهای فاز تفصیل باید به تمام ریسک‌ها پرداخته شده باشد.

۴-۱-۲-۴ طراحی معماری

در **BON** و در فاز اول، در تکلیف انتخاب کلاس‌ها و گروه‌بندی در خوشه‌ها، یک معماری ایستای اولیه ساخته می‌شود که مجموعه‌ای از نمودارها است که روابط بین کلاس‌ها و خوشه‌ها را در سیستم توصیف می‌کند و طی تکالیف مربوط به طراحی، آن مدل را تکامل داده و بروزرسانی می‌کند. بدین شکل زیرسیستم‌ها نیز مشخص می‌شوند.

در **USDP** در فاز آغاز، یک معماری اولیه سطح‌بالا عمدتا با هدف امکان‌سنجی ساخته می‌شود و در طی فاز تفصیل، تکمیل شده و تثبیت می‌شود و مبنای کارها قرار می‌گیرد. در این متدولوژی، طراحی نیز جدی است و تکنیک‌های متعدد مانند استفاده از UML و ایجاد نمودارهای ساختاری و رفتاری متعدد تجویز شده است و در نهایت طی این فعالیت‌ها به یک معماری مبنای قابل اجرا[۹] می‌رسد.

---

[۸] prototype

[۹] executable architectural baseline

۴-۱-۲-۵ طراحی تفصیلی

در **BON** در سه تکلیف طراحی یعنی دقیق‌تر سازی سیستم، عمومی‌سازی و تکمیل سیستم اقدام به دقیق‌تر سازی و طراحی جزئی مدل‌های پیشین می‌کند که در تکالیف فاز توصیف یعنی تعریف کلاس‌ها، ترسیم رفتار سیستم و تعریف ویژگی‌های عمومی و فاز جمع‌آوری شامل تکالیف مشخص کردن مرز سیستم، فهرست کلاس‌های کاندید و انتخاب کلاس‌ها و گروه‌بندی در خوشه‌ها، ساخته شده بودند. مدل‌هایی مانند نمودارهای کلاس، نمودارهای ساخت، معماری ایستا و غیره که در فصل BON خلاصه شدند.

در **USDP** طراحی تفصیلی جزو فعالیت‌ها بوده و طی جریان‌کاری طراحی در فاز تفصیل، در حد نیاز برای پیاده‌سازی use-case های دارای ریسک و EAB و مابقی در طی فاز ساخت برای ایجاد نیازمندی‌های سیستم، انجام می‌شود که سنگین بوده و از تکنیک‌های مختلف استفاده شده و مدل‌های متعدد ایجاد می‌شوند مانند نمودار کلاس، نمودار مولفه و غیره که در فصل USDP خلاصه شدند.

۴-۱-۲-۶ برنامه‌نویسی

در متدولوژی **BON** مطلبی درباره فعالیت برنامه‌نویسی ارائه نشده است.

در **USDP** نظام مربوط به پیاده‌سازی وجود دارد که در تکرارهای مختلف در فاز آغاز برای ایجاد نمونه‌های اولیه دور ریختنی، در فاز دوم برای ایجاد نیازمندی‌های ریسکی و EAB و در فاز سوم برای ساخت اجزای سیستم، انجام می‌شود. پس از انتقال محصول به محیط کاربر نیز برای رفع مشکلات و اعمال تغییرات پیاده‌سازی دیده می‌شود.

۴-۱-۲-۷ آزمون

در **BON** اشاره‌ای به فعالیتی مربوط به آزمون نشده است.

در **USDP** در فاز تفصیل، معماری مبنای قابل‌اجرا و نیازمندی‌های ریسکی که پیاده‌سازی شده‌اند طی نظام تست، مورد آزمون قرار می‌گیرد. در فاز ساخت، طی تکرارهای متعدد، در نظام تست، موارد پیاده‌سازی شده مورد آزمون قرار می گیرند. در این فاز تست آلفا نیز داریم. همچنین در فاز انتقال، نسخه‌ای که به محیط کاربر منتقل شده است، مورد آزمون های متعدد از جمله تست سیستم، کارایی، امنیت، تست پذیرش و غیره قرار می‌گیرد. در تست آلفا، یک محیط شبیه‌سازی شده از محیط کاربر در محیط ایجاد ساخته می‌شود و کاربر آن را اجرا و بازخورد می‌دهد. در تست بتا، نرم‌افزار برای مشتریان بالقوه فرستاده شده و کاربران در محیط خود نصب کرده و بازخورد می‌دهند و تست پذیرش در نهایت که سیستم بطور کامل در محیط کاربر استقرار یافت،

انجام می‌شود.

۴-۱-۲-۸ استقرار

در متدولوژی **BON** درباره فعالیتی مربوط به استقرار صحبت نشده است.

در **USDP** برخلاف متدولوژی RUP جریان‌کاری بخصوصی برای استقرار در نظر گرفته نشده است ولی در طی جریان‌کاری پیاده‌سازی انجام می‌شود و در فاز انتقال نیز محیط کاربر برای استقرار محصول آماده می‌شود و تمهیدات لازم قبل و بعد از انتقال محصول به محیط کاربر طی جریان‌های کاری متعدد انجام می‌شود.

۴-۱-۲-۹ مراقبت و نگهداری

در **BON** اشاره‌ای به فعالیت‌های بخصوصی برای انواع مراقبت و نگهداری نشده است.

USDP فرایندی برای نگهداری تعریف نکرده است و برای آن تکرار جریان‌های کاری موجود را پیشنهاد داده است که این کار برای نگهداری اصلا مناسب نیست. ماهیت نگهداری مبتنی بر وقفه[۱۰] است و غیر قابل پیش‌بینی. بنابراین استفاده از فرایند تکراری این متدولوژی برای این کار مناسب نیست و بهتر است فرایند دیگری برای نگهداری تعریف شود. البته در انتهای فاز انتقال، باید تیم پشتیبانی شکل گرفته و آموزش‌های لازم داده شده باشد.

در نتیجه بررسی‌های انجام شده، به طور کلی USDP پشتیبانی بهتری نسبت به BON از چرخه‌حیات عمومی ایجاد نرم‌افزار دارد.

۴-۱-۳ پشتیبانی از فعالیت‌های چتری

۴-۱-۳-۱ مدیریت ریسک

در متدولوژی **BON** هیچ یک از تکنیک‌های اشاره شده برای مدیریت ریسک از جمله تحلیل مقدماتی، ساخت نمونه‌اولیه، برنامه‌ریزی براساس ریسک، فرایند تکراری–افزایشی، دخالت فعالانه کاربر، ترخیص زودهنگام، صحت‌سنجی و اعتبارسنجی پیوسته، بازبینی تکراری و یکپارچه‌سازی مداوم دیده نمی‌شود. اما بررسی نظر کاربر نهایی در معیار پذیرش برخی تکالیف وجود دارد که می‌توان تا حدی در جهت مدیریت ریسک در نظر گرفت.

---

[۱۰] interrupt-driven

در **USDP** و در فاز آغاز، ریسک‌های بحرانی شناسایی و بررسی شده و تحلیل امکان‌پذیری صورت می‌گیرد. از تکنیک ساخت نمونه‌اولیه نیز بهره می‌برد. در فاز آغاز، سند ارزیابی ریسک تولید می‌شود. در فاز تفصیل، سند ساخته شده بازبینی و اصلاح شده و نیازمندی‌های دارای ریسک طراحی، پیاده‌سازی و تست می‌شوند. در انتهای فاز تفصیل باید تمام موارد مربوط به ریسک پرداخته شده باشند. فرایند USDP به صورت تکراری–افزایشی است و همچنین برای محصولات شامل توجیه اقتصادی و برنامه‌ریزی و غیره از کاربر بازخورد دریافت می‌شود و مشارکت فعال کاربر در نظر گرفته شده است. در جریان‌کاری تست در هر تکرار، یکپارچه‌سازی نیز انجام می‌شود.

۴-۱-۳-۲ مدیریت پروژه

در **BON** به صراحت به موارد مربوط به مدیریت پروژه پرداخته نشده است.

در **USDP** برخلاف RUP جریان‌کاری مجزای مدیریت پروژه وجود ندارد و این فعالیت طی فازها انجام می‌شود. در فاز آغاز فعالیت‌های مدیریت پروژه شامل برنامه‌ریزی و ایجاد نمودار گنت[۱۱] و تخمین اقتصادی است و در صورت تصمیم برای ادامه پروژه یک برنامه پروژه ساخته می‌شود. در انتهای فاز تفصیل، برنامه دقیق برای تکرارهای ابتدایی فاز ساخت ایجاد می‌شود و برنامه‌ریزی کلی به صورت تخمینی و با دقت کم‌تر انجام می‌گیرد. برنامه‌ریزی به صورت مکرر طی فازها مورد بازبینی و اصلاح قرار می‌گیرند. در واقع در انتهای هر فاز برای تکرارهای ابتدایی فاز بعدی برنامه ریزدانه و دقیق‌تر ارائه شده و برای تکرارهای بلندمدت برنامه درشت دانه و تخمینی ارائه می‌شود.

۴-۱-۳-۳ تضمین کیفیت

متدولوژی **BON** بسیار تحت تاثیر مفهوم طراحی با قرارداد[۱۲] است که یکی از تکنیک‌های مفید تضمین کیفیت است. به خصوص در بالابردن قابلیت اطمینان[۱۳] تاثیر خوبی دارد. در تکلیف تعریف کردن ویژگی‌های عمومی، توضیحات غیرصوری کلاس‌ها به واسطهای صوری کلاس با قرارداد همراه پس‌شرط و پیش‌شرط‌ها[۱۴] و نامتغیرهای کلاس[۱۵] ترجمه می‌شوند. اما مکانیزم مشخصی برای رهگیری مدل‌ها و پیاده‌سازی‌ها به نیازمندی ارائه نشده است. همچنین در فاز طراحی، مدل‌های ساخته شده پیشین مورد بازبینی و اصلاح قرار می‌گیرند و در انتهای بعضی تکالیف تایید کاربر نهایی نیز دریافت می‌شود.

---

[۱۱] gantt chart
[۱۲] design by contract
[۱۳] reliability
[۱۴] pre- post-conditions
[۱۵] class invariant

در **USDP** فرایند به صورت تکراری–افزایشی اجرا می‌شود و در هر تکرار جریان‌کاری مربوط به تست وجود دارد که موجب صحت‌سنجی و اعتبارسنجی محصولات می‌شود. این متدولوژی Use-Case Driven است و همه فعالیت‌ها مبتنی بر آن انجام می‌شوند و این باعث می‌شود رهگیری به نیازمندی‌ها به صورت مستقیم وجود داشته باشد. در فاز تفصیل نیز معیارهای کیفیت مانند نرخ کشف اشکال و تراکم اشکال قابل‌قبول تعریف می‌شوند که پایش آن‌ها می‌تواند در تضمین کیفیت مفید باشد.

در نتیجه، متدولوژی USDP در فعالیت‌های چتری خصوصا مدیریت ریسک و مدیریت پروژه بهتر عمل می‌کند و در کل پشتیبانی بهتری از فعالیت‌های چتری نسبت به BON دارد.

۴-۱-۴ بی‌درزی و همواری انتقال

یکی از اهداف متدولوژی **BON** بی‌درز بودن است. در تمام فعالیت‌ها از زبان بخصوص خود و مفاهیم شی‌گرایی برای مدل‌سازی استفاده می‌کند و با شناسایی تدریجی کلاس‌ها و ایجاد مدل‌های میانی سعی در حفظ بی‌درزی می‌کند. البته به طور طبیعی در گذار از نیازمندی به فعالیت‌های تحلیل می‌توان شاهد مقداری درز بود. همچنین این متدولوژی در هر فعالیت مدل‌های ساخته‌شده در فعالیت‌های پیشین را به مرور بروزرسانی کرده و مدل‌های جدید با استفاده از اطلاعات بدست آمده و با روند طبیعی ساخته می‌شوند بنابراین همواری در انتقال نیز دیده می‌شود.

متدولوژی **USDP** به صورت Use-Case Driven است و تمام فعالیت‌ها و محصولات حول آن پیشروی می‌کنند. همچنین از مفاهیم شی‌گرایی تبعیت می‌کند بنابراین همه فعالیت‌ها و محصولات بر یک مفهوم مشترک تکیه دارند که این یکی از تکنیک‌های کنترل درز است. البته به دلیل ماهیت غیر شی‌گرای Use-Case و ماهیت شی‌گرای سایر مدل‌هایی که ساخته می‌شود مانند نمودار توالی[۱۶] در UML، می‌توان شاهد درز بود که با کمک نمودار فعالیت[۱۷] و مفاهیم آن مانند خط‌شنا[۱۸] این درز تا حدی هموار می‌شود. همچنین فرایند این متدولوژی به صورت تکراری–افزایشی بوده و کارها با پیمانه نسبتا کوچک انجام شده و همچنین به دلیل استفاده از UML برای مدل‌سازی تمام محصولات که طی تکرارها و فازها تکامل پیدا می‌کنند، میتوان گفت انتقال هموار دیده می‌شود.

بنابراین هردو متدولوژی تا حد خوبی بی‌درز بوده و انتقال هموار دارند.

---

[۱۶] sequence diagram
[۱۷] activity diagram
[۱۸] swimlane

۴-۱-۵ مبتنی بودن بر نیازمندی‌ها (وظیفه‌ای و غیر وظیفه‌ای)

در متدولوژی **BON** تنها مدل‌های ساخته‌شده در فعالیت‌های ابتدایی و تحلیل براساس نیازمندی‌های غیر رسمی کاربر هستند و مابقی فعالیت‌ها و محصولات به صورت مستقیم مبتنی بر نیازمندی‌ها نیستند و از روی مدل‌های پیشین ساخته و تکامل پیدا می‌کنند. بنابراین نمی‌توان گفت که این متدولوژی Requirements-Driven است.

متدولوژی **USDP** به صورت Use-Case Driven بوده و مبنا در تمام فعالیت‌ها نیازمندی است. بنابراین رهگیری مستقیم به نیازمندی‌ها وجود دارد. همچنین نیازمندی‌ها طی فازهای مختلف استخراج می‌شوند. در فاز آغاز ۲۰% نیازمندی‌ها، در فاز تفصیل تا ۸۰% و مابقی در فاز ساخت استخراج می‌شوند و سیستم بر اساس آن‌ها ساخته می‌شود. بنابراین USDP عملکرد بهتری در این معیار نسبت به BON دارد.

۴-۱-۶ آزمون‌پذیری، ملموس بودن و قابلیت رهگیری به نیازمندی‌ها

۴-۱-۶-۱ آزمون‌پذیری

در متدولوژی **BON** محصولات پیچیدگی زیادی ندارند ولی تعداد نسبتا زیادی را شامل می‌شوند که بین آن‌ها وابستگی وجود داشته و تغییر در هرکدام می‌تواند باعث تغییر در دیگری شود. مصنوعات ایجاد شده مکمل یکدیگر بوده و در طی فعالیت‌ها به مرور تکمیل می‌شوند.

در **USDP** که مدل‌سازی بسیار سنگین است و مدل‌های متعدد طی فازهای مختلف بوجود می‌آیند و همچنین پیچیدگی موجود در آن‌ها و گاها مدل‌هایی که وجودشان باعث کامل‌شدن بقیه مدل‌ها نشده و صرفا تزیین میکنند، آزمون‌پذیری را کاهش می‌دهد.

۴-۱-۶-۲ ملموس بودن

در متدولوژی **BON** محصولات نسبتا ساده هستند و مدل‌های ساخته شده از دید کاربر نیز می‌توانند درک شوند و حتی تایید کاربر نهایی نیز گرفته شود. همچنین مصنوعات در جهت فعالیت‌های فرایند و تکمیل یک بخش از آن ایجاد می‌شوند که باعث می‌شود مصنوعات از دید ایجادکننده نرم‌افزار نیز ملموس باشند.

در **USDP** طی فازها محصولاتی ساخته می‌شوند مانند سند چشم‌انداز، سند توجیه اقتصادی و یا سند ارزیابی ریسک که ساده بوده و از جانب کاربر قابل درک هستند و نیاز به تایید آن‌ها نیز دارند. همچنین برخی مدل‌های UML مانند نمودار کلاس در تحلیل برای کاربر ملموس هستند ولی برخی دیگر مانند نمودار کلاس در طراحی که جزئیات پیاده‌سازی دارد، برای کاربر ملموس نیستند. مورد کاربر هرکدام از مدل‌ها و مصنوعات

در فازهای مختلف برای ایجادکنندگان نیز قابل درک هستند البته به شرط این که مدل‌ها مکمل یکدیگر بوده و صرفا برای تزئین یا از نوع هم‌ریخت نباشند و در راستای انجام فعالیت‌های فرایند باشند و همچنین این متدولوژی سنگین در پروژه مناسب استفاده شود، ملموس و قابل درک هستند.

۴-۱-۶-۳ قابلیت رهگیری به نیازمندی‌ها

در متدولوژی **BON** در تکلیف مشخص کردن مرز سیستم، نیازمندی‌ها استخراج شده و سپس براساس آن‌ها مدل ایجاد می‌شود و در تکالیف بعدی، مدل‌های جدید از روی مدل‌های قبل ایجاد می‌شوند و نه به صورت مستقیم از روی نیازمندی. همچنین یک نمودار سناریو ساخته می‌شود که رفتار سیستم را بر اساس محرک‌های داخلی و خارجی نمایش می‌دهد. بنابراین این متدولوژی قابلیت رهگیری به نیازمندی‌ها را بطور غیر مستقیم دارد.

متدولوژی **USDP** به صورت Use-Case Driven بوده و تمام کارها بر مبنای آن شکل می‌گیرند بنابراین رهگیری به نیازمندی‌ها به صورت مستقیم وجود دارد.

۴-۱-۷ تشویق مشارکت فعال کاربر

در متدولوژی **BON** تشویق برای مشارکت فعال کاربر در فرایند ایجاد مشاهده نمی‌شود و مطلبی درباره حضور نماینده مشتری در تیم نیز ارائه نشده است همچنین جلسات برنامه‌ریزی و مرور با حضور کاربر نیز وجود ندارد. ولی در بعضی تکالیف، تایید محصولات توسط کاربر نهایی در معیار پذیرش قرار دارد.

در **USDP** محصولات ساخته شده در هر فاز برای دریافت تایید به ذی‌نفعان ارائه می‌شوند. در فاز آغاز، هدف پروژه، نیازمندی‌های اساسی، حوزه سیستم، تخمین هزینه و زمان و در فاز تفصیل، توجیه اقتصادی و برنامه پروژه و در فاز ساخت، محصول قابل ارائه به کاربر نشان داده شده و تایید گرفته می‌شود. در فاز انتقال نیز پس از نصب و انجام تمهیدات لازم، تایید نهایی گرفته می‌شود و اینگونه مشارکت کاربر طی فازها دیده می‌شود.

