بسمه تعالی

درس مهندسی نرم افزار ۱

نیمسال اول ۹۳-۹۲

مرضیه سادات طباطبایی

فصل سوم

توسعه چابک

نگاهی گذرا

- توسعه چابک چیست؟ تلفیقی ازفلسفه و مجموعه ای از دستورالعمل های توسعه
- فلسفه:
  - جلب رضایت مشتری
  - تحویل افزایشی نرم افزار از ابتدای پروژه
  - تیم های پروژه کوچک با انگیزه بالا
  - روش های غیر رسمی
  - حداقل محصولات کاری مهندسی نرم افزار
  - سادگی کلی در توسعه نرم افزار
- دستورالعمل توسعه
  - تحویل نرم افزار بر اساس تحلیل و طراحی
  - برقراری ارتباط فعال و پیوسته میان توسعه دهندگان و مشتریان

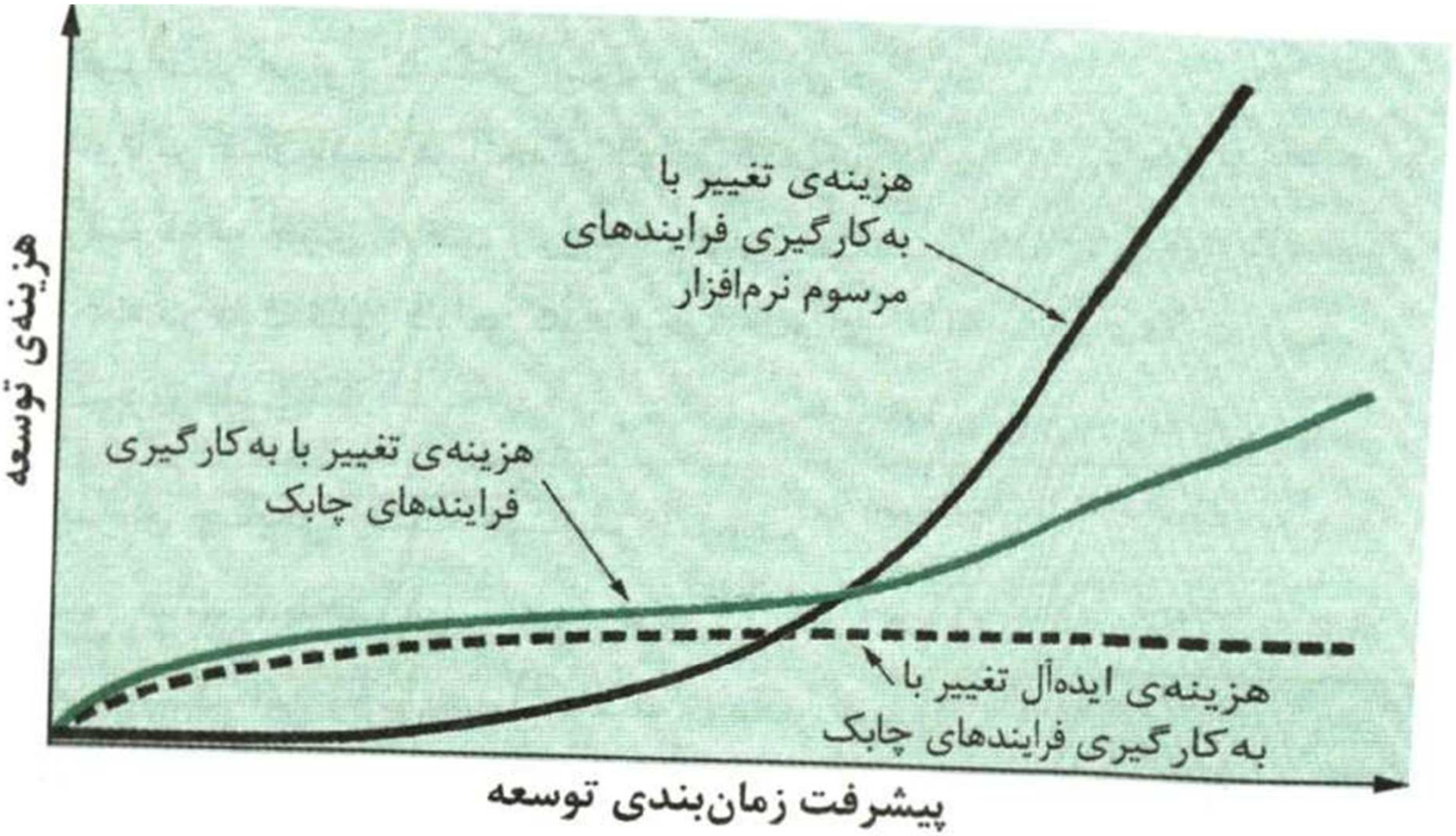
نگاهی گذرا

- چه کسی آن را انجام می دهد؟
  - مهندسان نرم افزار و سایر ذی نفعان پروژه (مدیران، مشتریان و کاربران نهایی) یک تیم چابک خود سازمان ده را تشکیل می دهند.
- چرا اهمیت دارد؟
  - محصولات کامپیوتری با سرعت در حال پیشرفت و تغییر
- چه مراحلی دارد؟
  - فعالیت های چارچوبی پایه (ارتباطات، برنامه ریزی، مدل سازی، ساخت و استقرار) ولی حداقلی و کمینه
  - محصول کار: یک نرم افزار عملیاتی که به موقع تحویل مشتری شود.

توسعه چابک

- برای همه پروژه ها، محصولات، افراد و شرایط قابل استفاده نیست.
