

عنوان: 200 بلاگ برتر Agile دنیا  
نویسنده: وحید نصیری  
تاریخ: ۱۳۹۰/۰۳/۱۸ ۲۲:۲۸:۰۰  
آدرس: [www.dotnettips.info](http://www.dotnettips.info)  
برچسب‌ها: Agile

لیستی در سایت زیر در مورد 200 بلاگ برتر Agile دنیا منتشر شده (طبقه بندی شده بر اساس تعداد خواننده و رتبه بندی‌های متداول):

[Top 200 Agile Blogs RSS + Twitter Lists](#)

که طبق معمول یک چیزی کم دارد و آن هم فایل [OPML](#) مربوطه است که آن را [از اینجا](#) می‌توانید دریافت کنید.

داخل وبلاگها و وب سایتهای فارسی زبان(مربوط به برنامه نویسی) که جستجو میکنیم شاهد کلمه ای هستیم که تازه به چشمان میخورد:Agile(چابک). البته لازم به ذکر است این کلمه چیز جدیدی نیست و سابقه ای در حدود 10 سال دارد ( February 2001 ).که جمعی از برنامه نویسان بیانیه ای را تحت عنوان چابک ( Agile ) تهیه کردند که متن آن به شرح زیر است:

ما با توسعه نرم افزار و کمک به دیگران در انجام آن . در حال کشف راههای بهتری برای توسعه نرم افزار هستیم. از این طریق باید دست یابیم به ارزش :

افراد و تعاملات بالاتر از فرآیندها و ابزارها

نرم افزار کارکننده بالاتر از مستندات جامع

مشارکت مشتری در انجام کار بالاتر از قرارداد کار

پاسخگویی به تغییرات بالاتر از پیروی یک طرح

با وجود اینکه موارد سمت چپ نیز ارزشمند هستند ولی ما برای موارد سمت راست ارزش بیشتری قائل هستیم .

که این بیانیه بر پایه 12 اصل(A gile principles ) بالاترین اولویت ما جلب رضایت مشتری با تحویل زود و مداوم نرم افزاری ارزشمند می باشد

استقبال از تغییر نیازمندی ها، حتی در اواخر فرآیند توسعه. فرآیندهای چابک، تغییر را در جهت مزیت رقابتی مشتری مهار میکنند

تحویل زود به زود نرم افزار قابل استفاده دو، سه هفته یک بار تا دو ، سه ماه یک بار با ترجیح بر فاصله های زمانی کوتاه تر

ذی نفعان کسب و کار و توسعه دهنده ها می بایست به صورت روزانه در طول پروژه با هم کار کنند

پروژه ها را بر دوش افراد با انگیزه بنا کنید. فضای لازم رابه آنها بدهید و از نیازهای آنها پشتیبانی کنید وبه آنها اعتماد کنید تا کارها را انجام دهند

کارآمدترین و موثرترین روش انتقال اطلاعات به تیم توسعه و تبادل آن در میان اعضای تیم ، گفتگوی چهره به چهره است

نرم افزار قابل استفاده اصلی ترین معیار سنجش پیشرفت است

فرآیندهای چابک توسعه پایدار را ترویج می دهند حامیان مالی ، توسعه دهندگان و کاربران باید بتوانند سرعت پیشرفت ثابتی را برای مدت نامحدودی حفظ کنند

توجه مداوم به برتری فنی و طراحی خوب باعث افزایش چابکی می شود

سادگی -- هنر به حداکثر رساندن مقدار کار انجام نشده -- ضروری است

بهترین معماری‌ها، نیازمندی‌ها و طراحی‌ها از تیم‌های خود سازمانده پدید آور می‌شود

در فواصل منظم، تیم برچگونگی موثرتر شدن تامل و تفکر می‌نماید و سپس تیم رفتار خود را بر اساس بازتاب این تفکر تنظیم و هم سو می‌نماید

متأسفانه در ایران حالا یا به علت سواد کم و یا به هر علتی از این بیانیه برداشتهایی متفاوت و غلط عده ای اونیو به بازی تشبیه کردن و عده ای هم با اون کار میکنن ولی هیچکدوم از اصل‌های اونیو رعایت نمیکنن و بعد که پروژه شکست خورد میگن:متدولوژی خوب نبود و ....