در این معیار USDP نسبت به BON بسیار بهتر عمل می‌کند.

۴-۱-۸ قابلیت اجرا و قابلیت اجرا به صورت کارا

۴-۱-۸-۱ قابلیت اجرا

متدولوژی **BON** مصنوعات نسبتا زیادی دارد ولی این مصنوعات به مرور طی تکالیفی که مشخص و تاحدی ساده هستند، اصلاح و تکامل می‌یابند. البته برای پروژه‌های بزرگ این مدل‌ها می‌توانند بسیار بزرگ شوند. همچنین این متدولوژی تا حدی از روش‌های صوری پشتیبانی می‌کند که می‌تواند برای برخی از این دست پروژه‌ها نیز مفید باشد.

متدولوژی **USDP** سنگین وزن بوده و فرایند پیچیده‌ای دارد که طی آن‌ها مدل‌ها و مستندات مفصلی ایجاد می‌شوند و این موضوع اجرای USDP به همان شکل که هست را بسیار سخت می‌کند و باعث کاهش عملکرد این متدولوژی در معیار قابل اجرا بودن می‌شود و برای اجرای آن نیاز به شخصی‌سازی متناسب با پروژه دارد که کار آسانی نیست.

۴-۱-۸-۲ قابلیت اجرا به صورت کارا

فرایند **BON** تا حدی سادگی در فعالیت‌های خود دارد و همچنین تکالیف در جهت جریان فعالیت‌های اصلی بوده و تمرکز ایجادکنندگان دچار انحراف نمی‌شود. البته جلسات مدیریت تیم در آن لحاظ نشده است ولی مدل‌های اولیه از روی نیازمندی‌ها و سایر مدل‌ها از روی مدل‌های پیشین ساخته می‌شوند. از تکنیک‌های نسبتا ساده استفاده شده است. هرچند وابستگی به ابزار یا فناوری خاص ندارد، اما برای مدل‌سازی با زبان خاص معرفی شده در متدولوژی و حفظ سازگاری بین مدل‌ها، پیشنهاد شده از ابزار CASE معرفی شده یعنی EiffelCase استفاده کرد. همچنین ضعف در قسمت استراتژی‌های مدیریت پروژه نیز حس می‌شود.

USDP به صورت Use-Case Driven بوده و مدل‌ها نیز مبتنی بر نیازمندی ساخته می‌شوند و همچنین مدل‌های معماری سیستم و معماری مبنای قابل‌اجرا همگی برای حفظ تمرکز ایجادکنندگان نرم‌افزار مفید هستند. همان‌طور که اشاره شد فرایند USDP پیچیده بوده و نیاز به شخصی‌سازی برای پروژه هدف دارد. این متدولوژی وابسته به ابزار یا فناوری خاصی نیست اما در مدل‌سازی بر UML تعصب و اجبار دارد و بنابراین ضعف‌های این زبان مدل‌سازی در متدولوژی حس می‌شوند. متدولوژی استراتژی‌های مدیریت پروژه نسبتا خوبی نیز دارد.

متدولوژی BON در این معیار تاحدودی وضعیت بهتری دارد و USDP برای قابلیت اجرا نیاز به شخصی‌سازی برای پروژه هدف دارد اما برای پروژه با سایز بالا و سطوح بحرانیت بالا مناسب‌تر است.

۴-۱-۹ قابلیت مدیریت پیچیدگی

همان‌طور که اشاره شد، متدولوژی **BON** از ۳ فاز جمع‌آوری، توصیف و طراحی تشکیل شده و ۹ تکلیف موجود در این ۳ فاز تقسیم شده‌اند و سطوح مختلف انتزاع دیده می‌شود. مصنوعات متعدد در طی تکالیف فرایند ساخته می‌شوند که قبلا معرفی شدند. این مصنوعات از تکنیک‌های لایه‌بندی[۱۹] و جزء بندی[۲۰] استفاده می‌کنند برای مثال تقسیم‌بندی به کلاس‌ها و گروه‌بندی به خوشه‌ها در مدل‌سازی‌ها دیده می‌شوند. از ایندکس نیز برای مدیریت مولفه‌ها استفاده می‌شود. بنابراین سطوح مختلف انتزاع در چرخه‌حیات و تقسیم آن به بخش‌های مجزا نشان از مدیریت پیچیدگی با تکنیک‌های گفته شده است.

متدولوژی **USDP** نیز از ۴ فاز آغاز، تفصیل، ساخت و انتقال تشکیل شده و در هرکدام از فازها تعدادی تکرار وجود دارد و هرکدام از تکرارها نیز متشکل از جریان‌های کاری نیازمندی‌ها، تحلیل، طراحی، پیاده‌سازی و تست هستند بنابراین سطوح انتزاع مختلف در فازها دیده می‌شود که بیانگر استفاده از تکنیک لایه‌بندی است. همچنین در هر تکرار ۵ جریان‌کاری انجام می‌شود و فعالیت‌ها و چرخه‌حیات به قسمت‌های مختلف تقسیم شده‌اند که نشان‌دهنده جزء بندی است. همچنین این متدولوژی به صورت تکراری–افزایشی نیز اجرا می‌شود.

۴-۱-۱۰ قابلیت‌های گسترش، مقایس‌پذیری، پیکربندی و انعطاف

۴-۱-۱۰-۱ قابلیت گسترش

متدولوژی **BON** به صورت هسته قابل گسترش نیست و اشاره‌ای به نقاط گسترش و مکانیزم‌های لازم نکرده است.

متدولوژی **USDP** فرایند بزرگ و پیچیده‌ای دارد و اساسا برای استفاده کارا از آن، باید به روش‌های مختلف مانند هرس یا تجمیع قطعات لازم، آن را متناسب با پروژه هدف ساخت و این متدولوژی به صورت هسته قابل گسترش نبوده، نقاط گسترش و مکانیزم‌هایی برای این کار تعریف نکرده است.

۴-۱-۱۰-۲ مقایس‌پذیری

متدولوژی **BON** مکانیزم‌های بخصوصی برای انواع پروژه‌ها با سطوح بحرانیت مختلف خصوصا سطوح بالاتر ارائه نکرده است و همچنین مدیریت ریسک قوی ندارد و به دلیل نداشتن مدیریت تیم و مدیریت پروژه قوی، برای پروژه با هر اندازه‌ای مناسب نیست. البته در این متدولوژی مدل‌سازی وجود دارد و مدل‌ها می‌توانند به

[۱۹] layering
[۲۰] partitioning

عنوان رسانه‌های انتقال و به اشتراک گذاری دانش نیز استفاده شوند. در مدل‌سازی نیز مکانیزم‌هایی مانند گروه‌بندی کلاس‌ها در خوشه‌ها وجود دارند که در کل تا حدی به مقیاس‌پذیری کمک می‌کنند.

متدولوژی **USDP** مدیریت ریسک و پروژه خوبی دارد، همچنین مدل‌سازی نیز در این متدولوژی جدی است و از زبان غنی UML استفاده می‌شود. درباره سطوح بحرانیت، UML تا حدی از فرمالیزم با استفاده از OCL پشتیبانی می‌کند البته درحد زبان‌های فرمال نیست. مدل‌های ایجاد شده نیز در انتقال و اشتراک گذاری دانش کمک می‌کنند. بنابراین درکل مقیاس‌پذیری بهتری نسبت به BON دارد.

۴-۱-۱۰-۳ قابلیت پیکربندی

متدولوژی **BON** فرایند ثابت دارد و مکانیزم‌هایی برای پیکربندی ارائه نشده است. هرچند بیان شده است که ترتیب تکالیف می‌تواند متغیر باشد، ولی اشاره شده است که تمام آن‌ها باید اجرا شده و مصنوعات معرفی شده تولید شوند.

متدولوژی **USDP** به دلیل پیچیدگی بالا، بهتر است قبل از شروع پروژه براساس موقعیت پروژه‌ای پیکربندی شود که این کار عمدتا با هَرَس کردن یا تجمیع قطعات با پشتیبانی ابزاری انجام می‌شود. البته خود پیکربندی این پروژه کار آسانی نیست هرچند نسبت به RUP بسیار آسان‌تر شده است. ولی درکل قابلیت پیکربندی اولیه را دارد.

۴-۱-۱۰-۴ انعطاف‌پذیری

در متدولوژی **BON** صحبتی از جلسات بازبینی فرایند تکراری یا بازنگری‌های مبتنی بر بازخورد نشده است و در حین اجرای فرایند قابلیت تغییر ندارد و فقط می‌توان ترتیب برخی تکالیف را با رعایت اصول گفته شده اندکی تغییر داد.

در **USDP** هرچند به اندازه متدولوژی‌های چابک نیست و جریان‌کاری مخصوصی ارائه نکرده است، اما در انتهای هر فاز، جلسات بازبینی‌های نقطه‌عطف[۲۱]، ارزیابی‌های انتهای تکرارها، حلقه‌های بازخورد و جلسات مدیریت پیکربندی، علاوه بر بازخوردها و بررسی‌های فنی و موارد کسب‌وکار، تاحدی فرایند نیز در نظر گرفته می‌شود.

[۲۱] milestone

۴-۱-۱۱ حوزه کاربرد

در مقاله **BON** به عنوان مثال، سیستم نرم‌افزاری برای خودکار سازی رزرو و صورت‌حساب برای یک شرکت اجاره خودرو، بیان شده است. البته در توصیف این متدولوژی اشاره‌ای به حوزه خاصی از سیستم‌ها نشده اما به دلیل ماهیت شی‌گرای آن و مدل‌سازی‌های موجود، برای انجام پروژه‌های اطلاعاتی کفایت می‌کند. البته مشکلِ نداشتن پشتیبانی از نگهداری و مراقبت را دارد.

متدولوژی **USDP** فرایند سنگین دارد که در آن مدل‌سازی، مستندسازی و فعالیت‌های مدیریتی اعم از مدیریت سازمانی سنگین است. این متدولوژی برای پروژه‌های با پیچیدگی و اندازه بالا که نیاز به روش ساختاریافته باشد، مناسب است و می‌توان انواع سیستم‌های اطلاعاتی را برای آن، مورد هدف قرار داد. اگرچه با استفاده از OCL در UML پشتیبانی تاحدی از فرمالیزم وجود دارد ولی برای پروژه‌های با بالاترین سطوح بحرانیت مناسب نیست و بهتر است از روش‌های فرمال استفاده کرد. البته این متدولوژی نیز در مورد نگهداری و مراقبت ضعف دارد.

۴-۲ زبان مدل‌سازی

در این بخش، متدولوژی‌ها از منظر زبان مدل‌سازی مورد ارزیابی قرار می‌گیرند.

۴-۲-۱ پشتیبانی از مدل‌سازی شی‌گرای سازگار، صحیح و دقیق و بدون ابهام

متدولوژی **BON** نمادگان[۲۲] بخصوصی برای مدل‌سازی شی‌گرای سیستم‌ها ارائه کرده است که با استفاده از آن طی فعالیت‌های متدولوژی، مصنوعاتی قابل تحویل که پوششی بر دیدگاه‌های ساختاری[۲۳]، رفتاری[۲۴] و وظیفه‌ای[۲۵] دارند، ارائه می‌کند. مصنوعاتی مانند نمودار کلاس، نمودار خوشه، نمودار رویداد، نمودار سناریو و غیره. در این متدولوژی مدل‌سازی از قلمرو مسئله تا قلمرو جواب وجود دارد و مدل‌های ساخته شده در فاز تحلیل، در فاز طراحی به مرور تکمیل می‌شوند و جزئیات قلمرو جواب به آن‌ها اضافه می‌شود. همچنین به دلیل ارتباط این متدولوژی با زبان Eiffel و مکانیزم‌های آن، امکان مدل‌سازی در سطح کد با استفاده از آن آسان‌تر می‌شود. پشتیبانی از مدل‌سازی در سطوح مختلف درشت‌دانگی از سطح سیستم به سمت زیرسیستم‌ها و اشیا و تعاملات بین اشیا وجود دارد و مصنوعاتی مانند معماری ایستا، نمودار خوشه، نمودار کلاس و سناریوهای

[۲۲] notation
[۲۳] structural
[۲۴] behavioral
[۲۵] functional

شی ساخته می‌شوند. این زبان پشتیبانی خوبی نیز از فرمالیزم دارد و همچنین از مکانیزم طراحی با قرارداد[۲۶] استفاده می‌کند که توضیحات بیش‌تر در فصل مربوط به BON ارائه شده است.

در متدولوژی **USDP** مدل‌سازی به صورت سنگین وجود دارد و در آن از UML که زبان غنی برای مدل‌سازی است، استفاده می‌شود. این زبان پشتیبانی خوبی از دیدگاه‌های مختلف مدل‌سازی اعم از ساختاری، رفتاری و وظیفه‌ای ارائه می‌کند و در طی فعالیت‌های فازها، مصنوعات متعددی ساخته می‌شوند شامل نمودار کلاس، نمودار مولفه، نمودار توالی، use-case و غیره. در این متدولوژی مدل‌سازی از قلمرو مسئله آغاز می‌شود و انواع مدل تحلیل بدون در نظر گرفتن جزئیات پیاده‌سازی ایجاد و به تدریج به سمت قلمرو جواب رفته و مدل‌هایی حاوی جزئیات پیاده‌سازی ساخته می‌شوند. همچنین سطوح مختلف انتزاع و درشت‌دانگی در آن پشتیبانی می‌شود و مصنوعاتی برای مدل کردن سیستم، زیرسیستم‌ها و مولفه‌ها، کلاس‌ها و تعاملات بین اشیا و غیره ساخته می‌شوند. در UML پشتیبانی از فرمالیزم از طریق OCL[۲۷] تا حدی وجود دارد، البته به اندازه روش‌های بخصوص فرمال نیست. این متدولوژی به دلیل تعصب بر UML، از ضعف‌های آن برای مثال در مدل‌سازی کسب و کار رنج می‌برد که می‌شد از روش‌هایی مانند نمودار جریان داده[۲۸] استفاده کرد.

۴-۲-۲ ارائه راهبردها و تکنیک‌هایی برای رفع ناسازگاری‌ها و مدیریت پیچیدگی

در متدولوژی **BON** مصنوعات قابل تحویل زیادی طی فعالیت‌ها ایجاد می‌شوند و وابستگی‌هایی بین آن‌ها وجود دارند که در شکل ۱-۲ ترسیم شدند. برای حفظ سازگاری بین مدل‌های ایجاد شده، معانی رسمی[۲۹] ارائه شده است ولی رعایت سازگاری بدون استفاده از ابزار مشکل است. برای این منظور ابزار CASE مانند EiffelCase برای پشتیبانی از زبان ارائه شده است. البته پشتیبانی ابزاری این زبان جای کار دارد و با افزایش تعداد مدل‌ها و وابستگی بین مدل‌های رفتاری و ساختاری، حفظ سازگاری مشکل می‌شود. همچنین در زبان ارائه شده در این متدولوژی، سازوکارهایی برای مدیریت پیچیدگی مدل ارائه شده‌اند. مانند نمودار خوشه و نمودار کلاس که با استفاده از آن‌ها طی فعالیت گروه‌بندی کلاس‌ها در خوشه‌ها، می‌توان زیرسیستم‌ها را نمایش داد و همچنین معماری ایستا که نمای کلی سیستم را نشان می‌دهد که قابل بزرگ‌نمایی است.

در متدولوژی **USDP** مدل‌سازی بسیار جدی است و تعداد بسیار زیادی مدل در سطوح مختلف ایجاد می‌شوند. حفظ سازگاری بین مدل‌ها بسیار سخت و البته مهم است. برای حفظ سازگاری در سطح مدل‌های UML، استفاده از ابزار می‌تواند بسیار مفید باشد و ابزارهایی مانند Enterprise Architect، امکاناتی برای بررسی سازگاری بین برخی مدل‌ها ارائه می‌کنند که این کار را با بررسی برخی معانی تعریف شده در UML انجام

[۲۶] design by contract
[۲۷] object constraint language
[۲۸] data flow diagram
[۲۹] semantic

می‌دهند. مانند بررسی ارتباطات میان کلاس‌ها، مولفه‌ها و بسته‌ها. همچنین امکاناتی برای بررسی سازگاری برخی مدل‌های رفتاری مانند نمودارهای توالی با مدل‌های ساختاری مانند نمودار کلاس ارائه می‌کند. البته درصورت استفاده نکردن از ابزار، حفظ سازگاری بین تعداد زیاد مدل که در این متدولوژی ایجاد می‌شوند بسیار سخت است، خصوصا که معانی UML در طول زمان برای گسترش حوزه کاربرد آن سبک‌تر شده ولی جا برای ناسازگاری افزایش یافته است. همچنین در UML امکاناتی برای لایه‌بندی و جزءبندی ارائه شده است از طریق ایجاد نمودارهای بسته[۳۰] در مدل‌های تحلیل و نمودارهای مولفه[۳۱] در مدل‌های طراحی که با کمک آن‌ها سیستم به لایه‌های متعدد و قسمت‌های مجزا تقسیم و در نهایت به کلاس‌ها می‌رسد و از این طریق مدیریت پیچیدگی مدل حاصل می‌شود.

[۳۰] package diagram
[۳۱] component diagram

فصل ۵

متدولوژی ASD

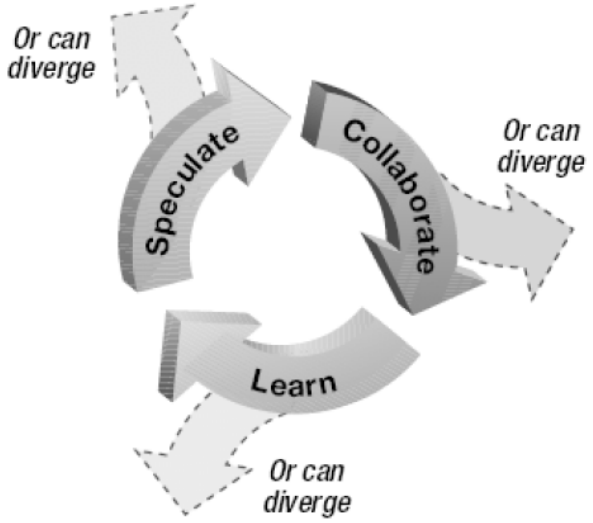
متدولوژی ASD[۱] در سال ۱۹۹۷ توسط جیمز هایسمیث[۲] معرفی شد. نسخه اصلاح شده و توسعه‌یافته آن در سال ۲۰۰۰ معرفی شد. این متدولوژی از یک فرایند RAD[۳] تکامل یافته و بر اساس آموزه‌های تئوری پیچیدگی، تلاش می‌کند یک جایگزین تطبیقی و متحمل تغییر[۴] بر چرخه‌حیات کلاسیک برنامه‌ریزی–طراحی–ساخت و برنامه‌ریزی–ساخت–اصلاح تکراری، ارائه نماید. چرخه‌حیات ایجاد مبتنی بر مولفه[۵] که توسط ASD تجویز شده است، فرض می‌کند که تمام جنبه‌ها و تشکیل‌دهنده‌های ایجاد نرم‌افزار، بسیار فرّار هستند و ساخت سیستم‌های پیچیده یک فرایند تکاملی است که دست‌یابی به آن بسیار سخت است مگر این که موارد لازم برای تسهیل همکاری بین افراد دخیل یا تحت تاثیر ایجاد سیستم، فراهم شوند.