- یکی از ویژگیهای بارز این روش توانایی آن در کاهش دادن هزینه های ناشی از تغییر در سراسر فرآیند است.
- تیم چابک تیمی فرز و چالاک که قادر به پاسخگویی مناسب به تغییرات است:
  - تغییرات در نرم افزارهای در حال ساخت
  - تغییرات در اعضای تیم
  - تغییرات به دلیل فن آوری جدید
- هسته اصلی موفقیت پروژه مهارتهای افراد تیم و توانایی آنها در همکاری با یکدیگر
- تاکید بر تحویل افزایشی و سریع نرم افزارهای عملیاتی
- مشتری بخشی از تیم توسعه

چابکی و هزینه های تغییر



شکل ۱-۳ هزینه‌ی تغییرات به‌عنوان تابعی از زمان در توسعه‌ی نرم‌افزار.

چابکی و هزینه های تغییر

- هزینه تغییر به صورت غیر خطی با پیشرفت پروژه افزایش می یابد.
- یک فرآیند چابک با طراحی خوب، منحنی هزینه تغییر را تسطیح می کند. تحویل افزایشی، آزمون واحدهای پیوسته و برنامه نویسی جفتی هزینه اعمال تغییرات را کاهش می دهد.

فرآیند چابک

- فرآیند چابک باید قادر به مدیریت موارد غیر قابل پیش بینی باشد پس فرآیند باید انطباق پذیری داشته باشد. برای دستیابی به انطباق پذیری تدریجی، تیم باید از مشتری بازخورد داشته باشد بنابراین نسخه های نرم افزار باید در بازه های کوتاه مدت تحویل شوند تا روند انطباق همگام با روند تغییرات ادامه یابد.
- فرآیندهای چابک همگی به دوازده اصل کلی پایبند هستند.