وقتی کتاب `#Agile Principles, Patterns, and Practices in C` رو مطالعه میکنید به این نتیجه میرسید که بیشترین چیزی که تاکید داره روی ارتباطات هستش.

قصد دارم در قالب چند پست به شما این اصول رو معرفی کنم.

## نظرات خوانندگان

نویسنده: bizhan

تاریخ: ۱۳۹۱/۰۴/۰۵ ۱۲:۴۹

سلام خسته نباشید  
تا همین حد هم که معرفی کردین خیلی خوبه  
و خیلی لطف میکنن اگر ادامه بدین و منابع بیشتری در مورد این بحث معرفی کنید ( از منابع ساده و کلی تا منابع پیشرفته تر و جزئی)

نویسنده: شهروز جعفری

تاریخ: ۱۳۹۱/۰۴/۰۶ ۱:۳۱

سلام .  
راستیتش من میخوام این بحثو گسترش بدم ولی دیدم یک وبلاگ خوب و چند تا کتاب فارسی خوب تواین زمینه هستش که بنظرم اگه بخوام بیشتر توضیح بدم درست کردن چرخ از اوله.  
این وبلاگ کاملی هستش: [Irscrum](http://Irscrum)

نویسنده: اژدری

تاریخ: ۱۳۹۱/۰۶/۱۳ ۱۳:۱۴

ممنون از لینک مفیدتون

نویسنده: م.رضا لایقی

تاریخ: ۱۳۹۱/۱۲/۲۲ ۱۵:۰۰

لینکی که معرفی کردین ظاهرا دیگه کار نمی‌کنه.....

نویسنده: بتیسا

تاریخ: ۱۳۹۱/۱۲/۲۳ ۸:۲۴

آدرس موسسه به [scrum.ir](http://scrum.ir) تغییر کرده

شاید شما هم مثل من فکر می‌کنید، به اندازه‌ای که در حرفه یا زندگی شخصی خود زحمت می‌کشید، نتیجه نمی‌گیرید! چندی پیش کتابی خواندم از آقای J.D Meier (مهندس نرم افزار و مدیر پروژه در شرکت مایکروسافت) که نحوه‌ی برنامه ریزی و زمان بندی من در کار و زندگی ام را به طور شگفت آوری متحول کرد. به همین دلیل به این فکر افتادم که در قالب چند مقاله به معرفی روش ایشان که یک روش بسیار آسان برای گرفتن نتایج بهتر و سریع‌تر در کار و زندگی است، بپردازم. امیدوارم همانطور که من این روش را مفید و کاربردی یافتم، برای خوانندگان این مقاله هم تأثیرگذار باشد.

این مدل که روش چابک (Agile way) نام دارد برپایه چهار اصل اساسی به نام‌های **قانون سه تایی، دیدگاه شنبه، خروجی روزانه و بازخورد پنجشنبه** شکل گرفته است که در ادامه به توضیح هریک از این اصول پرداخته شده است.

### 1) قانون سه تایی:

این قانون یک قانون ساده است و به شما کمک میکند تا در زمانی که خیلی پرمشغله هستید، با یک روش ساده بتوانید به کارهای خود سروسامان دهید و در زمان کوتاه به بهترین نتایج دلخواه‌تان دست یابید.

قانون سه تایی یعنی به صورت سه تایی فکر کردن. به این معنی که شما سه نتیجه‌ای را که مایل هستید در انتهای یک روز، یک هفته یا یک ماه بدست آورید را مشخص می‌کنید. با این سیستم سه در سه (3 x 3) شما نقشه آنچه را که می‌خواهید به دست آورید، به صورت خیلی ساده و سریع در مجموعه‌های سه تایی طراحی می‌کنید که به راحتی قابل بخاطر سپردن و ویرایش سریع است.