بر اساس ASD، ماهیت نامطمئن و غیرقابل پیش‌بینی ایجاد، راهی جز استفاده از تکرارها یا چرخه‌های کوتاه برای ایجادکنندگان باقی نمی‌گذارد. برای محدود کردن تلاش‌های ایجاد و حفظ تمرکز، یک مأموریت مشخص یعنی مجموعه‌ای از مولفه‌های مورد نیاز برای ایجاد، و یک بازه زمانی[۶] برای هر چرخه تعریف می‌شوند. تکرارها باید برنامه‌ریزی شوند، اما این برنامه‌ها تنها حدس‌هایی مبتنی بر ریسک هستند که پس از هر تکرار چرخه نیاز به بازبینی دارند؛ طراحی و پیاده‌سازی واقعی مولفه‌های سیستم نتیجه جانبی همکاری شدید است. برای اینکه فرایند تطبیقی باشد، در پایان هر چرخه بازبینی‌های گروهی انجام می‌شوند تا افراد درگیر بتوانند از تجربه‌ها بیاموزند و درس‌های گرفته‌شده را در فرایند اعمال کنند. چرخه‌حیات "حدس–همکاری–یادگیری" که بدین ترتیب شکل می‌گیرد، چارچوب اصلی روش ASD برای ایجاد سیستم‌های نرم‌افزاری می‌شود (شکل ۵–۱).

متدولوژی ASD فراتر از تعریف یک چارچوب ساده عمل می‌کند؛ این روش همچنین فازهای مشخص و

---

۱ adaptive software development
۲ James Highsmith
۳ rapid application development
۴ change-tolerant
۵ component
۶ time box



شکل ۵-۱: چرخه‌حیات اصلی ASD [۲]

قابل‌اجرای چرخه‌حیات را تعیین می‌کند. پنج فازی که فرایند ASD را تشکیل می‌دهند و به طوری‌که سه فاز میانی آن موتور ایجاد تکرارشونده این متدولوژی را شکل می‌دهند، به شرح زیر هستند:

۱. آغاز پروژه[۷]: تمرکز این مرحله بر درک اهداف پروژه، تخمین اندازه و حوزه آن، بررسی محدودیت‌ها و ریسک‌های مرتبط، سازمان‌دهی تیم‌های ایجاد، شناسایی نیازمندی‌های سطح بالا و مشخص کردن معیارهای موفقیت است.

۲. فازهای تکرارشونده ایجاد[۸]

(آ) برنامه‌ریزی چرخه تطبیقی[۹]: تمرکز این مرحله بر تعیین چارچوب‌های زمانی برای پروژه و چرخه‌های ایجاد، تعریف مولفه‌هایی که باید ایجاد شوند، اختصاص مولفه‌ها به چرخه‌ها و زمان‌بندی تکرارها است. این برنامه در آغاز هر تکرار مورد بازبینی و اصلاح قرار می‌گیرد.

(ب) مهندسی همروند مولفه[۱۰]: تمرکز این مرحله بر طراحی و پیاده‌سازی همروند مولفه‌هایی است که به چرخه‌های مجزا اختصاص داده شده‌اند.

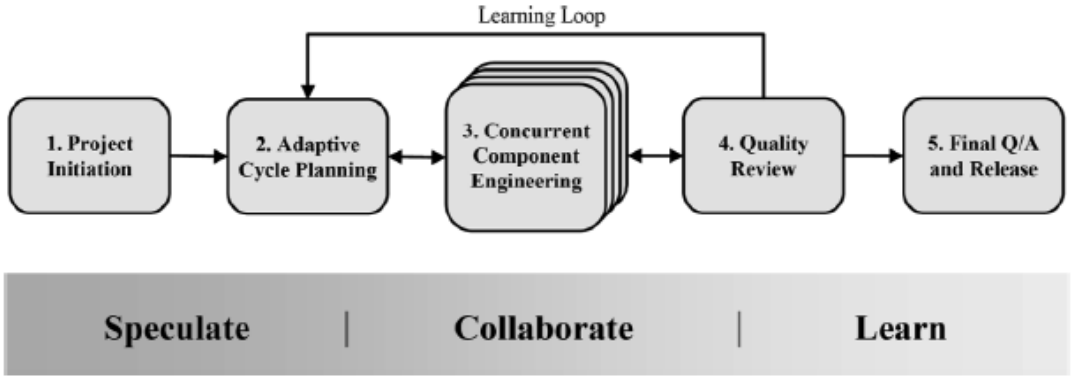
(ج) بازبینی کیفیت[۱۱]: تمرکز این مرحله بر انجام بازبینی‌های گروهی مولفه‌های تولیدشده و رفع مشکلات پیش‌آمده است.

---

[۷] project initiation
[۸] iterative development phases
[۹] adaptive cycle planning
[۱۰] concurrent component engineering
[۱۱] quality review

۳. تضمین کیفیت نهایی و انتشار[۱۲]: تمرکز این مرحله بر اعتبارسنجی سیستم تولیدشده و استقرار آن در محیط آماده به کار است.

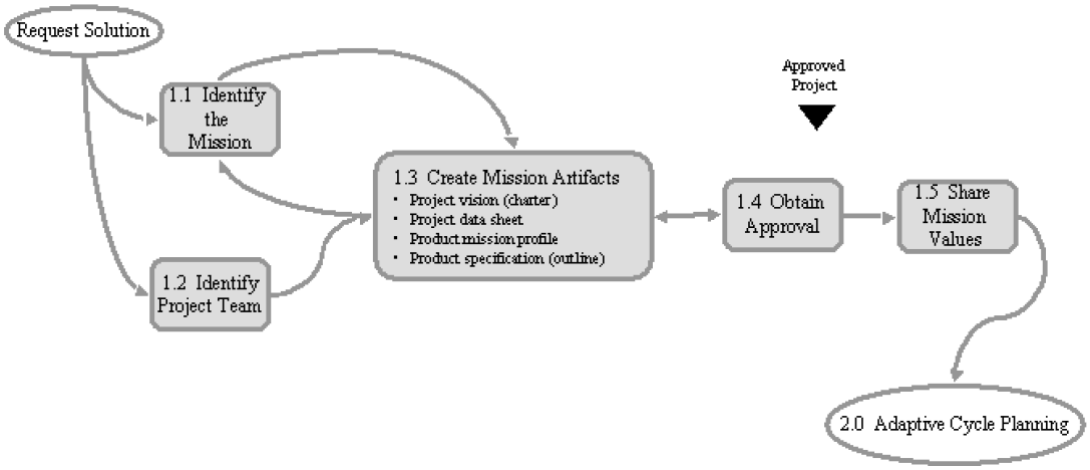
شکل ۵-۲ ترتیب فازها و تطابق نسبی آن‌ها با چرخه‌حیات پایه "حدس-همکاری-یادگیری" را نشان می‌دهد. پنج فاز ASD و فعالیت‌های هر یک به طور خلاصه در بخش‌های زیر توضیح داده شده‌اند.



شکل ۵-۲: فرایند ASD

۵-۱ آغاز پروژه

فعالیت‌هایی که در این فاز انجام می‌شوند که ترتیب آن‌ها در شکل ۵-۳ نشان داده شده است، به شرح زیر هستند:



شکل ۵-۳: ترتیب فعالیت‌ها در فاز آغاز پروژه

۱. مشخص کردن مأموریت پروژه، که اهدافی که باید به آن‌ها دست یافت و نیازمندی‌های کلی که باید توسط پروژه برآورده شوند را تعریف می‌کند.

[۱۲] final Q/A and release

۲. شناسایی تیم(های) پروژه.

۳. ایجاد مصنوعات ماموریت[۱۳]، که شامل موارد زیر می‌باشد:

- چشم‌انداز پروژه[۱۴]، که حد و مرز موارد زیر را مشخص می‌کند:
  - حوزه، اندازه و زمینه پروژه.
  - منابع تخصیص داده شده به پروژه.
  - پرسنل پروژه؛ تعریف مهارت‌ها، دانش و اختیارات مورد نیاز برای اجرای موفقیت‌آمیز پروژه.
  - ارتباط میان افرادی که در پروژه درگیر هستند یا تحت تأثیر آن قرار می‌گیرند، یعنی جامعه پروژه[۱۵].
- نمایه مأموریت محصول[۱۶]، که عوامل اصلی حاکم بر موفقیت محصول را شناسایی می‌کند. بخش اصلی این پروفایل یک ماتریس است که اولویت‌هایی را که باید به چهار متغیر پروژه شامل حوزه، کیفیت، زمان‌بندی و منابع اختصاص یابد تا پروژه را به سمت محصول موفق هدایت کند، نشان می‌دهد. این ماتریس همچنین مقادیر هدفی را که برای هر متغیر باید به دست آید و درجه مصالحه قابل‌قبول را نمایش می‌دهد.
- مشخصات محصول (طرح کلی)[۱۷]، که شامل نتایج تحلیل و مدل‌سازی سیستم‌ها است که در فازهای بعدی از نظر عمق و وسعت غنی‌تر می‌شود. در این مرحله معمولاً شامل فهرستی از نیازمندی‌ها (که اولویت‌ها، وابستگی‌ها و ریسک‌های مرتبط با ایجاد آن‌ها نیز نمایش داده می‌شود) و همچنین مدل‌هایی از سیستم است که وظیفه‌مندی کلی، کلاس‌های اصلی اشیاء و تعاملات مرتبط را نشان می‌دهد.
- برگه داده‌های پروژه[۱۸]، که یک سند یک‌صفحه‌ای است که دانش کلی که تاکنون در مورد پروژه جمع‌آوری شده را خلاصه می‌کند. معمولاً شامل اهداف پروژه، مشتریان و حامیان، تیم ایجاد، ویژگی‌های اصلی (وظیفه‌مندی سیستم)، حوزه کلی (به صورت یک نمودار زمینه)، منابع، مزایا و پیامدها، نقاط عطف، محدودیت‌ها، اولویت‌ها و ریسک‌های کلیدی مرتبط با پروژه می‌باشد. اطلاعات لازم برای تولید مصنوعات معمولاً از طریق جلسات JAD[۱۹] به دست می‌آید.

۴. کسب تأیید از مشتریان/حامیان و اجازه برای ادامه پروژه.

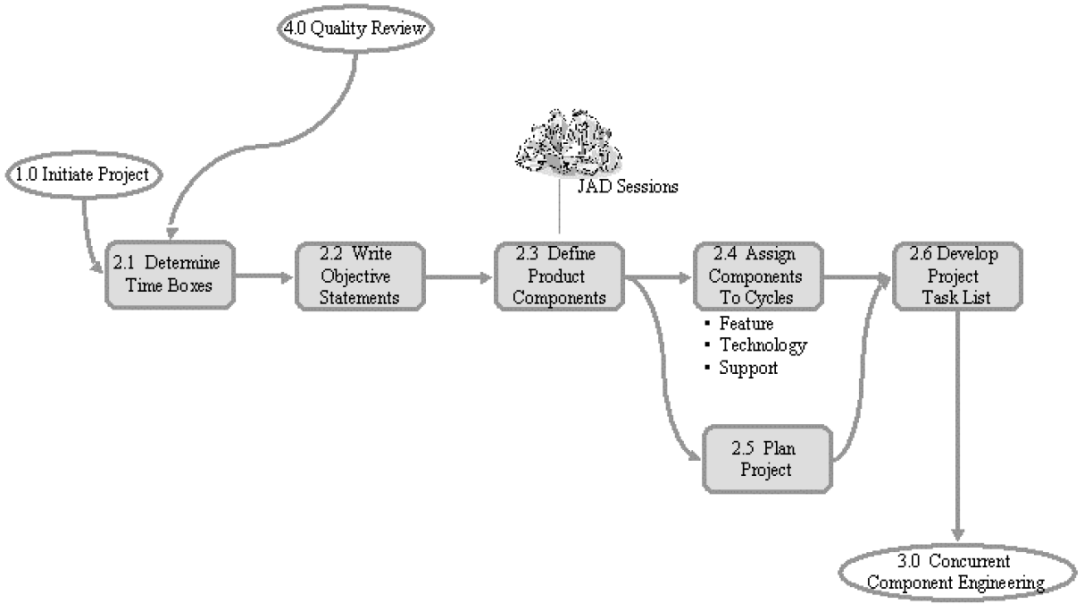
---

[۱۳] mission artifacts
[۱۴] project vision (charter)
[۱۵] project community
[۱۶] product mission profile
[۱۷] product specification (outline)
[۱۸] project data sheet
[۱۹] joint application development

۵. به اشتراک گذاشتن ارزش‌های مأموریت در میان جامعه پروژه، از طریق بحث و توافق بر سر اهداف کیفیت و معیارهای ارزیابی.

۵-۲ برنامه‌ریزی چرخه تطبیقی

فعالیت‌هایی که در اولین فاز از بخش ایجاد تکراری فرایند ASD انجام می‌شوند به شرح زیر است که ترتیب فعالیت‌ها در شکل ۵-۴ نشان داده شده است:



شکل ۵-۴: ترتیب فعالیت‌ها در فاز برنامه‌ریزی چرخه تطبیقی

۱. تعیین بازه‌های زمانی برای کل پروژه و هر یک از چرخه‌های ایجاد. قبل از مشخص کردن چارچوب‌های زمانی برای چرخه‌های ایجاد، باید تعداد چرخه‌های مورد نیاز برای ایجاد سیستم برآورد شود. بازه‌های زمانی چرخه‌ها در ASD معمولاً بین دو تا هشت هفته طول می‌کشند.

۲. نوشتن بیانیه‌های هدف برای چرخه‌های ایجاد. بیانیه هدف به تیم ایجاد کمک می‌کند تا تلاش‌های خود را در طول چرخه متمرکز کند.

۳. تعریف مولفه‌های محصول از طریق جلسات JAD. این مولفه‌ها سیستم نهایی را تشکیل می‌دهند که نیازمندی‌ها را پیاده‌سازی می‌کند و به سه نوع تقسیم می‌شوند: مولفه‌های ویژگی، که خاص دامنه هستند؛ مولفه‌های تحلیل که منطق کسب‌وکار سیستم را اجرا می‌کنند؛ و مولفه‌های فناوری و مولفه‌های پشتیبانی، که مستقل از دامنه هستند، مولفه‌های طراحی که به عنوان زیرساخت فنی عمل می‌کنند و مولفه‌های ویژگی برای اجرا و عملکرد صحیح در زمان اجرا به آن‌ها وابسته‌اند.

۴. اختصاص مولفه‌ها به چرخه‌ها بر اساس ریسک‌های مرتبط با ایجاد آن‌ها و با در نظر گرفتن وابستگی‌های متقابل آن‌ها. تخصیص باید به گونه‌ای باشد که هر چرخه یک نتیجه ملموس ارائه دهد.

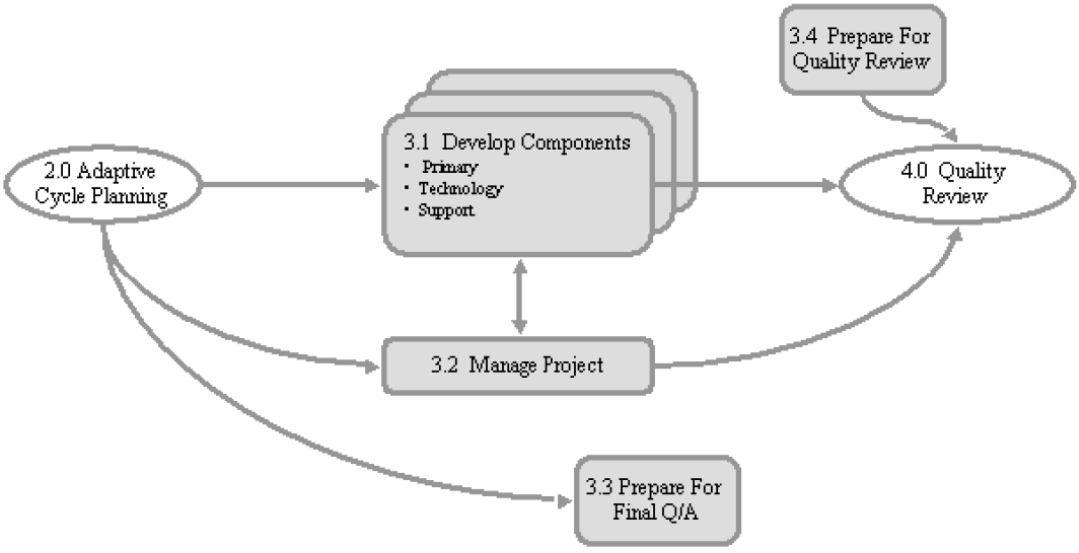
۵. برنامه‌ریزی پروژه؛ فعالیتی که معمولاً شامل ایجاد زمان‌بندی‌های بافر شده برای چرخه‌های ایجاد (با در نظر گرفتن ریسک‌های مرتبط با هر یک و منابع مورد نیاز آن‌ها) و ایجاد یک رسانه مناسب (روش‌ها، ابزارها و رویه‌ها) برای تسهیل و تقویت همکاری میان اعضای جامعه پروژه است.

۶. ایجاد فهرست تکالیف پروژه، که شامل تکالیفی است که باید در طول فازهای باقی‌مانده پروژه انجام شوند. به طور طبیعی، بیشتر تکالیف مستقیماً با ایجاد مولفه‌ها مرتبط هستند.

به دلیل ماهیت تکراری این فاز، برنامه‌های حدسی تولیدشده در اولین تکرار، در تکرارهای بعدی بازبینی و به‌روزرسانی می‌شوند تا درس‌های آموخته‌شده را بازتاب دهند.