دوازده اصل چابکی

- بیشترین الویت رضایت مشتری از طریق تحویل زودهنگام و پیوسته نرم افزارهای افزایشی
- پذیرا بودن تغییرات در خواسته ها حتی در اواخر فرآیند توسعه
- تحویل پیوسته نرم افزارهای کاری از دو هفته تا دو ماه
- دست اندر کاران و افراد تجاری باید در سرتاسر پروژه هر روز با هم کار کنند.
- سپردن پروژه به افراد با انگیزه، پشتیبانی آنها و اطمینان به آنها
- گفتگوی رو در رو، بهترین راه انتقال اطلاعات به درون و بیرون تیم

دوازده اصل چابکی

- میزان اصلی در سنجش پیشرفت، نرم افزار کاری است.
- حامیان، سازندگان و کاربران باید قادر به حفظ سرعت ثابت باشند.(ارتقای توسعه پایدار)
- توجه پیوسته به اعتلای فنی و طراحی خوب
- سادگی
- تیم خود سازماندهی شده
- بازخورد از میزان بهبود اثربخشی تیم در بازه های منظم

عوامل انسانی

- توسعه چابک بر استعدادها و مهارت های افراد و شکل دهی به فرایند بر اساس افراد و تیم های موجود تاکید دارد. فرآیند باید بر اساس نیازهای افراد و تیمها شکل یابد نه برعکس
- خصوصیات کلیدی افراد تیم چابک:
  - رقابت
  - کانون توجه مشترک : توجه به هدف مشترک «تحویل به موقع محصول»
  - همکاری
  - توانایی تصمیم گیری
  - توانایی حل مساله با منطق فازی
  - احترام و اطمینان متقابل
  - خودسازماندهی

برنامه نویسی حدی (XP)

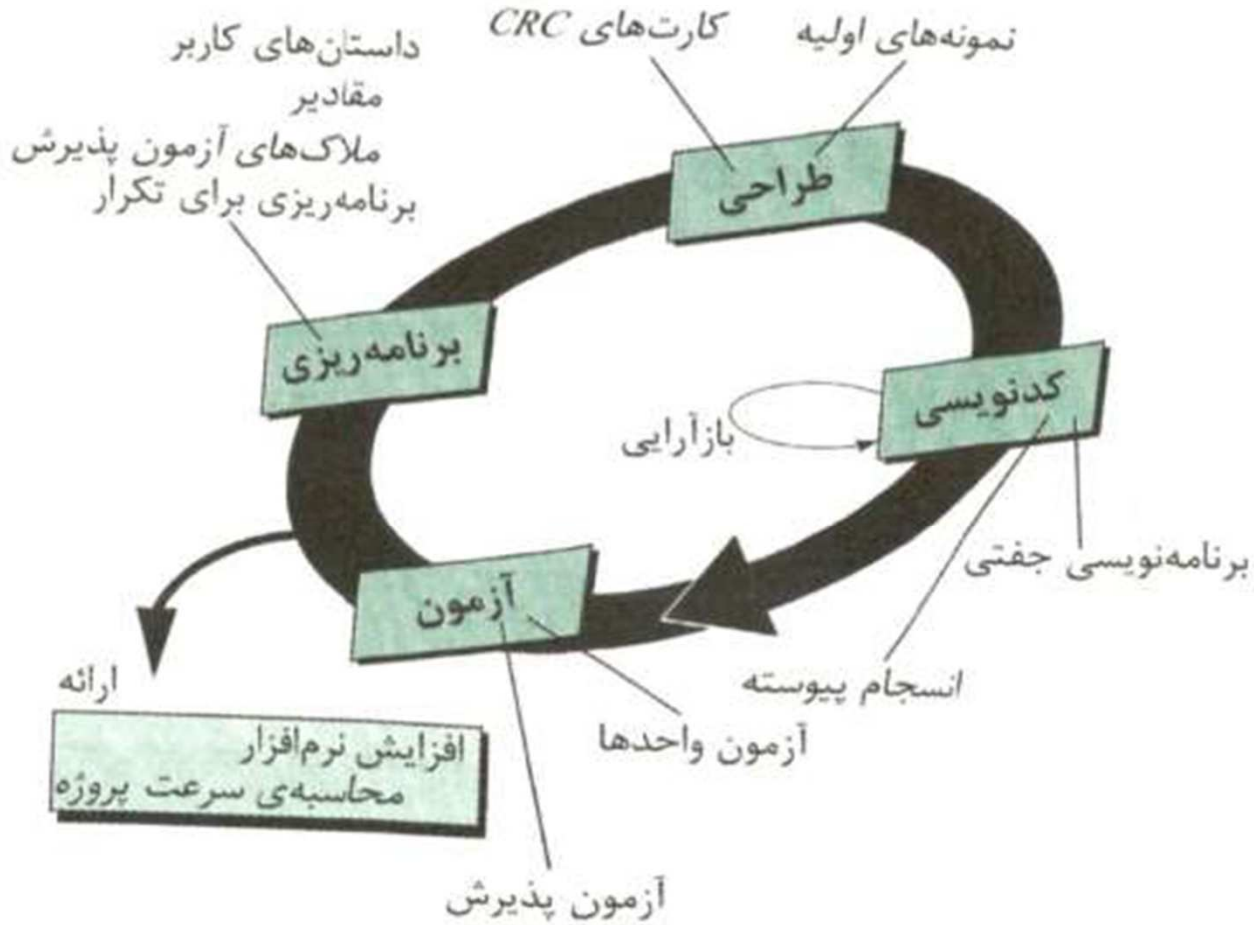
- پر کاربردترین رویکرد در توسعه نرم افزار به روش چابک است.
- IXP: XP صنعتی که پالایشی از XP است برای استفاده در سازمان های بزرگ
- یک مجموعه ای از ۵ ارزش را تعریف می کند که مبنای تمامی کارهای انجام شده در XP است.