داشتن یک برنامه‌ی هفتگی، قلب مدل "روش چابک" است. الگوی برنامه‌ی هفتگی در این روش براساس دیدگاه شنبه، خروجی روزانه، بازخورد پنجشنبه و با کمک قانون سه تایی بنا شده است. جدول زیر نحوه طراحی و برنامه ریزی در این روش را نشان می‌دهد.

| نقاط چرخش   | دیدگاه شنبه           | خروجی روزانه      |       |       |       |       |       | بازخورد پنجشنبه     |           |
|-------------|-----------------------|-------------------|-------|-------|-------|-------|-------|---------------------|-----------|
|             |                       | شنبه              | ۱شنبه | ۲شنبه | ۳شنبه | ۴شنبه | ۵شنبه | با موفقیت انجام شده | اصلاح شود |
| زندگی       | سه دست آورد برای هفته |                   |       |       |       |       |       |                     |           |
| کار         | ۱.                    | ۱.                | ۱.    | ۱.    | ۱.    | ۱.    | ۱.    | ۱.                  | ۱.        |
| مسائل شخصی  | ۲.                    | ۲.                | ۲.    | ۲.    | ۲.    | ۲.    | ۲.    | ۲.                  | ۲.        |
|             | ۳.                    | ۳.                | ۳.    | ۳.    | ۳.    | ۳.    | ۳.    | ۳.                  | ۳.        |
| برنامه ریزی |                       | انجام دادن برنامه |       |       |       |       |       | مرور کردن برنامه    |           |

این مدل به شما این اجازه را می‌دهد که در هر روز و هر هفته یک شروع جدید داشته باشید و بدین صورت از یک طرف یک برنامه ریزی دقیق، قابل اعتماد و از طرفی دیگر قابل انعطاف برای کار و زندگی روزمره خود داشته باشید.

### 2) دیدگاه شنبه

در این مدل در روز شنبه (اولین روز هفته) سه نتیجه‌ای را که مایل هستید در آخر هفته به دست بیاورید را مشخص می‌کنید. به طور مثال شما می‌خواهید طراحی اسلایدهای مربوط به جلسه‌ی هفته آینده را تا آخر هفته جاری به اتمام برسانید و یا مایل هستید که پس از یک هفته تمرین، به راحتی بتوانید دو مایل را در روز بدوید.

### (3) خروجی روزانه

در این مرحله، به صورت روزانه سه دستاوردی را که مایل هستید در انتهای یک روز داشته باشید را مشخص می‌کنید. به طور مثال در ارتباط با طراحی اسلایدهای جلسه آینده، شما می‌خواهید در پایان روز اسلایدهای مربوط به فاز اول پروژه را تکمیل کرده باشید و یا به طور مثال در ارتباط با رکورد دویدن، شما هدف هفتصد متر را برای خود در روز مشخص می‌کنید. به طور طبیعی خواهید دید که سه خروجی که برای هر روز خود در نظر می‌گیرید، باید در راستای سه دستاوردی باشند که برای هفته‌ی خود مشخص کرده‌اید.

### (4) بازتاب پنجشنبه

در روز پنجشنبه شما این امکان را دارید که یک قدم به عقب برگردید و نگاهی به برنامه ریزی هفته و دستاوردهای حاصل از آن بیاندازید و از خودتان بپرسید «کدام سه فعالیت هستند که باید بیشتر بر روی آن کار کنم؟» و «کدام سه فعالیت هستند که به خوبی پیش رفته اند؟». با این روش شما یاد می‌گیرید ظرفیتهای خود را شناسایی کرده و به صورتی شفاف مشخص کنید کدام فعالیتها نیاز به تمرکز، وقت و انرژی بیشتری دارند.