۵-۳ مهندسی همروند مولفه

فعالیت‌هایی که در این فاز انجام می‌شوند، که به‌درستی به‌عنوان قلب بخش ایجاد تکراری فرایند ASD در نظر گرفته می‌شود، به شرح زیر هستند که ترتیب فعالیت‌ها در شکل ۵-۵ نشان داده شده است:



شکل ۵-۵: ترتیب فعالیت‌ها در فاز مهندسی همروند مولفه

۱. ایجاد مولفه‌های اختصاص داده‌شده به چرخه. مولفه‌ها معمولاً به‌طور همزمان توسط تیم‌های ایجاد که به‌طور موازی کار می‌کنند، ایجاد می‌شوند و به‌طور روزانه یا هفتگی به‌عنوان نسخه‌های ساخته‌شده تحویل داده

می‌شوند. نسخه‌های تولیدشده بلافاصله وارد فرایند یکپارچه‌سازی می‌شوند. تست و ریفکتور فرایندهای مستمری هستند که در طول این فعالیت انجام می‌شوند.

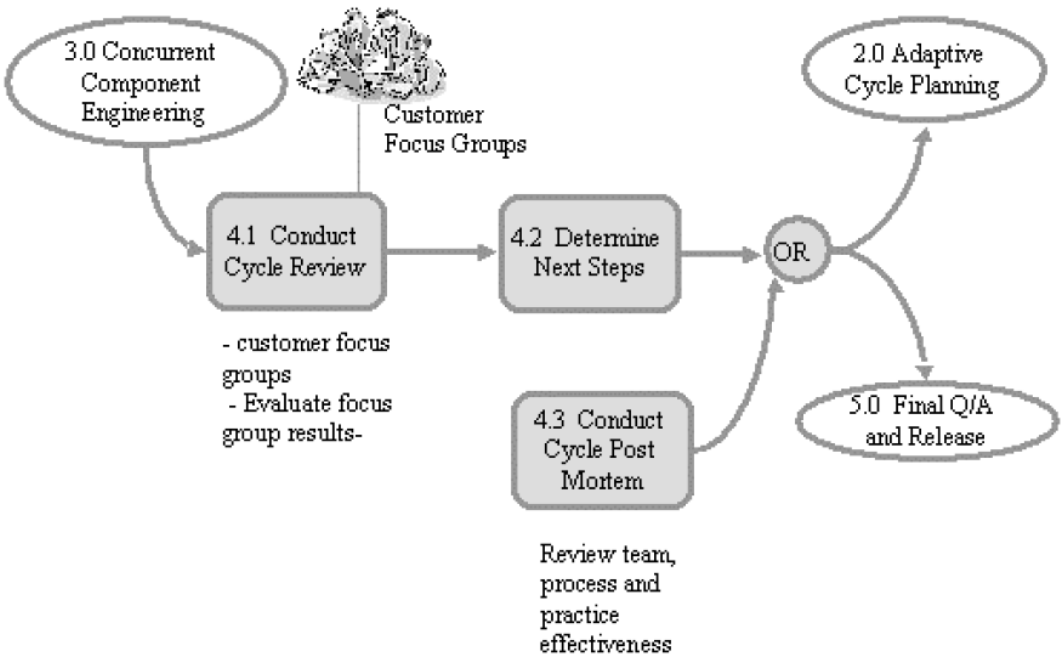
۲. مدیریت پروژه از طریق نظارت و کنترل مداوم. حفظ همکاری میان تیم‌ها و درون تیم‌ها و حفظ چرخه در مسیر صحیح از جمله نگرانی‌های اصلی هستند.

۳. آماده‌سازی برای تضمین کیفیت نهایی از طریق ایجاد برنامه‌های آزمون سطح سیستم و موارد آزمون.

۴. آماده‌سازی برای بازبینی کیفیت از طریق برنامه‌ریزی جلسات بازبینی که قرار است در فاز بازبینی کیفیت برگزار شوند.

۵-۴ بازبینی کیفیت

فعالیت‌هایی که در آخرین مرحله از مراحل تکراری انجام می‌شوند به شرح زیر است که ترتیب آن‌ها در شکل ۵-۶ نشان داده شده است:



شکل ۵-۶: ترتیب فعالیت‌ها در فاز بازبینی کیفیت

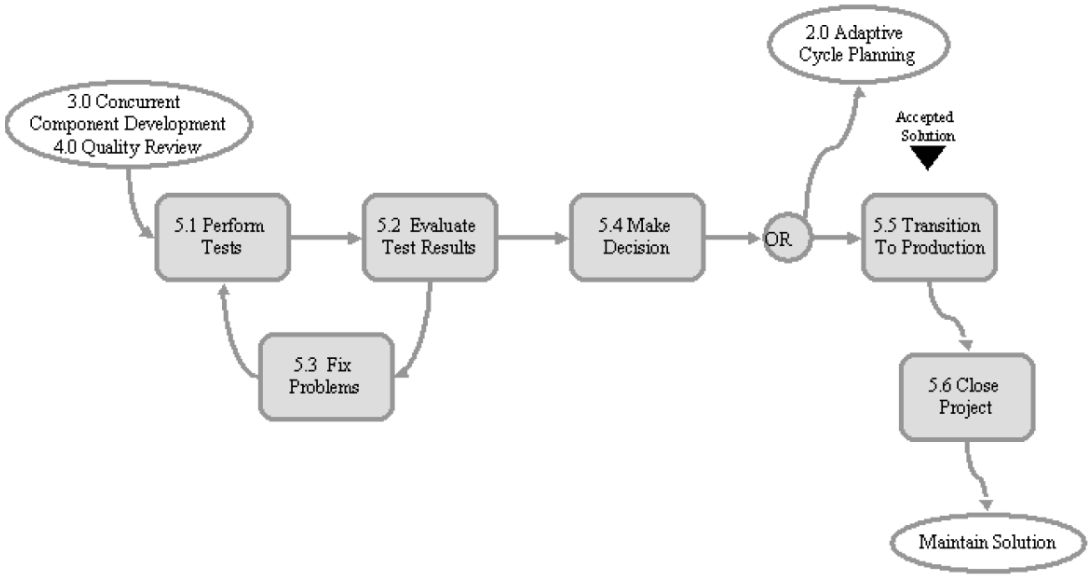
۱. برگزاری بازبینی چرخه از طریق برگزاری جلسات گروه‌های متمرکز مشتری. نتیجه چرخه به مشتریان ارائه می‌شود. بازخوردها و درخواست‌های تغییر به‌دقت مستندسازی می‌شوند تا در تکرارهای بعدی مورد بررسی قرار گیرند.

۲. تعیین گام بعدی؛ تصمیم‌گیری می‌شود که آیا چرخه تکرار دیگری باید آغاز شود یا سیستم برای انتشار آماده گردد.

۳. برگزاری جلسه بازنگری پس از مرگ[۲۰] چرخه، که معمولاً شامل بررسی عملکرد تیم‌ها و اثربخشی روش‌های استفاده شده است. سپس مشکلات برطرف می‌شوند تا تأثیر منفی بر تکرارهای بعدی نداشته باشند.

۵-۵ تضمین کیفیت نهایی و انتشار

فعالیت‌هایی که در این فاز انجام می‌شوند به شرح زیر هستند که ترتیب آن‌ها در شکل ۵-۷ نشان داده شده است:



شکل ۵-۷: ترتیب فعالیت‌ها در فاز تضمین کیفیت نهایی و انتشار

۱. انجام آزمون‌ها، با هدف اصلی اعتبارسنجی در سطح سیستم.

۲. ارزیابی نتایج آزمون‌ها.

۳. رفع کردن مشکلات.

۴. بر اساس نتایج آزمون‌ها تصمیم‌گیری کنید که آیا سیستم منتشر شود یا چرخه ایجاد جدیدی آغاز گردد.

۵. انتقال به محیط کاربر؛ که معمولاً شامل فعالیت‌های استقرار از جمله تبدیل سیستم، آموزش و آماده‌سازی مستندات است.

---

[۲۰] post-mortem

۶. بستن پروژه، که علاوه بر رویه‌های معمول جمع‌بندی و خاتمه، شامل یک جلسه بازنگری پس از مرگ پروژه نیز می‌شود که درس‌های آموخته‌شده از اجرای پروژه را خلاصه می‌کند.

فصل ۶

متدولوژی EUP

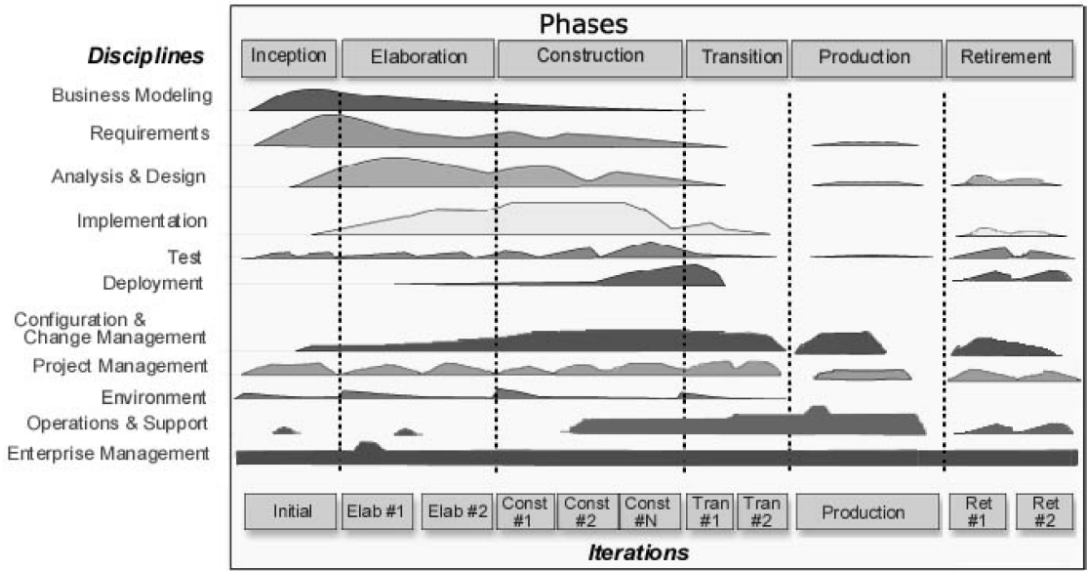
متدولوژی EUP[۱] در سال ۲۰۰۰ توسط امبلر[۲] و کنستانتین[۳] به عنوان یک نسخه گسترش یافته از متدولوژی RUP معرفی شد. ایجادکنندگان این فرایند معتقدند که RUP از مشکلات جدی‌ای رنج می‌برد که به ادعای آن‌ها در EUP اصلاح شده است. این مشکلات عبارت‌اند از:

- متدولوژی RUP، پشتیبانی از سیستم[۴] و بازنشستگی نهایی[۵] را پوشش نمی‌دهد.
- متدولوژی RUP به طور صریح از ایجاد زیرساخت‌های سطح سازمانی پشتیبانی نمی‌کند.
- طبیعت تکرارشونده RUP هم یک نقطه قوت و هم یک نقطه ضعف است، زیرا درک چرخه‌حیات تکرارشونده آن برای بسیاری از ایجادکنندگان باتجربه دشوار است.
- رویکرد شرکت رشنال برای ایجاد RUP در ابتدا ابزار–محور بود؛ بنابراین فرایند حاصل برای نیازهای ایجادکنندگان کافی نیست.

مدل چرخه‌حیات EUP در شکل ۶–۱ نشان داده شده است. این مدل با افزودن دو فاز جدید و دو discipline جدید و همچنین گسترش فعالیت‌های برخی discipline های قبلی، RUP را گسترش می‌دهد.

دیدگاه EUP نسبت به مدل‌سازی نیز تا حدی با RUP متفاوت است. در حالی که RUP بر پیروی از UML تأکید دارد، EUP از برخی نمادگان‌های قدیمی‌تر مدل‌سازی نیز استفاده می‌کند. نمونه‌ای از این موارد استفاده از نمودارهای جریان داده[۶] برای مدل‌سازی کسب‌وکار است. علاوه بر این، EUP تأکید می‌کند که

[۱] Enterprise Unified Process
[۲] Ambler
[۳] Constantine
[۴] system support
[۵] eventual retirement
[۶] data flow diagram



شکل ۶-۱: مدل چرخه‌حیات EUP

Use-Case به‌تنهایی برای مدل‌سازی نیازمندی‌ها کافی نیست؛ به همین دلیل، Use-Case نقش محوری‌ای را که در RUP دارد، در EUP ایفا نمی‌کند.

بخش‌های زیر به طور خلاصه افزوده‌ها و تغییراتی را که EUP نسبت به RUP دارد، توضیح می‌دهند.

۶-۱ فازهای جدید

دو فاز جدیدی که EUP به RUP اضافه کرده است، عبارت‌اند از:

- **Production**: به‌عنوان فاز پنجم اضافه شده است. تمرکز این فاز بر نگهداری نرم‌افزار در produc-tion است تا زمانی که یا با نسخه‌ای جدید جایگزین شود با اجرای مجدد چرخه‌حیات، یا بازنشسته و حذف شود. در این فاز هیچ تکراری وجود ندارد. این فاز تا حدی شبیه به فاز نگهداری در چرخه‌حیات کلی ایجاد نرم‌افزار است؛ زیرا بیشتر با عملیات و پشتیبانی از سیستم سروکار دارد. اما برخلاف نگهداری کلاسیک، هر نیازی به تغییر سیستم، حتی رفع یک خطا، منجر به شروع مجدد چرخه ایجاد خواهد شد.
- **Retirement**: در سال ۲۰۰۲ به‌عنوان فاز ششم اضافه شده است. تمرکز این فاز بر حذف با دقت یک سیستم از production است، یا به دلیل اینکه دیگر نیازی به آن نیست، یا در حال جایگزین شدن است. معمولاً شامل موارد زیر است:
  - شناسایی جفت‌شدگی‌های[۷] سیستم موجود با سایر سیستم‌ها.

[۷] coupling

- طراحی مجدد و اصلاح سایر سیستم‌ها به‌گونه‌ای که دیگر به سیستم درحال بازنشسته شدن وابسته نباشند.
- تبدیل داده‌های قدیمی موجود[۸].
- بایگانی داده‌هایی که قبلاً توسط سیستم نگهداری می‌شدند و دیگر نیازی به آن‌ها در سایر سیستم‌ها نیست.
- مدیریت پیکربندی نرم‌افزار حذف شده به‌گونه‌ای که در صورت نیاز بتوان آن را مجدداً نصب کرد.
- آزمون ادغام[۹] سیستم‌های باقی‌مانده تا اطمینان حاصل کنیم پس از بازنشستگی سیستم، دچار اشکال نشده‌اند.

۲-۶ نظام جدید

دو نظام جدیدی که EUP به RUP اضافه کرده است، عبارت‌اند از:

- **عملیات و پشتیبانی**[۱۰]: مربوط به مسائل مرتبط با عملیات و پشتیبانی از سیستم است که معمولاً با فاز نگهداری در چرخه‌حیات کلی ایجاد نرم‌افزار مرتبط است. با این حال، این نظام چندین فاز را شامل می‌شود و تنها به فاز production محدود نیست. در طول فاز ساخت، و شاید حتی زودتر در فاز تفصیل، ایجاد برنامه‌ها، اسناد و راهنماهای آموزشی مربوط به عملیات و پشتیبانی آغاز می‌شود. این مصنوعات در طول فاز انتقال بهبود یافته و تکمیل می‌شوند، جایی که این نظام شامل آموزش کارکنان عملیات و پشتیبانی نیز می‌شود. در طول فازهای production و بازنشستگی، این نظام فعالیت‌های کلاسیک نگهداری را پوشش می‌دهد یعنی کارکنان عملیات، نرم‌افزار را فعال نگه می‌دارند، نسخه‌برداری‌های لازم و پردازش‌های دسته‌ای را انجام می‌دهند و کارکنان پشتیبانی با کاربران ارتباط برقرار می‌کنند تا به آن‌ها در کار با نرم‌افزار کمک کنند.
- **مدیریت سازمانی**[۱۱]: مربوط به فعالیت‌هایی است که برای ایجاد، تکامل و نگهداری مصنوعات بین‌سیستمی سازمان، مانند مدل‌های سطح سازمانی (نیازمندی‌ها و معماری)، فرایند نرم‌افزار، استانداردها، راهنماها و مصنوعات قابل استفاده مجدد، مورد نیاز است.

---

[۸] legacy data
[۹] integration testing
[۱۰] operations and support
[۱۱] enterprise management

۶-۳ نظام تغییر داده شده

در EUP تغییرات متعددی در نظام RUP اعمال شده است. مانند موارد زیر:

- نظام آزمون در EUP برای شامل شدن اعتبارسنجی نیازمندی‌ها در طول فاز آغاز، گسترش یافته است. این اعتبارسنجی با استفاده از تکنیک‌هایی مانند بازبینی‌ها، بازرسی‌ها و آزمون‌های سناریو انجام می‌شود.
- نظام استقرار در EUP با فعالیت‌های مدل‌سازی استقرار که در RUP بخشی از نظام تحلیل و طراحی بود، تقویت شده است. همچنین، EUP توصیه می‌کند که برنامه‌ریزی استقرار تا حد ممکن در چرخه‌حیات زودتر آغاز شود. به دلیل این دو تغییر، نظام استقرار در EUP به فازهای آغاز و تفصیل گسترش یافته است.
- نظام محیط به‌روزرسانی شده است تا شامل کارهای لازم برای تعریف محیط production شود.
- نظام مدیریت پیکربندی و تغییر و نظام مدیریت پروژه، به فازهای جدید production و بازنشستگی گسترش یافته‌اند. علاوه بر این، ویژگی‌های جدیدی به نظام مدیریت پروژه اضافه شده است که شامل مدیریت معیارها، مدیریت پیمانکاران فرعی و مدیریت نیروی انسانی می‌شود.

۶-۴ نقاط قوت و ضعف

نقات قوت:

- دارای همان نقاط قوت RUP است.
- به مسائل سطح سازمانی می‌پردازد.
- نگهداری به عنوان یک فاز مستقل در نظر گرفته می‌شود.
- به فعالیت‌های پس از اتمام پروژه[۱۲] هنگام بازنشستگی آن توجه می‌شود به صورت یک فاز جدید به نام فاز بازنشستگی.
- به UML به طور سخت‌گیرانه پایبند نیست و از زبان‌های مدل‌سازی دیگر مانند DFD نیز استفاده می‌شود.

نقاط ضعف:

- مشابه RUP، متدولوژی EUP نیز این مشکلات را دارد:

[۱۲] post-mortem

- بسیار پیچیده است.
- دارای تعداد زیادی مدل‌های غیرضروری است.
- پتانسیل بالایی برای ناسازگاری بین مدل‌ها دارد.
- فرایند استفاده شده در آن گیج‌کننده است.
- سفارشی‌سازی آن دشوار است.