ارزشهای XP

- ارتباطات :
  - همکاری نزدیک و در عین حال غیر رسمی میان مشتریان و سازندگان
  - برقراری استعاره های اثربخش (استعاره داستانی است که توسط مشتری،برنامه نویس یا مدیر درباره چگونگی کارکردن سیستم روایت میشود)
  - بازخورد پیوسته و پرهیز از مستندات پرحجم به عنوان واسطه ارتباط
- سادگی
  - طراحی تنها برای نیازهای فوری و نه برای نیازهای آینده
  - هدف ایجاد یک طراحی ساده که به آسانی در قالب کدنویسی پیاده شود.
- بازخورد
  - از سه منبع نرم افزار، مشتری، سایر اعضای تیم نرم افزاری
  - سناریوها و use case ها مبنایی برای آزمون پذیرش
- جرات
  - جرات لازم را در مورد طراحی برای امروز را داشته باشیم.
- احترام
  - احترام برای فرآیند XP با دنبال کردن این ارزش ها

فرآیند XP

- استفاده از یک روش شی گرا به عنوان الگوی توسعه



فرایند برنامه‌نویسی حدی.

فعالیتهای چارچوبی فرآیند XP

- برنامه ریزی
  - نوشتن داستانها روی یک کارت شاخص و دادن یک الویت به آن توسط مشتری
  - ارزیابی داستانها توسط تیم XP و دادن یک هزینه(بر حسب تعداد هفته های لازم برای توسعه) به هر یک
  - تقسیم داستانها با ارزش بیش از سه هفته به داستانهای کوچکتر توسط مشتری
  - نوشتن داستان جدید در هر زمان امکانپذیر است.
  - پیاده سازی داستانها به یکی از سه روش زیر
    - پیاده سازی همه داستانها بلافاصله
    - پیاده سازی داستانهای با بیشترین ارزش
    - پیاده سازی داستانها با بیشترین ریسک
  - محاسبه سرعت پروژه پس از ارائه نسخه نخست
    - سرعت پروژه برابر با تعداد داستانهای پیاده سازی شده در نسخه نخست است.
  - پرداختن به داستانهای باقیمانده