**نکته کلیدی:** در روش چابک، نکته‌ی کلیدی این است که موفقیت‌های خود را حتی اگر بسیار کوچک هم باشند، جشن بگیرید. به این دلیل که شما کار یا فعالیتی مشخص شده را خواه اینکه کوچک یا بزرگ باشد، به طور کامل انجام داده‌اید. توجه کنید که این به نوع نگاه شما بستگی دارد که بتوانید سه نتیجه‌ی مورد نظر خود در هفته و روز را بر اساس اولویت‌های خود و به صورت درست انتخاب کنید. بازتاب پنجشنبه به شما این امکان را می‌دهد که اولویت‌های خود را بازبینی و اصلاح کنید و از این بازخورد برای طراحی و برنامه ریزی هفته آینده و روزهای آتی استفاده کنید.

### نقاط جوش

در جدول برنامه ریزی مدل چابک ستونی به نام نقاط جوش ( hot spots ) گنجانده شده است. اگر مسایل مهم زندگی خود را در نظر بگیرید، احتمالاً آنها جزء یکی از مسایل مربوط به تفکرات و یا جسم شما، مسایل احساسی و یا مالی، معاشرت با دیگران و تفریحات خواهند بود. نقاط جوش زندگی در حقیقت بخش‌هایی از زندگی شما هستند که شما بر روی آنها تمرکز بیشتری دارید و به طور ساده آنها می‌توانند بیانگر درد و رنج فراوان، یا موفقیت‌ها و موقعیت‌های فراوان برای شما باشند. نکته‌ی مهم این است که این نقاط جوش هرچه باشند، بخش‌های مهمی در زندگی شما، از نظر شما هستند.

سعی کنید یک لیست از نقاط جوش زندگی خود تهیه کنید. مثلاً لذت بردن از زندگی، داشتن اندامی متناسب، به پایان رساندن یک مقطع تحصیلی، ارتقا یافتن در محل کار و یا داشتن یک رابطه خوب، می‌تواند مثالهایی از نقاط جوش زندگی باشند. مشخص کردن این نقاط جوش به شما کمک می‌کنند تا بتوانید بین کار، زندگی شخصی، مسایل مالی و سرگرمی، یک تعادل ایجاد کنید. به این معنی که به صورت متعادل و منصفانه، انرژی، زمان و تمرکز خود را صرف آنها کنید. به این نکته توجه کنید که وقتی به صورت متعادل بر روی هریک از نقاط جوش زندگی خود سرمایه‌گذاری می‌کنید، به طور ضمنی توانایی خود را برای رویارویی با اتفاقات جدید در تمام زمینه‌های یاد شده در بالا، افزایش می‌دهید. علاوه براین، اگر در یکی از زمینه‌ها مثلاً زندگی شخصی خود به اندازه‌ی کافی سرمایه‌گذاری نکنید، تاثیرات منفی آن را در محیط کار و حرفه‌ی خود مشاهده خواهید کرد.

### خلاصه‌ی مطلب

1) قانون سه تایی به عنوان یک روش ساده برای جلوگیری از بی‌نظمی و هرج و مرج و به سرانجام رساندن کارهای موردنظر با کیفیت بالا در مدت زمان مناسب به کار می‌رود.

2) ابتدا سه نقطه‌ی جوش را در زندگی خود انتخاب کنید. سه نتیجه‌ی کلی و نهایی را برای هریک، براساس اولویت‌ها و اهمیت آنها مشخص کنید. به طور مثال این سوال را از خود بپرسید که دوست دارید در سال آینده در هر یک از این بخشهای زندگی خود، چه دستاوردی را داشته باشید.

3) انتخاب سه دستاورد دلخواه را به صورت ماهیانه، هفتگی و روزانه تکرار کنید. توجه کنید که خروجی هر روز باید در راستای خروجی هفته و خروجی هر هفته در راستای خروجی هر ماه، باشد.

4) در پایان هر هفته کارهای انجام شده را با خروجی مورد انتظار مقایسه کنید. مشخص کنید که چه کارهایی خوب پیش رفته اند و

چه کارهایی نیاز به وقت، تمرکز و یا زمان بیشتری نیاز دارند و از اطلاعات و بازخورد به دست آمده برای برنامه ریزی هفته‌ی آینده استفاده کنید.