- EUP با افزودن دو فاز جدید و دو نظام جدید، پیچیدگی بیشتری به RUP اضافه کرده است.
- افزودن فاز نگهداری کافی نیست، زیرا هرگونه تغییری که موردنیاز باشد، منجر به شروع دوباره فرایند ایجاد خواهد شد.

فصل ۷

مقایسه متدولوژی‌های EUP و ASD

در این قسمت، دو متدولوژی ASD و EUP که در فصول قبل معرفی شدند، با رویکرد مبتنی بر معیار[۱] ارزیابی و مقایسه می‌شوند.

۷-۱ فرایند

ابتدا قسمت فرایندی متدولوژی‌ها، مبتنی بر معیارهایی که در فصول قبل معرفی شدند، مقایسه و ارزیابی می‌شوند.

۷-۱-۱ تعریف

۷-۱-۱-۱ چرخه‌حیات و واحدهای کاری

در توصیف متدولوژی **ASD** می‌توان به وضوح موارد پوشش چرخه‌حیات و واحدهای کاری را مشاهده کرد. چرخه‌حیات پایه این متدولوژی حدس–همکاری–یادگیری بوده و از ۵ فاز تشکیل شده است که عبارتند از آغاز پروژه، برنامه‌ریزی چرخه تطبیقی، مهندسی همروند مولفه، بازبینی کیفیت و تضمین کیفیت نهایی. ۳ فاز میانی، موتور ایجاد تکرارشونده این متدولوژی و حلقه یادگیری را تشکیل می‌دهند. فعالیت‌های ذیل هر فاز نیز توصیف شده‌اند.

متدولوژی **EUP** نیز در توصیف خود تمام موارد مربوط به چرخه‌حیات و واحدهای کاری را بیان کرده است.

[۱] criteria-based

این متدولوژی نسخه گسترش یافته از متدولوژی RUP بوده و ۲ فاز جدید و ۲ نظام جدید به آن اضافه کرده و همچنین فعالیت‌های برخی از نظام‌ها را گسترش داده است. این متدولوژی شامل فازهای آغاز، تفصیل، ساخت، انتقال، استفاده[۲] و بازنشستگی است. هرکدام از فازها شامل تعدادی تکرار می‌شود که در هر تکرار نظام‌های مدل‌سازی کسب و کار، نیازمندی‌ها، تحلیل و طراحی، پیاده‌سازی، آزمون، استقرار، مدیریت پیکربندی و تغییر، مدیریت پروژه، محیط، عملیات و پشتیبانی و مدیریت سازمان، وجود دارند.

مقایسه: هردو متدولوژی در توصیف خود به موارد چرخه‌حیات و واحدهای کاری اشاره کرده‌اند.

۷-۱-۱-۲ تولیدکنندگان (نقش‌ها)

متدولوژی **ASD** بر همکاری و تطبیق تاکید دارد و نه نقش‌های اکید. در فاز آغاز پروژه، یکی از فعالیت‌ها، شناسایی تیم(ها) و پرسنل پروژه است. در این متدولوژی، تیم به صورت یک کل و هسته در نظر گرفته می‌شود که افراد تیم وظایف مختلف داشته و با همکاری وظایفی در جنبه‌های مختلف پروژه به دوش می‌کشند و تولید مصنوعات یک وظیفه مشترک تلقی می‌شود. البته نقش‌هایی مانند تسهیل‌گر[۳] یا رهبر تیم نیز تعریف شده است ولی این نقش بجای مدیریت خُرد یا کنترل، نقش تسهیل همکاری و فرایند را ایفا می‌کند.

در متدولوژی **EUP** نقش‌ها به طور کامل و جامع به همراه توصیف وظایف و مصنوعاتی که تولید می‌کنند، بیان شده است. در EUP به نقش‌های موجود در RUP، نقش‌های جدید که عمدتا مربوط به فعالیت‌های سازمانی هستند اضافه شده است.

مقایسه: توصیف نقش‌ها در EUP بسیار دقیق‌تر و کامل‌تر است. هرچند اهداف ASD متفاوت بوده و بر همکاری تاکید دارد، EUP توصیف بهتری از نقش‌ها ارائه کرده است.

۷-۱-۱-۳ زبان مدل‌سازی

متدولوژی **ASD** استفاده از زبان مدل‌سازی خاصی را الزام نمی‌کند و به تیم‌ها این امکان را می‌دهد تا ابزارها و تکنیک هایی را انتخاب کنند که به بهترین وجه با شرایط و نیازهای آنها مطابقت دارد. این متدولوژی چابک بوده و از مدل‌سازی و مستندسازی اولیه سنگین و ثابت، دوری می‌کند و مدل‌ها بر اساس نیاز، برای حمایت از درک و ارتباط ایجاد می شوند. مانند ترسیم‌های غیر رسمی برای حل یک مسئله یا تسهیل ارتباط. این متدولوژی، مدل‌سازی را یک فعالیت پشتیبان می‌بیند و نه یک مصنوع هدف. هدف، ایجاد یک نرم‌افزار کاری است.

---

[۲] production
[۳] facilitator

در **EUP** مدل‌سازی بسیار جدی و غنی است. در این متدولوژی مانند RUP از UML که بسیار زبان غنی است، استفاده می‌شود تا مدل‌های ساختاری، رفتاری و وظیفه‌ای ایجاد شوند. همچنین در این متدولوژی برخلاف RUP، اجبار بر استفاده از صرفا UML وجود ندارد و ضعف‌های UML در مدل‌سازی کسب و کار مشخص شده و می‌توان برای حل آن موضوع از نمودار جریان داده[۴] یا BPMN[۵] استفاده کرد و این بحث در توصیف متدولوژی به خوبی شرح داده شده است.

مقایسه: زبان مدل‌سازی به صورت عمدی در توصیف متدولوژی چابک ASD در نظر گرفته نشده است ولی در EUP مدل‌سازی بسیار جدی بوده و توصیفات لازم در آن مبنی بر استفاده از UML یا زبان‌های دیگر مانند DFD برای انواع مدل‌سازی، شرح داده شده است.

۴-۱-۱-۷ محصولات

در توصیف متدولوژی **ASD**، مصنوعاتی که در هر فعالیت باید ایجاد شوند مشخص شده است. در فاز آغاز پروژه مصنوعاتی مانند چشم‌انداز پروژه، نمایه ماموریت محصول[۶]، طرح کلی و برگه داده‌های پروژه[۷] و در فاز برنامه‌ریزی چرخه تطبیقی مواردی مانند بیانیه‌های هدف، تعریف مولفه‌ها، برنامه‌ریزی پروژه و فهرست تکالیف و در فاز مهندسی همروند مولفه، ایجاد مولفه‌ها و در فاز بازبینی کیفیت مواردی مانند مستندهای بازخورد و درخواست مشتری‌ها و در فاز تضمین کیفیت نهایی و انتشار، مواردی مانند مستندات بعد از انتقال به محیط کاربر و موارد پس از مرگ[۸] که درس‌های آموخته شده را خلاصه کند، شرح داده شده و وجود دارند.

در **EUP** توصیف بسیار خوبی از محصولات و جزئیات آن‌ها که در هر فاز و طی تکرارها ایجاد می‌شوند، بیان شده است. این متدولوژی علاوه بر مصنوعاتی که در متدولوژی RUP ساخته می‌شوند، مواردی نیز اضافه کرده است که عمدتا مربوط به موارد سازمانی و کسب و کار می‌شوند. مصنوعاتی مانند مدل معماری سازمانی، مستند نیازمندی‌های معماری کسب و کار سازمانی، مدل فرایند کسب و کار سازمانی و غیره. سایر موارد مربوط به مصنوعات هر فاز، در فصل مربوط به USDP خلاصه شده‌اند.

مقایسه: تعداد مصنوعات در ASD بسیار کمتر از EUP است و این باعث می‌شود از نظر تعداد محصول، ASD بسیار سبک وزن‌تر باشد. از طرفی در EUP محصولات متنوع و زیادی تولید می‌شوند که بخشی نیز به مدیریت سازمان کمک می‌کنند و در کل در توصیف EUP با جزئیات به موارد مربوط به مصنوعات پرداخته شده است. علی رغم این موارد، محصولات در توصیف هردو متدولوژی بیان شده‌اند.

---

[۴] DFD

[۵] business process modeling and notation

[۶] product mission profile

[۷] project data sheet

[۸] post-mortem

۷-۱-۱-۵ تکنیک‌ها و قواعد

متدولوژی **ASD** بیشتر بر اصول کلی و دستورالعمل‌های کلی تاکید دارد و نه تجویز تکنیک‌ها و قواعد دقیق و سفت و سخت. بنابراین چارچوبی از چگونگی انجام کارها با تاکید بر تطبیق، همکاری و یادگیری ارائه کرده و از تعریف قواعد سخت‌گیرانه و یا بکارگیری ابزارهای خاص پرهیز کرده است. درنتیجه توصیفات آن اصول محور، سطح بالا و قابل تطبیق هستند که تیم‌ها را توانمند می‌سازد تا از طریق آزمون، همکاری و یادگیری کارها را به سرانجام برسانند.

در توصیف متدولوژی **EUP** تکنیک‌ها و رویه‌های عملی مناسب برای فعالیت‌ها اعم از مسائل سازمانی، به خوبی و با جزئیات بیان شده است. به مواردی مانند مدیریت ریسک و تغییر در طرح سازمان و قواعدی حاکم بر چرخه‌حیات، محصولات، موارد مدیریتی و غیره اشاره شده است.

مقایسه: توصیف و شرح تکنیک‌ها و قواعد در متدولوژی EUP بسیار با جزئیات و دقت بیشتری بیان شده است و اساسا ASD شامل چنین دقت و جزئیاتی در تکنیک‌ها و قواعد نمی‌شود و بسیار سبک وزن‌تر است.

۷-۱-۱-۶ موارد مربوط به فعالیت‌های چتری

در توصیف **ASD** به صراحت به موارد مربوط به فعالیت‌های چتری پرداخته نشده است ولی با این حال شامل بسیاری از اصول و شیوه‌ها در فعالیت‌ها است که می‌توان به عنوان فعالیت چتری در نظر گرفت اگرچه با این عنوان برچسب‌گذاری نشده‌اند. کیفیت در ASD در طول ایجاد تکراری و چرخه‌ها و همچنین در فاز انتهایی، نقش کلیدی دارد و طی بازبینی‌ها و اعتبارسنجی‌های مداوم از سوی مشتری، مورد پایش قرار می‌گیرد. برای مدیریت ریسک، در فاز ابتدایی، عدم قطعیت‌ها، محدودیت‌ها و ریسک‌های مرتبط مورد بررسی قرار می‌گیرند. در ایجاد تکراری، مولفه‌ها برحسب ریسک به چرخه‌ها اختصاص داده می‌شوند و همچنین تکراری–افزایشی بودن و حضور فعال مشتری همگی در جهت مدیریت ریسک بوده و در فعالیت‌های متدولوژی دیده می‌شوند. فعالیت‌هایی نیز در طی فازها با هدف مدیریت پروژه بیان شده‌اند مانند فاز برنامه‌ریزی چرخه تطبیقی که حاوی فعالیت برنامه‌ریزی پروژه می‌شود.

در توصیف متدولوژی **EUP** صراحتا موارد مربوط به فعالیت‌های چتری بیان شده‌اند و به عنوان نظام‌های پشتیبان در طول تمام فازها و تکرارها انجام می‌شوند. این فعالیت‌ها در نظام‌های مدیریت پیکربندی و تغییرات، مدیریت پروژه، محیط و مدیریت سازمان که خود مدیریت سازمان شامل ۷ نظام است، تقسیم‌بندی و انجام می‌شوند.

مقایسه: توصیف EUP صراحتا شامل دغدغه‌ها و موارد مربوط به فعالیت‌های چتری می‌شود و از این نظر

توصیف بهتری است. اگرچه که خود فعالیت‌های متدولوژی‌های مورد بحث شامل موارد مربوط به فعالیت‌های چتری می‌شوند و دارای شباهت‌ها و تفاوت‌هایی هستند که در قسمت‌های بعدی شرح داده می‌شوند، در این بخش توصیف EUP برای این مورد بهتر عمل کرده است.

۷-۱-۱-۷ چگونگی تعریف

ASD در توصیف خود به صورت فرایند–محور عمل کرده است.

EUP نیز توصیف اصلی یعنی فرایند–محور را شامل می‌شود و البته توصیفات مکمل محصول و نقش–محور برای تبیین جزئیات از دیدگاهی متفاوت برای تکمیل دیدگاه اصلی، بکار برده شده‌اند.

مقایسه: از منظر نحوه تعریف، EUP به دلیل داشتن توصیفات مکمل، بهتر عمل کرده است. البته هردو متدولوژی نحوه توصیف اصلی و مورد نیاز یعنی فرایند–محور را شامل می‌شوند.

۷-۱-۲ پوشش چرخه‌حیات عمومی ایجاد نرم‌افزار

۷-۱-۲-۱ کاوش و مدل‌سازی قلمرو مسئله

در متدولوژی **ASD** در فاز آغاز پروژه، فعالیت‌هایی نظیر مشخص کردن ماموریت پروژه، ایجاد مصنوعاتی مانند چشم‌انداز پروژه که حاوی حوزه، اندازه و زمینه پروژه می‌شود و همچنین توصیف و مشخصات محصول (طرح کلی) که شامل نتایج تحلیل و مدل‌سازی سیستم‌ها است که در فازهای بعدی از نظر عمق و وسعت به مرور غنی‌تر می‌شوند. در این فاز همچنین فهرستی از نیازمندی‌ها و مدل‌هایی از سیستم شامل وظیفه‌مندی کلی، کلاس‌های اصلی و تعاملات آن‌ها وجود دارد. این متدولوژی بر همکاری قوی تاکید دارد و اطلاعات لازم برای تولید مصنوعات و تکامل آن‌ها طی فازها از طریق مشارکت و همکاری و جلسات JAD به‌دست می‌آیند.

در **EUP** مدل‌سازی و کاوش قلمرو مسئله خیلی جدی وجود دارد. در فاز آغاز، هدف شناخت چیستی مسئله در جهت امکان‌سنجی و مدل‌سازی نیازمندی‌های سطح‌بالا و متصور شدن یک معماری برای سیستم است که انجام پروژه را امکان‌پذیر می‌سازد. این مدل‌ها و تحلیل‌ها در طی فاز بعدی یعنی تفصیل، بسیار کامل‌تر شده و در تکرارهای موجود در این فاز، به مرور شناخت، کاوش و مدل‌سازی در قلمرو مسئله تکمیل می‌شود.

مقایسه: مدل‌سازی و کاوش قلمرو مسئله تا حدی در هردو متدولوژی وجود دارد اما این فعالیت در EUP بسیار مفصل‌تر انجام می‌شود و طی فازهای ابتدایی، مدل‌های دقیق از چیستی سیستم ساخته می‌شوند. در ASD نیز علی‌رغم وجود مدل‌سازی تحلیل در فاز آغاز پروژه و تکمیل به مرور آن، این عمل بسیار سبک‌تر بوده و طی همکاری و جلسات مشارکتی، به مرور طی چرخه‌ها ایجاد و تکمیل می‌شوند.

۷-۱-۲-۲ استخراج نیازمندی‌ها

در **ASD** فاز بخصوصی برای استخراج نیازمندی‌ها تعبیه نشده است و این کار در ابتدا در فاز آغاز پروژه برای ایجاد مصنوعات ماموریت و طرح کلی که شامل فهرستی از نیازمندی‌های سطح‌بالا همراه با اولویت، وابستگی‌ها و ریسک‌های آنها، می‌شود. هدف اصلی این متدولوژی بروز یافتن به‌مرور نیازمندی‌ها طی چرخه‌ها و جلسات مشارکت با ذی‌نفعان و دریافت بازخورد است.

در متدولوژی **EUP** در طی فاز آغاز، حدود ۲۰% از نیازمندی‌ها استخراج می‌شوند که نیازمندی‌های هسته و سطح‌بالا هستند و در طی فاز تفصیل، تا ۸۰% نیازمندی‌ها استخراج می‌شوند و مابقی نیازمندی‌ها طی فاز ساخت بروز می‌یابند. EUP از تکنیک‌های ساخت‌یافته مانند use-case پشتیبانی می‌کند و همچنین یک نظام بخصوص مربوط به نیازمندی‌ها در تکرارهای این متدولوژی وجود دارد.

مقایسه: در متدولوژی EUP با استفاده از تکنیک‌هایی مانند use-case طی فاز آغاز، تفصیل و ساخت و در نظام مربوط به نیازمندی‌ها، سعی می‌شود تمام نیازمندی‌ها استخراج شوند که عمده این اتفاق در فاز تفصیل رخ می‌دهد. نظام بخصوصی نیز برای این کار در آن تعبیه شده است. در متدولوژی ASD فاز بخصوصی برای استخراج نیازمندی‌ها وجود ندارد و بیشتر به مرور و بروز نیازمندی‌ها طی چرخه‌ها تاکید شده است و صرفا نیازمندی‌های سطح‌بالا در فاز آغاز پروژه فهرست می‌شوند و سپس در طی همکاری و جلسات مشارکت در هر چرخه، تغییرات مورد نیاز و نیازمندی‌ها بروز می‌یابند.

۷-۱-۲-۳ تحلیل امکان‌پذیری

متدولوژی **ASD** بر تطبیق، حدس و همکاری تاکید دارد و فاز بخصوص برای تحلیل امکان‌پذیری در آن بیان نشده است ولی در فاز آغاز پروژه و طی فعالیت ایجاد مصنوعات ماموریت، مشخصات محصول یا طرح کلی ایجاد می‌شود که در آن نیازمندی‌ها همراه نمایش ریسک‌های مرتبط، فهرست می‌شوند و در ایجاد مصنوع برگه داده‌های پروژه[۹]، محدودیت‌ها و ریسک‌های کلیدی مرتبط با پروژه بیان می‌شوند و از مشتریان کسب تایید و اجازه ادامه پروژه گرفته می‌شود. در برنامه‌ریزی چرخه نیز مولفه‌ها بر اساس ریسک به چرخه‌ها تخصیص می‌شوند و طی بازخوردها و جلسات مشترک برای همکاری، سعی در تطبیق و برطرف کردن مشکلات می‌کنند.

در **EUP** یکی از اصلی‌ترین اهداف فاز آغاز، تحلیل امکان‌پذیری و شناسایی ریسک‌های مهم است. در این راستا، فرد خبره همراه با تیم خود در طی مدت کوتاه حداکثر ۴ هفته، با تحلیل چیستی سیستم و تکنیک‌هایی مانند ساخت نمونه‌اولیه[۱۰] هم برای امکان‌سنجی فنی و هم برای ارزیابی درک خود از نیازمندی‌ها، انجام شده و

[۹] project data sheet
[۱۰] prototype

سعی می‌کنند یک معماری برای پروژه متصور شوند و در انتها تصمیم بر ادامه دادن یا ندادن پروژه گرفته می‌شود. در انتهای فاز تفصیل باید به تمام ریسک‌ها پرداخته شده باشد.