فعالیتهای چارچوبی فرآیند XP

- طراحی
  - پیروی از اصل سادگی KIS
  - نوشتن راهنمای پیاده سازی برای هر داستان به موازات نوشتن آن
  - کارتهای CRC تنها محصول طراحی
    - کارتهای CRC (کلاس-مسئولیت-همکار) کلاسهای شی گرایی مربوط به نسخه فعلی نرم افزار را شناسایی و سازماندهی می کنند.
  - استفاده از راهکار خیزشی
    - ایجاد فوری یک نمونه اولیه عملیاتی در صورت مشاهده مشکل در بخشی از طراحی یک داستان، به منظور پایین آوردن خطر در هنگام پیاده سازی واقعی
  - تاکید بر بازآرایی
    - بازآرایی عبارتست از تغییر دادن یک سیستم نرم افزاری طوریکه رفتار خارجی کد را تغییر ندهد و در عین حال ساختار داخلی را بهبود بخشد.
    - طراحی هم قبل و هم بعد از کد نویسی رخ می دهد.

فعالیتهای چارچوبی فرآیند XP

- کد نویسی
  - پس از اتمام کارهای طراحی مقدماتی، تیم به کد نویسی نمی پردازد بلکه یکسری «آزمون واحد» (فصل ۱۷) تهیه می کند که هر یک از داستانهای نسخه فعلی را مورد آزمون قرار دهد.
  - از طریق آزمون واحد سازنده بهتر می تواند پیاده سازی کند تا آزمون مورد نظر موفقیت آمیز باشد.
  - برنامه نویسی جفتی
    - حل مسئله به صورت بی درنگ
    - تضمین کیفیت بی درنگ
  - انسجام بخشی پیوسته
- آزمون
  - آزمون واحدها
  - آزمون انسجام و اعتبارسنجی
  - آزمونهای پذیرش(آزمونهای مشتری)

IXP (XP صنعتی)

- تفاوت عمده در اعمال مدیریت بیشتر، گسترش نقش مشتریان و ارتقای روشهای فنی است. همچنین شامل ۶ عمل جدید:
  - ارزیابی آمادگی
  - جامعه پروژه
  - چارتر کردن پروژه
  - مدیریت مبتنی بر آزمون
  - بازنگری
  - آموزش پیوسته