در مقالات آینده درباره اینکه چطور به صورت بهینه و کارآمد دیدگاه‌های هفتگی و خروجی‌های روزانه را مشخص و برنامه ریزی کنیم صحبت خواهیم کرد.

## نظرات خوانندگان

نویسنده: امیر بختیاری  
تاریخ: ۱۰:۳۴ ۱۳۹۳/۱۱/۱۱

با تشکر از مطلب مفیدتون  
می‌خواستم اسم کتاب به زبان اصلی و اگر آدرس لینک کتاب رو دارید قرار بدید

نویسنده: پگاه  
تاریخ: ۱۵:۵۶ ۱۳۹۳/۱۱/۱۱

خواهش می‌کنم. اسم کتاب " Getting Results the Agile Way: A Personal Results System for Work and Life " است که  
می‌شود از [آمازون](#) خریداری کرد. اما می‌توانید یک خلاصه بسیار مفید از این کتاب را که توسط خود نویسنده کتاب تهیه شده در  
این [آدرس](#) مطالعه کنید. موفق باشید.



در مقاله «[برنامه ریزی به روش چابک](#)» به قانون سه تایی اشاره کردیم. در این قانون سه خروجی یا دستاوردی را که مایل هستیم در ماه، هفته و یا یک روز داشته باشیم، به عنوان دیدگاه‌های هفته و خروجی‌های روزانه‌ی خود مشخص می‌کنیم. اما اگر تعداد کارها و دستاوردهای مورد نظرمان از سه مورد برای یک ماه، یک هفته و یا یک روز بیشتر باشد چه باید کرد؟ این مقاله درباره اینکه چطور به صورت بهینه و کارآمد دیدگاه‌های هفتگی و خروجی‌های روزانه را مشخص و برنامه ریزی کنیم می‌پردازد.

رمز موفقیت در روش برنامه ریزی چابک، اولویت بندی مؤثر کارها و خواسته‌هاست. زمانیکه شما احساس کنید دارید بر روی کار و هدفی درست، در زمانی مناسب کار می‌کنید تمرکز بیشتری بر روی کارتان خواهید داشت و بنابراین نتایج بهتری از انجام آن کار خواهید گرفت.

همه‌ی ما با اولویت بندی آشنا هستیم و معمولاً با اختصاص دادن شماره به هر مورد، آن مورد را اولویت بندی می‌کنیم. به طور مثال، «نوشتن مقاله برنامه ریزی به روش چابک» اولویت 2، «شبیه سازی الگوریتم همزمان سازی» اولویت 1 و «دویدن به مدت 30 دقیقه» اولویت 3، مثال‌هایی از اولویت بندی به روش سنتی است. اما آقای [Meier .J.D](#) در کتاب خودش روش مؤثرتری را که بر اساس **بایدها و نبایدها** بنا شده است، پیشنهاد می‌کند که در ادامه به آن اشاره می‌کنیم.

### اولویت‌ها در مدل چابک

در این روش سه درجه از اولویت وجود دارند. درجه‌ی اول، کارهایی هستند که حتماً باید انجام بگیرند. درجه دوم، کارهایی هستند که بهتر است (بایستی) انجام شوند و درجه سوم، کارهایی هستند که می‌شود انجام شوند و یا نشوند. آقای Meier این سه درجه را به ترتیب با سه واژه "Should"، "Must" و "Could" مشخص کرده است.

### روش اولویت بندی در مدل چابک

برای تهیه سه دیدگاه هفته و خروجی روزانه، ابتدا لیستی از اهداف و کارهای مورد نظر خود را تهیه کنید. سپس از خود بپرسید: (1) کدامیک از اقلام این لیست را باید انجام دهید؟ (2) کدامیک را بهتر است که انجام دهید؟ (3) کدامیک را می‌توانید انجام دهید؟ پس از مشخص کردن اولویت‌ها، سه خروجی روزانه و یا سه دیدگاه هفتگی خود را از میان اقلامی که در دسته اول، یعنی **بایدها** قرار می‌گیرند، انتخاب کنید.