مقایسه: شباهتی در بررسی ریسک‌های مهم در فاز ابتدایی و کسب تایید و اجازه برای ادامه پروژه از مشتریان/حامیان در هردو متدولوژی دیده می‌شود. اما بحث تحلیل امکان‌پذیری به صورت جدی در EUP بیان شده و یکی از اهداف اصلی فاز ابتدایی همین موضوع می‌باشد و در آن از تکنیک‌های مختلف مانند ساخت نمونه‌اولیه نیز استفاده شده است. بنابراین در این معیار EUP بهتر عمل می‌کند و تحلیل بهتری از امکان‌پذیری ارائه می‌کند.

۷-۱-۲-۴ طراحی معماری

در متدولوژی **ASD** فاز یا فعالیت بخصوصی برای طراحی معماری تخصیص داده نشده است و معماری به صورت بروزیابنده در چرخه‌ها طراحی می‌شود و طراحی حداقلی در ابتدا وجود دارد تا فقط کار ایجاد بتواند آغاز شود که صرفا روی مسائل ضروری تمرکز می‌کند. ASD تاکید دارد که تیم باید به صورت جمعی سیستم را درک و با همکاری یکدیگر معماری را ایجاد و اصلاح کند. همچنین مدل‌سازی رسمی و استفاده از تکنیک‌های مدل‌سازی خاص یا استفاده از زبان‌هایی مانند UML را تجویز نمی‌کند و رویکردهای سبک‌وزن و بروزیابنده را ترجیح می‌دهد و توصیه می‌کند تیم‌ها روی سادگی و تطبیق تمرکز کنند.

در **EUP** در فاز آغاز، یک معماری اولیه سطح‌بالا عمدتا با هدف امکان‌سنجی ساخته می‌شود و در طی فاز تفصیل، تکمیل شده و تثبیت می‌شود و مبنای کارها قرار می‌گیرد. در این متدولوژی، طراحی نیز جدی است و تکنیک‌های متعدد مانند استفاده از UML و ایجاد نمودارهای ساختاری و رفتاری متعدد تجویز شده است و در نهایت طی این فعالیت‌ها به یک معماری مبنای قابل اجرا[۱۱] می‌رسد.

مقایسه: در ASD که یک متدولوژی چابک است، بجای تلاش برای طراحی معماری اولیه پر جزئیات با استفاده از تکنیک‌های سنگین، بر این تاکید دارد که در طی چرخه‌ها، فقط به اندازه نیاز، درنتیجه همکاری و مشارکت تیم و با استفاده از روش‌های سبک انجام شود. درحالی که در EUP طراحی معماری بسیار جدی بوده و در فازهای آغاز و تفصیل، با تکنیک‌های متعدد انجام می‌شود و مدل‌های متعدد ایجاد شده و در نهایت یک EAB ساخته می‌شود که یک نمونه‌اولیه تکاملی است.

[۱۱] executable architectural baseline

۷-۱-۲-۵ طراحی تفصیلی

متدولوژی **ASD** طراحی تفصیلی و سنگین از ابتدا را تجویز نمی‌کند و جزو فعالیت‌های آن نیست. ترجیح می‌دهد یک طراحی حداقلی و سبک در ابتدا با هدف شروع فعالیت‌های ایجاد انجام شود و در طی چرخه‌ها و در اثر مشارکت و در راستای تطبیق بر اساس نیاز، این طراحی تکمیل و اصلاح شود.

در **EUP** طراحی تفصیلی جزو فعالیت‌ها بوده و در فاز تفصیل، در حد نیاز برای پیاده‌سازی use-case های دارای ریسک و EAB و مابقی در طی فاز ساخت برای ایجاد نیازمندی‌های سیستم، انجام می‌شود که سنگین بوده و از تکنیک‌های مختلف استفاده شده و مدل‌های متعدد ایجاد می‌شوند.

مقایسه: در EUP طراحی تفصیلی به صورت جدی و سنگین در جهت ایجاد نیازمندی‌ها وجود دارد که برای سیستم‌های بزرگ و پیچیده که نیازمندی طراحی با جزئیات هستند، مناسب‌تر است. ASD این مورد را ندارد و بر سبک و چابک بودن و طراحی حداقلی و به اندازه نیاز تاکید دارد که در طی چرخه‌ها و مشارکت‌ها ایجاد و اصلاح می‌شود.

۷-۱-۲-۶ برنامه‌نویسی

در متدولوژی **ASD** فاز مهندسی همروند مولفه که قلب بخش تکراری فرایند را تشکیل می‌دهد، شامل برنامه‌نویسی و پیاده‌سازی مولفه‌ها است. تیم‌های ایجاد به صورت موازی کار می‌کنند و مولفه‌های تخصیص داده شده را ایجاد و تحویل می‌دهند. نسخه‌های تولید شده بلافاصله وارد فرایند یکپارچه‌سازی می‌شوند. همچنین بازآرایی[۱۲] نیز به صورت مستمر در طول این فعالیت انجام می‌شود.

در **EUP** نظام مربوط به پیاده‌سازی وجود دارد که در تکرارهای مختلف در فاز آغاز برای ایجاد نمونه‌های اولیه دور ریختنی، در فاز دوم برای ایجاد نیازمندی‌های ریسکی و EAB و در فاز سوم برای ساخت اجزای سیستم، انجام می‌شود. پس از انتقال محصول به محیط کاربر نیز برای رفع مشکلات و اعمال تغییرات پیاده‌سازی دیده می‌شود.

مقایسه: در هردو متدولوژی فعالیت برنامه‌نویسی برای پیاده‌سازی اجزای سیستم دیده می‌شود.

۷-۱-۲-۷ آزمون

در متدولوژی **ASD** انجام آزمون‌ها به عنوان فعالیت مستمر در طی فاز مهندسی همروند مولفه برای صحت‌سنجی[۱۳] در ایجاد مولفه‌های سیستم، وجود دارد. همچنین در این فاز، برنامه‌های آزمون سطح سیستم و موارد آزمون

[۱۲] refactoring
[۱۳] verification

برای تضمین کیفیت نهایی آماده می‌شوند. در طی بازبینی کیفیت در جلسات با مشتری، اعتبارسنجی[۱۴] انجام می‌شود. در فاز نهایی نیز آزمون‌های سطح سیستم با هدف اصلی اعتبارسنجی انجام و بر اساس نتایج آزمون‌ها تصمیم گرفته می‌شود که آیا سیستم منتشر شود یا چرخه ایجاد جدید آغاز گردد.

در **EUP** در فاز تفصیل، معماری مبنای قابل‌اجرا و نیازمندی‌های ریسکی که پیاده‌سازی شده‌اند طی نظام تست، مورد آزمون قرار می‌گیرد. در فاز ساخت، طی تکرارهای متعدد، در نظام تست، موارد پیاده‌سازی شده مورد آزمون قرار می گیرند. در این فاز تست آلفا نیز داریم. همچنین در فاز انتقال، نسخه‌ای که به محیط کاربر منتقل شده است، مورد آزمون های متعدد از جمله تست سیستم، کارایی، امنیت، تست پذیرش و غیره قرار می‌گیرد. در تست آلفا، یک محیط شبیه‌سازی شده از محیط کاربر در محیط ایجاد ساخته می‌شود و کاربر آن را اجرا و بازخورد می‌دهد. در تست بتا، نرم‌افزار برای مشتریان بالقوه فرستاده شده و کاربران در محیط خود نصب کرده و بازخورد می‌دهند و تست پذیرش در نهایت که سیستم بطور کامل در محیط کاربر استقرار یافت، انجام می‌شود.

مقایسه: در هردو متدولوژی انجام تست‌های متعدد وجود دارد. در ASD انجام تست‌های مستمر در حین پیاده‌سازی مولفه‌ها و همچنین آزمون‌های سطح سیستم و اعتبارسنجی‌های لازم دیده می‌شود. در EUP نیز طی نظام مشخص برای آزمون، در تکرارهای مختلف، آزمون‌هایی در سطوح مختلف انجام می‌شود.

۷-۱-۲-۸ استقرار

در متدولوژی **ASD** طی فاز تضمین کیفیت نهایی و انتشار، آزمون‌های لازم انجام و مشکلات رفع می‌شوند سپس در صورتی که سیستم کیفیت تعیین شده را داشته باشد، به محیط کاربر منتقل شده و فعالیت استقرار انجام می‌شود. در این حین فعالیت‌های تبدیل سیستم، آموزش و آماده‌سازی مستندات نیز بیان شده‌اند.

در **EUP** یک نظام بخصوص برای استقرار تعریف شده است و در طی فازهای متعدد مانند ساخت و انتقال، سیستم قابل ارائه، به محیط کاربر منتقل شده و تغییرات لازم انجام، سیستم نصب و در نهایت برای انجام آزمون پذیرش آماده می‌شود.

مقایسه: هردو متدولوژی فعالیت‌های مربوط به استقرار محصول را پوشش داده‌اند. البته این فعالیت در EUP به صورت یک نظام مشخص و در ASD طی فاز نهایی انجام می‌شود.

---

[۱۴] validation

۹-۲-۱-۷ مراقبت و نگهداری

در متدولوژی **ASD** تدابیری برای مراقبت و نگهداری محصول پس از انتقال به محیط کاربر و بستن پروژه اندیشیده نشده است. در طی ساخت محصول و در چرخه‌های متعدد، بر اساس جلسه‌های مشارکت و بازخوردها، سیستم با تغییرات تطبیق پیدا کرده و مشکلات نیز رفع می‌شوند ولی بعد از استقرار سیستم در محیط کاربر، اشاره‌ای به نحوه و فعالیت‌های مربوط به مراقبت و نگهداری نشده است.

در **EUP** فاز جدید production با تمرکز نگهداری نرم‌افزار تا زمانی که یا با نسخه‌ای جدید جایگزین شود با اجرای مجدد چرخه‌حیات، یا بازنشسته و حذف شود، اضافه شده است. در این فاز هیچ تکراری وجود ندارد. این فاز تا حدی شبیه به فاز نگهداری در چرخه‌حیات کلی ایجاد نرم‌افزار است؛ زیرا بیشتر با عملیات و پشتیبانی از سیستم سروکار دارد. اما برخلاف نگهداری کلاسیک، هر نیازی به تغییر سیستم، حتی رفع یک خطا، منجر به شروع مجدد چرخه ایجاد خواهد شد. همچنین نظام جدید عملیات و پشتیبانی نیز اضافه شده است که مربوط به مسائل مرتبط با عملیات و پشتیبانی از سیستم است که معمولا با فاز نگهداری در چرخه‌حیات کلی ایجاد نرم‌افزار مرتبط است. این نظام شامل آموزش کارکنان عملیات و پشتیبانی نیز می‌شود. در طول نظام production و بازنشستگی، این نظام فعالیت‌های کلاسیک نگهداری را پوشش می‌دهد یعنی کارکنان عملیات، نرم‌افزار را فعال نگه می‌دارند، نسخه‌برداری‌های لازم و پردازش‌های دسته‌ای را انجام می‌دهند و کارکنان پشتیبانی با کاربران ارتباط برقرار می‌کنند تا به آن‌ها در کار با نرم‌افزار کمک کنند.

مقایسه: در ASD به فعالیت‌ها و نحوه انجام مراقبت و نگهداری بعد از انتقال محصول به محیط کاربر اشاره نشده است و در این زمینه ضعف دارد. در EUP نیز علی‌رغم این که فاز مستقل برای نگهداری تعریف شده است، این امر کافی نیست زیرا هرگونه تغییری که موردنیاز باشد، حتی رفع یک خطا، منجر به شروع دوباره فرایند ایجاد خواهد شد.

۳-۱-۷ پشتیبانی از فعالیت‌های چتری

۱-۳-۱-۷ مدیریت ریسک

متدولوژی **ASD** از تکنیک‌های متعدد مدیریت ریسک استفاده می‌کند. فرایند آن تکراری–افزایشی است و تاکید بر همکاری و مشارکت فعالانه ذی‌نفعان دارد. هم‌زمان با ایجاد مولفه‌های سیستم، به صورت مستمر صحت‌سنجی و در انتهای هر چرخه اعتبارسنجی وجود دارد[۱۵] و بازبینی‌های مکرر از محصول انجام می‌شود. مولفه‌های تعریف شده برای ایجاد، بر اساس ریسک‌های مرتبط به چرخه‌ها تخصیص داده می‌شوند. مولفه‌هایی

[۱۵] continuous V&V

که توسط تیم‌ها که به صورت موازی کار می‌کنند، به طور روزانه یا هفتگی به عنوان نسخه‌های ساخته شده تحویل و بلافاصله وارد فرایند یکپارچه‌سازی می‌شوند. همچنین در فاز آغاز پروژه، در ایجاد طرح‌کلی و برگه داده‌های پروژه، ریسک‌های کلیدی بررسی و نمایش داده می‌شوند.

در **EUP** در فاز آغاز، ریسک‌های مهم شناسایی و بررسی می‌شوند و سند ارزیابی ریسک تولید می‌شود. از تکنیک‌هایی مانند ساخت نمونه‌اولیه دور ریختنی و نمونه‌اولیه اثبات مفهوم[۱۶]، برای امکان‌سنجی فنی و همچنین سنجش درک از نیازمندی‌ها استفاده می‌شود. در فاز تفصیل نیازمندی‌های دارای ریسک طراحی و پیاده‌سازی می‌شوند و باید در انتهای فاز تفصیل تمام ریسک‌ها بررسی شده باشند. این متدولوژی نیز به صورت تکراری–افزایشی اجرا می‌شود که در مدیریت ریسک کمک می‌کند. همچنین برای محصولات شامل توجیه اقتصادی و برنامه‌ریزی و غیره از کاربر بازخورد دریافت می‌شود و مشارکت فعال کاربر در نظر گرفته شده است. در نظام تست در هر تکرار، یکپارچه‌سازی نیز انجام می‌شود.

مقایسه: هردو متدولوژی از تکنیک‌های مفید برای مدیریت ریسک استفاده می‌کنند و شباهت‌هایی نظیر تکراری–افزایشی بودن فرایند دارند. البته در بعضی تکنیک‌ها مانند تحلیل اولیه امکان‌پذیری و ساخت نمونه‌اولیه، EUP بهتر عمل می‌کند و در بعضی دیگر مانند صحت‌سنجی و اعتبارسنجی مستمر و مشارکت فعال کاربر، ASD بهتر عمل می‌کند. علی‌رغم تفاوت‌ها هردو به مدیریت ریسک از طریق تکنیک‌های متعدد پرداخته‌اند.

۷-۱-۳-۲ مدیریت پروژه

در متدولوژی **ASD** از تکنیک‌های سبک‌تر برای برنامه‌ریزی، زمان‌بندی و کنترل استفاده می‌شود. عمدتا در فاز برنامه‌ریزی چرخه تطبیقی، با تعیین بازه‌های زمانی برای کل پروژه و هر یک از چرخه‌ها انجام می‌شود. بازه‌های زمانی چرخه‌ها نیز معمولا بین دو تا هشت هفته طول می‌کشند. در این فاز فعالیت برنامه‌ریزی پروژه نیز داریم که شامل ایجاد زمان‌بندی‌های بافر شده برای چرخه‌های ایجاد می‌شود. البته برنامه‌های تولید شده در هر تکرار مورد بازبینی قرار می‌گیرند و درس‌های آموخته شده و تغییرات مورد نیاز در آن اعمال می‌شود. این متدولوژی روش‌ها و ابزارهای خاص مانند نمودار گنت[۱۷] یا PERT تجویز نمی‌کند. همکاری و مشارکت تیمی یکی از مفاهیم اصلی در ASD است و طی ایجاد جامعه پروژه، جلسات JAD و مسئولیت‌های مشترک انجام می‌شود. البته به صراحت تکنیک‌ها و مکانیزم‌های ارتباط بین‌تیمی یا داخل تیم، بیان نشده‌اند.

در **EUP** فعالیت‌های مدیریتی پروژه تحت نظام مدیریت پروژه انجام می‌شوند. در این متدولوژی از روش‌هایی همچون نمودار گنت و PERT استفاده می‌شود. در انتهای فاز آغاز، برای تکرارهای ابتدایی فاز

[۱۶] proof of concept prototype
[۱۷] gantt chart

تفصیل، برنامه دقیق‌تر ارائه شده و برای سایر تکرارها برنامه‌ریزی تخمینی و کلی است و برای فاز بعدی نیز به همین رویکرد انجام می‌شود. برنامه‌ها به طور مکرر در فازها مورد بازبینی و اصلاح قرار می‌گیرند. در EUP یک نظام جدید مدیریت سازمان تعبیه شده است که در آن به فعالیت‌های مدیریتی سطح سازمان پرداخته می‌شود. ارتباطات داخل تیم‌ها و بین تیم‌ها نیز از اهمیت زیادی برخوردار است و طی جلسات روزانه، برنامه‌ریزی‌های تکرار و بازبینی‌ها انجام می‌شود. همچنین از مواردی همچون مستندسازی و مدل‌سازی مفصل برای ارتباط و به اشتراک‌گذاری دانش بین تیم‌ها نیز بهره می‌برد.

مقایسه: متدولوژی EUP به دلیل داشتن مدیریت پروژه ساختار یافته و به خصوص رسیدگی به دغدغه‌های مدیریت سازمان، برای پروژه‌های بزرگ و پیچیده سطح سازمان مناسب‌تر است. همچنین مدل‌سازی و مستندسازی سنگین و مفصل در آن به نوبه خود به ارتباط بین تیم‌ها حتی در موقعیت جغرافیایی متفاوت نیز کمک می‌کند. از طرفی مدیریت در ASD سبک‌تر بوده و بر تطبیق‌پذیری، چابکی، همکاری و مشارکت تاکید دارد و طی چرخه‌ها و جلسات بازبینی، برنامه‌ها بروزرسانی می‌شوند.

۷-۱-۳-۳ تضمین کیفیت

متدولوژی **ASD** توجه خوبی به کیفیت دارد. یکی از فازهای قسمت ایجاد تکراری فرایند، بازبینی کیفیت است که در آن طی جلسات با مشتری، بازخورد در رابطه با مولفه‌های ایجاد شده دریافت می‌شود و اعتبارسنجی مستمر وجود دارد. در مهندسی همروند مولفه، همراه پیاده‌سازی به صورت مستمر، بازآرایی و آزمون انجام می‌شود و صحت‌سنجی مستمر وجود دارد. همچنین در فاز تضمین کیفیت نهایی، آزمون‌ها و اعتبارسنجی‌های سطح سیستم انجام و مشکلات رفع می‌شوند.