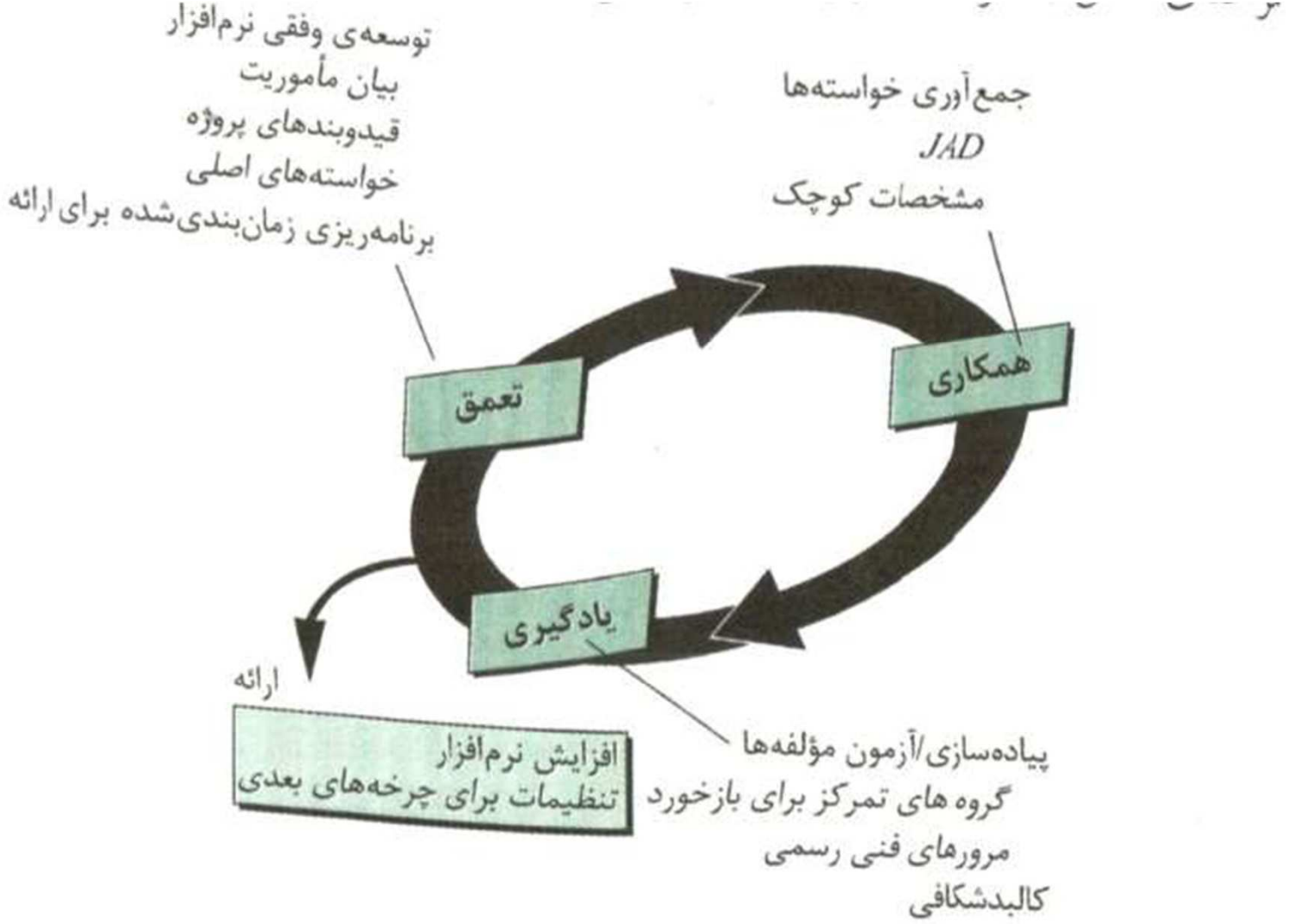
انتقادات به XP

- متغیر بودن خواسته ها، زیرا مشتری عضو فعالی از تیم XP است لذا تغییراتی که در خواسته ها به عمل می آید به صورت غیر رسمی تقاضا می شود.
- نیازهای متناقض مشتریان در پروژه های با چند مشتری
- خواسته ها به صورت غیر رسمی بیان می شوند.
- فقدان طراحی رسمی

سایر مدلهای فرآیند چابک

- توسعه وفقی نرم افزار ASD
- اسکرام scrum
- روش توسعه سیستمهای پویا DSDM
- کریستال
- توسعه ویژگی محور FDD
- توسعه نرم افزار ناب LSD
- مدلسازی چابک AM
- فرآیند یکپارچه چابک AUP

توسعه وفقی نرم افزار ASD



شکل ۳-۳ توسعه‌ی وفقی نرم‌افزار

توسعه وفقی نرم افزار ASD

- تکنیکی برای ساخت نرم افزارها و سیستم های پیچیده
- فلسفه: همکاری انسانی و خود سازماندهی تیمی
- چرخه حیات برای ASD شامل سه مرحله
  - تعمق
    - آغاز پروژه و برنامه ریزی و تعیین خواسته های پایه
  - همکاری
    - برقراری ارتباط و کار تیمی
    - خلاقیت فردی
    - اعتماد
  - یادگیری