### مزایای اولویت بندی

1- نتیجه‌ای که از اولویت بندی کارها نصیبتان می‌شود ارزش این را دارد که روی فرآیند اولویت بندی، مدت زمانی را صرف کنید. بدون اولویت بندی شما نگران یک لیست طولانی از کارهای خود هستید؛ در حالیکه فکر می‌کنید مشغول انجام یک کار مهم هستید. در آخر روز متوجه می‌شوید که کاری که باید انجام می‌گرفته است، انجام نشده است.

2- با تعیین اولویت‌ها برای هفته و هر روز خود، شما حداقل کارهایی را که باید برای هفته یا هر روز خود انجام دهید، مشخص کرده‌اید. زمانیکه این **بایدها** را انجام دهید، بقیه هفته و یا روز برای شما خواهد بود و می‌توانید از آن لذت ببرید!

3- اولویت بندی به شما قابلیت انعطاف می‌دهد. اگر یک کار یا فعالیت جدید پیش بیاید مثلاً اگر رئیس شما کار جدیدی را به شما محول کند، شما می‌توانید با تعیین درجه اولویت آن کار و مقایسه آن با بایدهای درحال انجام (سه خروجی روزانه یا دیدگاه هفتگی) تصمیم بگیرید که فعالیت محول شده جدید را انجام دهید یا انجام آن را به زمان دیگری موکول کنید.

بدون شک اولویت بندی یک لیست طولانی از کارها و فعالیت‌ها، کار مشکل و زمان بری است. در مقاله‌ی آینده درباره‌ی اینکه چگونه لیست کارها و فعالیت‌های خود را با مهارت، انتخاب و سازماندهی کنیم، خواهیم پرداخت.

تا کنون با **روش برنامه ریزی چابک**، اهمیت و نحوه **اولویت بندی** فعالیت‌ها در این مدل آشنا شدیم. اما بدون شک اولویت بندی یک لیست طولانی از کارها و فعالیت‌ها کار مشکل و زمان بری خواهد بود، به‌خصوص اگر فردی پر مشغله با مسئولیت‌های فراوان باشید. در مدل برنامه ریزی به روش چابک، پیشنهاد شده‌است که لیست فعالیت‌های کاری خود را دسته بندی کنید. در این روش، دسته بندی با الهام از روش کار بخش اورژانس بیمارستان‌ها، براساس درجه ضرورت کار و اضطراری بودن آن فعالیت انجام می‌شود.

در این روش چهار دسته وجود دارد که هر فعالیت می‌تواند در آن دسته قرار بگیرد: (1) **انجام شود**، (2) **موکول شود**، (3) **زمانبندی شود**، (4) **واگذار شود**.

(1) دسته اول کارهایی هستند که **هم اکنون** زمان انجام آنها فرا رسیده است. به این معنی که یا اکنون بهترین زمان انجام آنهاست، یا اگر آنها را به آینده موکول کنید مجبور خواهید بود انرژی و وقت بیشتری را صرف انجام آنها کنید.

(2) دسته دوم کارهایی هستند که **می‌بایستی** انجام شوند اما اکنون زمان مناسبی برای انجام آنها نیست. بنابراین شما می‌توانید آنها را در **صف انتظار** قرار دهید و در آینده تصمیم بگیرید که با آنها چه کنید.

(3) دسته سوم کارهایی هستند که برای شما **اهمیت بالایی** دارند، اما اکنون زمان مناسبی برای انجام آنها نیست. آنها را **زمانبندی** کنید. به این معنی که در تقویم خود برای انجام آنها یک زمان مشخص در نظر بگیرید تا در آن زمان انجام شوند.