در **EUP** نظام بخصوص آزمون وجود دارد و همچنین در این متدولوژی انجام آزمون‌های واحد، ادغام و سیستم دیده می‌شوند. این نظام در فازها گسترده شده است و در نظام تفصیل برای EAB و نیازمندی‌های ریسکی که پیاده‌سازی شده‌اند و در نظام ساخت در حین پیاده‌سازی اجزای سیستم و در نظام انتقال موارد آزمون بعد از استقرار از جمله اعتبارسنجی و آزمون پذیرش انجام می‌شوند. همچنین این متدولوژی مبتنی بر نیازمندی بوده و از این طریق رهگیری به نیازمندی‌ها محقق می‌شود.

مقایسه: متدولوژی ASD به دلیل ماهیت مشارکتی آن و جلسات مداوم طی چرخه‌ها، در معیار صحت‌سنجی و اعتبارسنجی مستمر بهتر عمل می‌کند و بازخوردهای دریافت شده و اعمال آن‌ها باعث افزایش کیفیت می‌شود. همچنین هردو متدولوژی آزمون در سطح واحد و سیستم انجام می‌دهند. متدولوژی EUP به دلیل مبتنی بودن بر نیازمندی، رهگیری مستقیم به نیازمندی‌ها دارد که از این جهت بهتر است.

۷-۱-۴ بی‌درزی و همواری انتقال

در **ASD** بی‌درزی خوب پوشش داده نشده است. در این متدولوژی از شروع یعنی نیازمندی‌ها تا پیاده‌سازی، مدل‌سازی مناسب برای پوشش درزها وجود ندارد و وابستگی زیادی به مشارکت و همکاری تیم‌ها دارد اما به دلیل این که فرایند تکراری–افزایشی است و کارها با پیمانه‌های کوچک انجام می‌شوند، خط بین فازها کمرنگ می‌شود و انتقال هموار به خوبی دیده می‌شود. همچنین به دلیل این که ASD ایجاد مدل‌های متعدد و سنگین را تجویز نمی‌کند، در نتیجه ناسازگاری کم‌تر نیز دیده می‌شود.

متدولوژی **EUP** با این که از مفاهیم شیگرا برای مدل‌سازی و پیاده‌سازی سیستم استفاده می‌کند و از این نظر تغییر پارادایم و الگوها در آن دیده نمی‌شود، اما برخلاف RUP که استفاده از UML را اجبار کرده بود، در این متدولوژی می‌توان از روش‌هایی نظیر DFD نیز برای مدل‌سازی کسب و کار استفاده کرد که دید شیگرا ندارد و این بر بی‌درزی خدشه وارد می‌کند. همچنین به دلیل این که نگاه use-case شیگرا نیست و در تبدیل آن به نمودار توالی[۱۸] مقداری درز ایجاد می‌شود، می‌توان از روش‌هایی همانند ساخت نمودار فعالیت[۱۹] و تکنیک‌های آن مانند خط شنا، تا حدی این درز را پوشش داد. همچنین فعالت‌ها در این متدولوژی ادامه طبیعی یکدیگر هستند و در آن، مدل‌ها به مرور طی فرایند تکراری–افزایشی، تکامل پیدا می‌کنند و یا از روی مدل‌های دیگر ساخته می‌شوند بنابراین انتقال هموار دیده می‌شود.

مقایسه: متدولوژی EUP علی‌رغم اجازه بر استفاده از روش‌هایی همچون DFD در کنار UML که پارادایم‌های متفاوتی دارند، پوشش بهتری بر بی‌درزی نسبت به ASD دارد. زیرا مدل‌سازی در آن جدی است و از نیازمندی تا پیاده‌سازی، مدل‌های متعدد ایجاد می‌شوند تا فاصله بین فعالیت‌ها هموار شوند. هردو تکراری–افزایشی اجرا می‌شوند ولی ASD یک متدولوژی چابک بوده و به دلیل انجام کارها در پیمانه‌های کوچک‌تر، خط بین فازها کمرنگ‌تر بوده و همچنین به دلیل کم‌تر بودن ناسازگاری مدل‌ها، همواری انتقال در آن بهتر دیده می‌شود.

۷-۱-۵ مبتنی بودن بر نیازمندی‌ها (وظیفه‌ای و غیر وظیفه‌ای)

در متدولوژی **ASD** در فاز آغاز پروژه و طی فعالیت ایجاد مصنوعات ماموریت، در مشخصات محصول یا طرح کلی، فهرستی از نیازمندی‌های سطح بالا ایجاد می‌شود تا بتوان فرایند ایجاد را آغاز نمود. این متدولوژی بر حدس–همکاری–یادگیری تاکید دارد و طی جلسات مشارکتی و بازخوردها در هر چرخه، برنامه‌های ایجاد شده مورد بازبینی قرار می‌گیرند تا درس‌های آموخته شده بازتاب شوند همچنین نیازمندی‌های بروز یافته در برنامه

[۱۸] sequence diagram
[۱۹] activity diagram

چرخه‌ها دخیل شوند. در نتیجه علی‌رغم این که این متدولوژی قالبی برای ثبت نیازمندی‌ها مانند use-case تجویز نکرده است، ولی تمام کارها در آن مبتنی بر نیازمندی انجام می‌شوند و مولفه‌های مورد نیاز سیستم، ایجاد و در هرچرخه نیز مطابق بازخوردها بروزرسانی و اصلاح می‌شوند.

متدولوژی **EUP** مبتنی بر نیازمندی بوده و در آن می‌توان از روش‌هایی مانند use-case استفاده و نیازمندی‌ها را ثبت کرد و تمام کارها در طراحی و پیاده‌سازی مبتنی بر نیازمندی پیش می‌روند. بنابراین رهگیری مستقیم به نیازمندی در این متدولوژی دیده می‌شود. در فاز آغاز ۲۰% نیازمندی‌ها، در فاز تفصیل تا ۸۰% و مابقی در فاز ساخت استخراج می‌شوند و سیستم بر اساس آن‌ها ساخته می‌شود.

مقایسه: در هردو متدولوژی بخشی از نیازمندی‌های سطح بالا در ابتدای فرایند ثبت می‌شوند و در هردو به بروزیابندگی و تکامل نیازمندی‌ها توجه شده است. در EUP تمام نیازمندی‌ها تا انتهای فاز ساخت استخراج می‌شوند. در ASD نیز به صورت تطبیقی در طی اجرای چرخه‌های تکراری ایجاد نرم‌افزار، در نتیجه جلسات مشارکتی و بازخوردها، تغییرات جدید توسط مشتری درخواست شده، ثبت و اجرا می‌شوند.

۷-۱-۶ آزمون‌پذیری، ملموس بودن و قابلیت رهگیری به نیازمندی‌ها

۷-۱-۶-۱ آزمون‌پذیری

در متدولوژی **ASD** تعداد مصنوعات تولید شده کم بوده و همچنین از سادگی برخوردارند و مصنوعاتی که صرفا به عنوان زینت تولید شوند وجود ندارند در نتیجه از آزمون‌پذیری خوبی برخوردارند.

در **EUP** که مدل‌سازی بسیار سنگین است و مدل‌های متعدد طی فازهای مختلف بوجود می‌آیند و همچنین پیچیدگی موجود در آن‌ها و گاها مدل‌هایی که وجودشان باعث کامل‌شدن بقیه مدل‌ها نشده و صرفا تزیین میکنند، آزمون‌پذیری را کاهش می‌دهد.

مقایسه: در این معیار، متدولوژی ASD به دلیل سادگی مصنوعات و تعداد کم آن‌ها، آزمون‌پذیری بهتری نسبت به EUP دارد.

۷-۱-۶-۲ ملموس بودن

در متدولوژی **ASD** مصنوعاتی که تولید می‌شوند، حداقلی بوده و در ایجاد آن‌ها سعی می‌شود از روش‌های ساده استفاده کرد که برای کاربران ملموس و قابل درک باشد. همچنین موارد ایجاد شده همگی به اندازه نیاز و در جهت امور فرایند بوده و از این نظر بر ایجاد کنندگان نیز ملموس و قابل درک‌اند.

در **EUP** طی فعالیت‌های مختلف، مصنوعاتی نظیر سند چشم‌انداز، سند توجیه اقتصادی یا سند ارزیابی

ریسک و غیره ایجاد می‌شوند که ساده بوده و از سمت کاربر نیز قابل درک هستند و نیاز به تایید آن‌ها نیز وجود دارد. همچنین برخی مدل‌های UML ساخته شده مانند نمودار کلاس در تحلیل، برای کاربر ملموس و قابل درک هستند اما برخی دیگر که مربوط به طراحی بوده و جزئیات پیاده‌سازی در آن‌ها وجود دارد، توسط کاربر ملموس نیستند. از منظر ایجاد کنندگان نیز مورد کاربرد هرکدام از مدل‌ها به شرط این که مکمل هم‌دیگر بوده و زینتی نباشند و در راستای انجام فعالیت‌های فرایند باشند و همچنین این متدولوژی سنگین در پروژه مناسب استفاده شود، ملموس و قابل درک هستند.

مقایسه: مصنوعات تولید شده در ASD به دلیل تعداد کم و سادگی، هم برای کاربران و هم برای ایجاد کنندگان ملموس هستند. در EUP نیز ملموس بودن تا حد خوبی وجود دارد و برخی مصنوعات برای کاربر کاملا ملموس بوده و برخی مانند مدل‌های طراحی، کم‌تر ملموس‌اند.

۷-۱-۶-۳ قابلیت رهگیری به نیازمندی‌ها

در متدولوژی **ASD** نیازمندی‌های سطح بالا در فاز آغاز پروژه فهرست می‌شوند تا امکان شروع فرایند ایجاد فراهم شود و پس از آن، با رویکرد تطبیق‌پذیری در طی چرخه‌ها و جلسات مشارکتی، بازخوردها و درخواست‌های تغییر توسط مشتری دریافت، ثبت و اجرا می‌شوند و این‌گونه تمام اقدامات در راستای انجام درخواست‌های مشتری بوده و قابلیت رهگیری این‌گونه فراهم می‌شود.

متدولوژی **EUP** مبتنی بر نیازمندی بوده و در آن تمام کارها مبتنی بر نیازمندی انجام می‌شوند و در نتیجه آن، رهگیری مستقیم به نیازمندی‌ها وجود دارد.

مقایسه: هردو متدولوژی رهگیری به نیازمندی‌ها را به خوبی پشتیبانی می‌کنند. در EUP به دلیل مبتنی بودن بر نیازمندی‌ها، رهگیری مستقیم وجود دارد و در ASD نیز تمام کارها طی جلسات مشارکتی و درخواست‌های مشتری انجام میگیرد.

۷-۱-۷ تشویق مشارکت فعال کاربر

در متدولوژی **ASD**، مشارکت فعال کاربر یکی از اصول اصلی است و همکاری و مشارکت بین کاربر و ایجاد کنندگان در طول چرخه‌حیات را تشویق می‌کند و مفهومی به نام جامعه پروژه شامل افراد درگیر یا تحت تاثیر در پروژه، دارد. در طی جلسات مشارکت در چرخه ایجاد، بازخوردهای مستمر مشتری، نیازمندی‌ها، اولویت‌ها و تصمیمات را شکل می‌دهند و در طول ایجاد مولفه‌ها اعتبارسنجی[۲۰] مستمر که یکی از تکنیک‌های مهم مدیریت ریسک و تضمین کیفیت است، دیده می‌شود.

[۲۰] validation

متدولوژی **EUP** نیز مشارکت فعال کاربر را تشویق و در متدولوژی تعبیه می‌کند اما فرکانس آن در مقایسه با ASD کم‌تر بوده و به عنوان یک اصل اساسی در متدولوژی قرار نگرفته است و مشارکت کاربر ساختار یافته‌تر و محدود به برخی فعالیت‌ها است. برای مثال گرفتن تایید برای برخی مصنوعات.

مقایسه: همان‌طور که اشاره شد، متدولوژی ASD که یک متدولوژی چابک بوده و مشارکت فعال کاربر در آن یک اصل اساسی است و اگرچه این معیار در EUP نیز تا حدی پوشش داده شده است، اما ASD پوشش بهتری در این زمینه دارد.

۷-۱-۸ قابلیت اجرا و قابلیت اجرا به صورت کارا

۷-۱-۸-۱ قابلیت اجرا

متدولوژی **ASD** فرایند ساده‌ای دارد و در نتیجه معیار قابل اجرا بودن را به خوبی پوشش می‌دهد. البته این متدولوژی چابک بوده و به دلیل نداشتن مدل‌سازی و مستندسازی سنگین و بهره نبردن از تکنیک‌های خاصی که در متدولوژی‌های سنگین وزن‌تر وجود دارد، برای دامنه کوچک‌تری از پروژه‌ها قابل اجرا است.

متدولوژی **EUP** بسیار سنگین وزن بوده و فرایند پیچیده‌ای دارد که طی آن‌ها مدل‌ها و مستندات مفصلی ایجاد می‌شوند و این موضوع اجرای EUP به همان شکل که هست را بسیار سخت می‌کند و باعث کاهش عملکرد این متدولوژی در معیار قابل اجرا بودن می‌شود و برای اجرای آن نیاز به شخصی‌سازی متناسب با پروژه دارد که کار آسانی نیست.

مقایسه: قابلیت اجرای متدولوژی ASD به دلیل ساده بودن فرایند، بیشتر است و EUP فرایند پیچیده داشته و برای اجرا نیاز به شخصی‌سازی دارد. البته مدل‌سازی و مستندسازی سنگین در EUP باعث می‌شود برای برخی پروژه‌های پیچیده‌تر و همچنین پروژه‌هایی که دغدغه‌های سازمانی دارند، مناسب و قابل اجرا باشد.

۷-۱-۸-۲ قابلیت اجرا به صورت کارا

متدولوژی **ASD** در معیار قابلیت اجرا به صورت کارا نسبتا خوب عمل می‌کند. فعالیت‌ها در آن ساده بوده و سعی شده است از کارهایی که تمرکز ایجاد کنندگان را مخدوش میسازند، دوری کرده و جلسات مشارکتی و تاکید این متدولوژی بر همکاری و دریافت بازخورد در هر چرخه، در حفظ تمرکز افراد تیم، موفق عمل می‌کند. در فاز برنامه‌ریزی چرخه تطبیقی، نوشتن بیانیه‌های هدف[۲۱] باعث متمرکز ساختن تلاش‌های تیم ایجاد می‌شود. البته تاکید بر مکالمات رو در رو طی جلسات، در همه شرایط و برای همه چیز مناسب نبوده و در صورت استفاده بیش

[۲۱] objective statements

از حد از این تکنیک، می‌تواند خطا خیز باشد. همچنین این متدولوژی بر ابزار یا تکنولوژی خاصی وابستگی ندارد.

فرایند متدولوژی **EUP** بسیار پیچیده بوده و اجرای آن به همان شکل که هست، ناکارمد بوده و نیاز به شخصی‌سازی برای پروژه دارد و به دلیل این که مدل‌سازی، مستندسازی و فعالیت‌های مختلف مدیریتی اعم از مدیریت سازمانی در آن به صورت جدی وجود دارد، بیشتر مناسب پروژه با سایز و پیچیدگی بالا و دارای دغدغه‌های سازمانی است. البته این متدولوژی از تکنیک‌هایی نظیر معماری سیستم و مبتنی بودن بر نیازمندی برای حفظ تمرکز تیم ایجاد استفاده می‌کند و همچنین وابستگی بیش از حد به تکنیک‌هایی نظیر مکامله رو در رو ندارد. پیکربندی و شخصی‌سازی آن برای پروژه هدف نیز نیاز به استفاده از ابزار دارد.

مقایسه: در کل متدولوژی ASD در این معیار بهتر عمل می‌کند که عمدتا به دلیل سادگی واحدهای فرایند و نداشتن فعالیت‌هایی که تمرکز تیم ایجاد را به هم بزند که البته در EUP نیز برای این منظور از تکنیک‌هایی نظیر معماری سیستم و مبتنی بودن بر نیازمندی‌ها استفاده شده است. متدولوژی ASD وابستگی به ابزار یا تکنولوژی خاصی ندارد. برخلاف RUP که صرفا از UML استفاده می‌کرد که در بعضی موارد مانند مدل‌سازی کسب و کار، کارا نبود، در EUP استفاده از روش‌هایی مانند DFD نیز مجاز است. همچنین EUP مدیریت پروژه خوبی دارد و دغدغه‌های سازمانی را نیز پوشش می‌دهد اما در نهایت به دلیل پیچیدگی بسیار زیاد فرایند EUP، باید برای پروژه مناسب انتخاب شده و از ابزار برای شخصی‌سازی استفاده شود که کار آسانی نیست.

۷-۱-۹ قابلیت مدیریت پیچیدگی

در فرایند متدولوژی **ASD** تکنیک‌های مدیریت پیچیدگی یعنی لایه‌بندی[۲۲] و جزءبندی[۲۳] دیده می‌شود. این متدولوژی چرخه‌حیات سطح‌بالا و کلی حدس–همکاری–یادگیری را ارائه می‌کند و در لایه‌ای پایین‌تر که انتزاع کم‌تری دارد، فازهای آغاز پروژه، برنامه‌ریزی چرخه تطبیقی، مهندسی همروند مولفه، بازبینی کیفیت و تضمین کیفیت نهایی و انتشار را توصیف می‌کند و همچنین برای هرکدام از فازها فعالیت‌های بخصوص و مصنوعاتی که طی آن فعالیت‌ها باید ایجاد شوند را شرح می‌دهد که در فصل مربوط به ASD فهرست شدند. فازهای میانی نیز موتور ایجاد تکراری فرایند را تشکیل می‌دهند. بنابراین لایه‌بندی در فرایند استفاده شده است. همچنین شکستن چرخه‌حیات به فازها و فعالیت‌های مجزا نیز در فرایند ASD دیده می‌شود.

در متدولوژی **EUP** نیز استفاده از تکنیک‌های مذکور را می‌توان دید. این متدولوژی چرخه‌حیات کلی را در لایه‌ای دقیق‌تر به ۶ فاز تقسیم کرده است که شامل فاز آغاز، تفصیل، ساخت، انتقال، اجرا و بازنشستگی است. همچنین ذیل هرکدام از فازها تعداد تکرار انجام می‌شود که هرکدام از تکرارها به نظام‌های مدل‌سازی کسب و

[۲۲] layering
[۲۳] partitioning

کار، نیازمندی‌ها، تحلیل و طراحی، پیاده‌سازی، آزمون، استقرار، مدیریت تغییر و پیکربندی، مدیریت پروژه، محیط، عملیات و پشتیبانی و مدیریت سازمان، تقسیم می‌شوند.