اسکرام

- فعالیت های چارچوبی:خواسته ها،تحلیل، طراحی، تکامل، تحویل
- فهرست جامانده ها backlogs
  - فهرستی الویت بندی شده از خواسته های پروژه
- Sprint ها
  - شامل واحدهای کاری مورد نیاز برای دستیابی به خواسته ای تعیین شده در فهرست جامانده ها
  - این واحدها در یک کادر زمانی معین می گنجند.
  - تغییرات در طول sprint وارد نمی شوند
- نشستهای اسکرام : نشستهای کوتاه(۱۵ دقیقه) روزانه شامل سه پرسش
  - از آخرین جلسه چه کردید؟
  - با چه موانعی مواجه شدید؟
  - در جلسه بعدی چه چیز برای ارائه دارید؟
- دموها
  - نسخه نرم افزاری تحویل شده به مشتری
  - شامل عملکردهای مربوط به کادر زمانی مورد نظر

روش توسعه سیستمهای پویا DSDM

- فلسفه آن مبتنی بر اصل پارتو(۸۰٪ از یک برنامه کاربردی را می توان در ۲۰٪ از زمان لازم برای تحویل برنامه کاربردی تحویل داد)
- یک فرآیند تکراری که در هر نسخه می توان جزئیات را بعدا کامل کرد.
- ابتدا دو فعالیت چرخه حیاتی
  - امکان سنجی : این نرم افزار کاندیدای مناسبی برای DSDM است؟
  - مطالعه تجاری : تعیین خواسته های عملیاتی و اطلاعاتی ارزش آفرین برای نرم افزار
- سپس سه چرخه تکرار متفاوت:
  - تکرار مدلهای عملیاتی : مجموعه ای از نمونه های اولیه
  - تکرار طراحی و ساخت : بازبینی نمونه های اولیه ساخته شده در مرحله قبل
  - پیاده سازی : استقرار آخرین نسخه نرم افزار در محیط کاری
  - تکرار این سه مرحله

کریستال

- در واقع مجموعه مثالهایی از فرآیندهای چابک است که برای انواع پروژه های متفاوت موثر واقع شده اند. مقصود این است که تیمهای چابک بتوانند عضوی از این مجموعه را انتخاب کنند که بیشترین مناسبت را با پروژه و محیط آنها داشته باشد.

توسعه ویژگی-محور FDD

- فرآیند عملی برای مهندسی نرم افزار شی گرا
- مناسب برای پروژه های با ابعاد متوسط و بزرگ
- ویژگی
  - یک عملکرد است که نزد متقاضی دارای ارزش بوده و در کمتر از دو هفته قابل پیاده سازی است.
- مزایای تعریف ویژگی
  - چون ویژگیها قطعات کوچک از قابلیتهای قابل تحویل هستند، توصیف، بازبینی و رفع ابهام و خطای آنها راحت تر
  - سازماندهی آنها به صورت سلسله مراتبی و ارتباط آنها با یکدیگر
  - هر ویژگی یک نسخه قابل تحویل که هر دو هفته یک بار توسعه می یابد.
- قالب یک ویژگی
- <action>the<result><by|for|to>a(an)<object>
- قالب یک گروه ویژگی
- <action><ing>a(n)<object>

توسعه ویژگی-محور FDD

- مثال : گروه ویژگی «به فروش رساندن محصول» که شامل ویژگیهای زیر می شود:
  - افزودن محصول به سبد خرید
  - نمایش مشخصات فنی محصول
  - نگهداری اطلاعات حمل برای مشتری
- ۵ فعالیت چارچوبی
  - توسعه یک مدل کلی
  - تهیه فهرستی از ویژگیها
  - برنامه ریزی بر اساس ویژگیها
  - طراحی بر اساس ویژگیها
  - ساخت بر اساس ویژگیها

توسعه ویژگی-محور FDD

- بیش از هر روش چابک دیگر بر روشهای مدیریتی تاکید دارد
- به همین منظور ۶ نقطه عطف در طول پروژه تعریف می کند:
  - بررسی در طراحی
  - طراحی
  - بازرسی طراحی
  - کد نویسی
  - بازرسی کد
  - ارتقا به ساخت