مقایسه: هردو متدولوژی از تکنیک‌های مدیریت پیچیدگی مانند لایه‌بندی و جزءبندی در فرایند خود استفاده کرده‌اند. البته با چرخه‌حیات، فازها و فعالیت‌های متفاوت.

۷-۱-۱۰ قابلیت‌های گسترش، مقیاس‌پذیری، پیکربندی و انعطاف

۷-۱-۱۰-۱ قابلیت گسترش

متدولوژی **ASD** یک موتور ایجاد تکراری دارد که مولفه‌ها در آن طی چرخه‌ها ایجاد می‌شوند. اما این متدولوژی نقاط گسترش[۲۴] و مکانیزم‌های لازم برای این کار را به صراحت مشخص نکرده است و بنابراین قابلیت گسترش ندارد.

متدولوژی **EUP** فرایند بسیار بزرگ و پیچیده‌ای دارد و اساسا برای استفاده کارا از آن، باید به روش‌های مختلف مانند هرس یا تجمیع قطعات لازم، آن را متناسب با پروژه هدف ساخت و این متدولوژی به صورت هسته قابل گسترش نبوده، نقاط گسترش و مکانیزم‌هایی برای این کار تعریف نکرده است.

مقایسه: هردو متدولوژی در این معیار ضعف دارند. متدولوژی ASD با وجود چابک، سبک و ساده بودن، نقاط گسترش و مکانیزم‌های لازم را توصیف نکرده است. متدولوژی EUP نیز بسیار بزرگ و پیچیده بوده و برای استفاده نیاز به شخصی‌سازی دارد.

۷-۱-۱۰-۲ مقیاس‌پذیری

متدولوژی **ASD** یک متدولوژی چابک و سبک‌وزن است که در آن مدل‌سازی، مستندسازی و فعالیت‌های مدیریتی سنگین وجود ندارد و بسیار بر همکاری و جلسات مشارکتی وابسته است. برای این که متدولوژی بتواند بر پروژه‌هایی با اندازه‌های بزرگ یعنی پروژه‌هایی که تعداد افرادی که به صورت همزمان در پروژه دخیل هستند، استفاده شود، نیاز به مدیریت پروژه و مدیریت تیم قوی دارد. همچنین سطوح بالای بحرانیت، نیاز به مدل‌سازی سنگین و حتی در بالاترین سطح آن باید از روش‌های فرمال استفاده کرد. بنابراین این متدولوژی توانایی این که در پروژه با سطوح بحرانیت و اندازه مختلف استفاده شود، ندارد. اما بنابر ایده و چرخه‌حیات آن که مبتنی بر حدس–همکاری–یادگیری بوده و بر تطبیق‌پذیری تاکید دارد و طی چرخه‌ها بازخورد گرفته و می‌آموزد، برای پروژه‌هایی که قطعیت و اطمینان در آن‌ها کم است، مناسب‌تر است.

[۲۴] extension points

در متدولوژی **EUP** به دلیل وجود مدل‌سازی، مستندسازی و فعالیت‌های مدیریتی سنگین، می‌توان از آن در پروژه با اندازه‌های بالاتر و سطوح بحرانیت بالاتر استفاده کرد البته به دلیل این که پروژه‌های با بالاترین سطح بحرانیت نیاز به روش‌های فرمال دارند، علی‌رغم این که در EUP می‌توان از UML استفاده کرد که برای پشتیبانی از فرمالیزم، OCL را دارد، این کافی نبوده و بهتر است از روش‌های فرمال استفاده کرد. برای استفاده از این متدولوژی برای پروژه با اندازه و پیچیدگی کم، نیاز به پیکربندی و شخصی‌سازی به روش هرس یا تجمیع قطعات دارد تا صرفا قسمت‌های مورد نیاز متدولوژی استفاده شوند که این کار بسیار سخت است.

مقایسه: متدولوژی ASD به دلایلی که ذکر شد مقایس‌پذیری خوبی ندارد و EUP برای پروژه‌های ذکر شده قابلیت استفاده بهتری دارد. البته همان‌طور که ذکر شد حتی EUP نیز برای برخی پروژه‌ها مانند پروژه‌هایی که بالاترین سطح بحرانیت را دارند یا پروژه‌هایی که پیچیدگی نداشته و اندازه کوچکی دارند مناسب نیست.

۷-۱-۱۰-۳ قابلیت پیکربندی

در متدولوژی **ASD** صحبتی در رابطه با پیکربندی و مکانیزم‌های مربوطه نشده است.

متدولوژی **EUP** فرایند پیچیده و سنگینی دارد و نیاز دارد در ابتدای پروژه با استفاده از ابزار، متناسب با موقعیت پروژه هدف با روش‌هایی نظیر هرس یا تجمیع قطعات متدولوژی، پیکربندی شود که البته کار آسانی نیست.

مقایسه: متدولوژی EUP همراه با پشتیبانی ابزاری خوب، قابلیت پیکربندی در ابتدای پروژه برای متناسب ساختن آن برای موقعیت پروژه‌ای هدف را دارد اما ASD صحبتی در این رابطه نکرده است.

۷-۱-۱۰-۴ انعطاف‌پذیری

متدولوژی **ASD** یک متدولوژی چابک بوده و در انتهای موتور ایجاد تکراری آن یعنی فاز بازبینی کیفیت و در فعالیت برگزاری جلسه پس از مرگ[۲۵]، عملکرد تیم‌ها، فرایند و اثربخشی روش‌های به‌کار برده شده بررسی و از درس‌های آموخته شده استفاده می‌شود تا تاثیر منفی بر تکرارهای بعدی نداشته باشند. بنابراین انعطاف‌پذیری خوبی دارد.

در متدولوژی **EUP** اگرچه به اندازه متدولوژی‌های چابک نیست ولی طی نظام محیط[۲۶] در فازها، فرایند بر اساس نیاز پروژه یا سازمان، تطبیق پیدا می‌کند در نتیجه تاحدی انعطاف‌پذیری دارد.

مقایسه: هردو متدولوژی موارد انعطاف‌پذیری را پوشش می‌دهند ولی در متدولوژی ASD که چابک بوده و

[۲۵] post mortem
[۲۶] environment

در هر چرخه طی جلساتی، عملکرد روش‌های استفاده شده بررسی و درس‌های آموخته شده در متدولوژی اعمال می‌شود، تا حدی در این معیار بهتر عمل می‌کند.

۷-۱-۱۱ حوزه کاربرد

متدولوژی **ASD** به دلیل چرخه‌حیات و فعالیت‌هایی که تعریف می‌کند و تاکید آن بر تطبیق‌پذیری و همکاری، برای پروژه‌هایی که پویا، complex و در معرض تغییرات مکرر قرار دارند و نیاز به انعطاف دارند، مناسب‌تر است و می‌توان از آن برای سیستم‌های اطلاعاتی که تطبیق‌پذیری مهم‌تر از مدل‌سازی سنگین است استفاده نمود. البته در آن حداقل انتظارات مدل‌سازی مورد توجه قرار دارد.

متدولوژی **EUP** فرایند سنگین دارد که در آن مدل‌سازی، مستندسازی و فعالیت‌های مدیریتی اعم از مدیریت سازمانی سنگین است. این متدولوژی برای پروژه‌های با پیچیدگی و اندازه بالا که خصوصا دغدغه‌های سازمانی نیز مورد توجه باشند و نیاز به روش ساختاریافته باشد، مناسب است و می‌توان انواع سیستم‌های اطلاعاتی را برای آن، مورد هدف قرار داد. اگرچه با استفاده از OCL در UML پشتیبانی تاحدی از فرمالیزم وجود دارد ولی برای پروژه‌های با بالاترین سطوح بحرانیت مناسب نیست و بهتر است از روش‌های فرمال استفاده کرد.

مقایسه: علاوه بر تفاوت‌های ذکر شده از هردو متدولوژی، موردی که هردو در آن ضعف دارند، فاز نگهداری و مراقبت از سیستم است که فاز بسیار مهمی از چرخه‌حات عمومی است. اگرچه EUP به دلیل تعریف فاز و نظام بخصوص تا حدی از ASD بهتر است ولی به دلیل این که حتی کوچک‌ترین تغییر مانند رفع یک اشکال نیازمند شروع مجدد چرخه در EUP است، برای مراقبت و نگهداری کافی نیست.

۷-۲ زبان مدل‌سازی

در این بخش، متدولوژی‌ها از منظر زبان مدل‌سازی مورد ارزیابی قرار می‌گیرند.

۷-۲-۱ پشتیبانی از مدل‌سازی شی‌گرای سازگار، صحیح و دقیق و بدون ابهام

متدولوژی **ASD**، مدل‌سازی سنگین، ساختار یافته و استفاده از روش‌ها یا ابزارهای خاص را تجویز نمی‌کند و مدل‌سازی مفصل از ابتدا ندارد و بیشتر بر تطبیق‌پذیری و کاوش مبتنی بر مشارکت و همکاری در طی چرخه‌ها تاکید دارد. برای همین منظور، در فاز آغاز پروژه و فعالیت ایجاد مصنوعات ماموریت، در مشخصات محصول یا طرح‌کلی، نتایج تحلیل و مدل‌سازی سبک‌وزن اولیه که شامل وظیفه‌مندی کلی، کلاس‌های اصلی و

تعاملات مرتبط است، نمایش داده می‌شود و این کار حداقلی بوده و به جهت شروع چرخه ایجاد نرم‌افزار انجام می‌شود. در طی چرخه‌ها بسته به نیاز، این مدل‌ها از نظر عمق و وسعت غنی‌تر می‌شوند. بنابراین پشتیبانی از دیدگاه‌های ساختاری[۲۷]، وظیفه‌ای[۲۸] و رفتاری[۲۹] از این طریق صورت می‌گیرد. مولفه‌های محصول شامل مولفه‌های ویژگی[۳۰]، فناوری و پشتیبان، از طریق جلسات JAD تعریف می‌شوند که در نهایت سیستم نهایی را تشکیل می‌دهند. در این متدولوژی مدل‌سازی از منطقی تا فیزیکی یعنی قلمرو کسب و کار تا قلمرو جواب، پشتیبانی زیادی ندارد و فقط مقداری مدل‌سازی در ابتدای فرایند و شناخت محصول انجام شده و سپس چرخه ایجاد آغاز می‌شود و مابقی طی چرخه‌ها و جلسات مشارکتی انجام می‌شود. همچنین پشتیبانی صریح و ساختار یافته‌ای از سطوح درشت‌دانگی و تجرید مختلف مانند سطح سازمان و سیستم تا سطح شی وجود ندارد. امکاناتی برای پشتیبانی فرمالیزم نیز ارائه نشده است.

متدولوژی **EUP** در این معیار خوب عمل می‌کند و ذاتا مدل‌سازی در آن سنگین است. در این متدولوژی برخلاف RUP که فقط استفاده از UML مجاز بود، استفاده از روش‌های دیگر مانند DFD یا BPMN برای مدل‌سازی کسب و کار نیز مجاز است. دیدگاه‌های مختلف ساختاری، وظیفه‌ای و رفتاری در زبان غنی UML از طریق مدل‌هایی نظیر نمودار کلاس، نمودار پکیج، نمودار فعالیت، نمودار توالی، use case و غیره کاملا پشتیبانی می‌شوند. در این متدولوژی مدل‌سازی از منطقی تا فیزیکی یعنی قلمرو مسئله و کسب و کار تا قلمرو جواب، طی فازهای مختلف کاملا پشتیبانی می‌شود و مدل‌های مختلف در هر سطح ایجاد می‌شوند و سیر کاملی از تحلیل تا پیاده‌سازی طی می‌شود. این متدولوژی در پشتیبانی از سطوح مختلف تجرید و درشت‌دانگی نیز به خوبی عمل می‌کند و خصوصا در EUP که موارد مربوط به سازمان نیز اضافه شده است، از سطح سازمان تا سطح شی مدل‌سازی به خوبی انجام می‌شود. علی‌رغم وجود OCL در UML که تا حدی امکانات فرمالیزم ارائه می‌کند، برای پروژه‌های با بالاترین سطح بحرانیت کافی نبوده و اساسا بهتر است از روش‌های فرمال استفاده کرد.

مقایسه: مشخصا بنابر توضیحات، EUP در این معیار نسبت به ASD بهتر عمل می‌کند. البته یک نکته حائز اهمیت در EUP این است که به دلیل تعداد زیاد مدل‌ها که گاها هم‌پوشانی داشته و هم‌ریخت[۳۱] یکدگیر می‌شوند، می‌تواند باعث ایجاد ناسازگاری شود که باید به این نکته توجه نمود.

---

[۲۷] structural
[۲۸] functional
[۲۹] behavioral
[۳۰] feature
[۳۱] isomorph

۷-۲-۲ ارائه راهبردها و تکنیک‌هایی برای رفع ناسازگاری‌ها و مدیریت پیچیدگی

متدولوژی **ASD** به طور خاص زبان‌های مدل‌سازی رسمی مانند UML را تجویز نمی‌کند. بنابراین، امکان استفاده از معانی رسمی[۳۲] برای تعریف وابستگی‌ها و محدودیت‌های مدل و چک کردن ناسازگاری‌ها وجود ندارد. در کل ASD برای رفع ناسازگاری‌ها به جای استفاده از ابزارهای رسمی، بر تعامل تیمی، بازخورد مستمر، و ارزیابی‌های دوره‌ای در فاز همکاری و یادگیری تکیه می‌کند. این متدولوژی سازوکارهایی برای مدیریت پیچیدگی مدل و لایه‌بندی، تجزیه و کاهش پیچیدگی سیستم‌های بزرگ ارائه نمی‌کند و اساسا بر ساده‌سازی طراحی و ارتباطات تیمی تکیه دارد. هیچ‌گونه ساختار بسته[۳۳] یا مولفه[۳۴] برای تجزیه سیستم به زیرسیستم‌ها ارائه نمی‌کند. تمرکز آن، مدل‌های ذهنی مشترک، طرح‌ها و اسناد ساده و همچنین تکامل تدریجی است.

در متدولوژی **EUP** مدل‌سازی بسیار جدی است و تعداد بسیار زیادی مدل در سطوح مختلف ایجاد می‌شوند. در EUP علاوه بر استفاده از UML، از ابزارهای دیگر مانند نمودار جریان داده[۳۵] نیز استفاده می‌شود و در کل حفظ سازگاری بین مدل‌ها بسیار سخت و البته مهم است. برای حفظ سازگاری در سطح مدل‌های UML استفاده از ابزار می‌تواند بسیار مفید باشد و ابزارهایی مانند Enterprise Architect، امکاناتی برای بررسی سازگاری بین برخی مدل‌ها ارائه می‌کند که این کار را با بررسی برخی معانی تعریف شده در UML انجام می‌دهد مانند بررسی ارتباطات میان کلاس‌ها، مولفه‌ها و بسته‌ها. همچنین امکاناتی برای بررسی سازگاری برخی مدل‌های رفتاری مانند نمودارهای توالی با مدل‌های ساختاری مانند نمودار کلاس ارائه می‌کند. البته درصورت استفاده نکردن از ابزار، حفظ سازگاری بین تعداد زیاد مدل که در این متدولوژی ایجاد می‌شود بسیار سخت است خصوصا که معانی UML در طول زمان برای گسترش حوزه کاربرد آن سبک‌تر شد ولی جا برای ناسازگاری افزایش یافت. همچنین در UML امکاناتی برای لایه‌بندی و جزءبندی ارائه شده است از طریق ایجاد نمودارهای بسته در مدل‌های تحلیل و نمودارهای مولفه در مدل‌های طراحی که با کمک آن‌ها سیستم به لایه‌های متعدد و قسمت‌های مجزا تقسیم و در نهایت به کلاس‌ها می‌رسد و از این طریق مدیریت پیچیدگی مدل حاصل می‌شود. در مدل‌های DFD نیز لایه‌بندی دیده می‌شود. این‌گونه که در سطح صفر یک نمودار داریم که یک پردازه در وسط دارد و نماینده سیستم است و به آن نمودار زمینه[۳۶] گویند. در سطح بعدی پردازه‌های درشت‌دانه سیستم موجودند و به همین شکل به سمت پایین و ریزدانه شکسته می‌شود.

مقایسه: مشخصا بنابر توضیحات، EUP پشتیبانی بهتری از مدل‌سازی و مدیریت پیچیدگی ارائه می‌کند. البته تعدد زیاد مدل و وابستگی آن‌ها در EUP می‌تواند مشکل‌ساز باشد که باید به تجویزات متدولوژی در این مورد توجه کرد و همچنین استفاده از ابزار مناسب نیز کمک شایانی می‌کند.

---

[۳۲] semantic
[۳۳] package
[۳۴] component
[۳۵] data flow diagram
[۳۶] context diagram

فصل ۸

نتیجه‌گیری

در طی انجام این تمرین، آشنایی کلی با متدولوژی‌های BON، USDP، ASD و EUP حاصل شد و این متدولوژی‌ها با دیدگاهی مبتنی بر معیار در دو قسمت فرایند و زبان مدل‌سازی مورد ارزیابی و تحلیل قرار گرفتند. دید تحلیلی مبتنی بر معیارهای فهرست شده، کمک شایانی در مقایسه و انتخاب متدولوژی برای به کار گیری در یک پروژه مهندسی نرم‌افزار می‌کند.

Bibliography

[1] R. Ramsin and R. Paige. Process-centered review of object oriented software development methodologies. *ACM Comput. Surv*, 40(1):23–26, 2008.

[2] R. Ramsin and R. Paige. Process-centred review of object oriented software development methodologies. pages 126–131, 2004.

[3] J.-M. Nerson. Applying object-oriented analysis and design. 1992.

[4] R. Ramsin. Software development methodologies, 2024. Lecture slides, Sharif University of Technology, Tehran, Iran.

[5] R. Ramsin and R. Paige. Iterative criteria based approach to engineering the requirements of software development methodologies. *IET Software*, 4(2):91–104, 2010.

Abstract

In this study, the BON and USDP methodologies were introduced and compared in one section, while the ASD and EUP methodologies were addressed in another. For this comparison, 11 criteria for the process and 2 criteria for the modeling language were introduced. First, the BON and USDP methodologies were evaluated based on these criteria, followed by a similar analysis for the ASD and EUP methodologies. This analysis highlights the characteristics, strengths, and weaknesses of each methodology, aiding in their selection for developing software systems.

Keywords: methodology evaluation, process criteria, modeling language criteria

Sharif University of Technology
Department of Computer Engineering

Software Development Methodologies

First Assignment

By:

Mehdi Eidi

Professor:

Dr. Raman Ramsin

Fall 2024