با تخصیص دادن زمان به انجام یک کار در آینده فکر خود را از مشغولیت درباره آن آزاد می‌کنید. به طور مثال، شما می‌دانید که باید گزارش ماهیانه خود را در پایان ماه برای رییس خود آماده و ارایه کنید. بنابراین برای نوشتن گزارش، زمانی را مانند آخرین هفته‌ی ماه، در نظر می‌گیرید. به این ترتیب هرگاه مشغول انجام کار دیگری هستید و ناگهان فکر نوشتن گزارش به شما هجوم می‌آورد، می‌دانید که برای انجام این کار زمان خاصی تعیین شده‌است و آن را در زمان مناسب انجام خواهید داد. بدین ترتیب، آرامش فکری خود را باز می‌گردانید.

(4) دسته چهارم کارهایی هستند که در حقیقت می‌توانند توسط افراد دیگری انجام شوند. بنابراین آنها را واگذار کنید. توجه کنید که واگذار کردن کارها باید با مهارت انجام شود. بدین معنی که فرد موردنظر باید تخصص و انگیزه انجام آن کار را داشته باشد. به طور مثال، اگر شما سرپرست یک تیم در محیط کار خود هستید، کارها را با مهارت به اعضای گروه اختصاص دهید؛ به جای اینکه سعی کنید خودتان همه کارها را انجام دهید.

سؤالات زیر را در نظر بگیرید:

1- چه دستاوردی را می‌خواهید داشته باشید؟

2- آیا واقعا مهم است؟

3- چقدر مهم است؟

4- به انجام رساندن این کار چه تاثیر یا نتایج خواهد داشت؟

5- بهترین کاری که می‌توانید الان انجام دهید چیست؟

6- آیا حتما باید توسط من انجام گیرد؟

7- و...

این‌ها نمونه سؤالاتی هستند که می‌توانید در زمان دسته بندی کردن کارها از خود بپرسید. سؤالاتی از این قبیل شما را راهنمایی می‌کنند تا دسته بندی کارها را با مهارت بیشتر و بهتری انجام دهید.

بعد از دسته بندی کارها، حالا لیستی از کارهایی را دارید که **هم اکنون** زمان انجام آنها فرا رسیده است. آنها را توسط سیستم اولویت بندی چابک، اولویت بندی نمایید و 3 تا از بایدهای لیست را به عنوان دیدگاه هفتگی و یا خروجی روزانه انتخاب کنید.

## نظرات خوانندگان

نویسنده: فرید بکران  
تاریخ: ۱۳۹۳/۱۲/۱۶ ۱۷:۰۶

سلام.

فرق موکول و زمان بندی دقیقا چیست؟

هر دو در زمان آینده انجام خواهند شد؛ ولی برای موکول ها زمان بندی در نظر گرفته نمی شود؟ در واقع اولویت آن ها کمتر از زمان بندی ها است. درسته؟

نویسنده: پگاه  
تاریخ: ۱۳۹۳/۱۲/۱۶ ۲۰:۲۰

سلام. درست است. در حقیقت کاری را زمان بندی می کنیم که در آینده نزدیک باید انجام شود. به طور مثال بخشی از یک پروژه که باید تا تاریخ خاصی به اتمام برسد را در نظر بگیرید. اگر اکنون زمان مناسبی برای انجام این پروژه نیست بهتر است برای انجام آن زمان مشخصی را در تقویم خود ثبت کنید. حال فرض کنید شما می خواهید راجع به ایده جدیدی با همکار خود بحث و گفتگو کنید اگر اکنون زمان مناسبی برای انجام آن نباشد بهتر است آن را به آینده موکول کنید. ناگفته پیداست این تصمیمات کاملاً شخصی است و وابسته به شرایط هر فرد است.

با همکاری آقایان [سید مجتبی حسینی](#) و [محمد شریفی](#) طی یک سری مقالات سریالی قصد داریم ترجمه آزادی از کتاب [Pro Agile .NET Development With Scrum](#) نوشته Matthew Bussa و Jerrel Blankenship، داشته باشیم.

با توجه به اینکه در سایت جاری مطالب قسمت اول کتاب پوشش داده شده است، ما هم دوباره کاری نکرده و میتوانید از [این مقاله](#) استفاده کنید.

### مدیریت پروژه‌های چابک با اسکرام

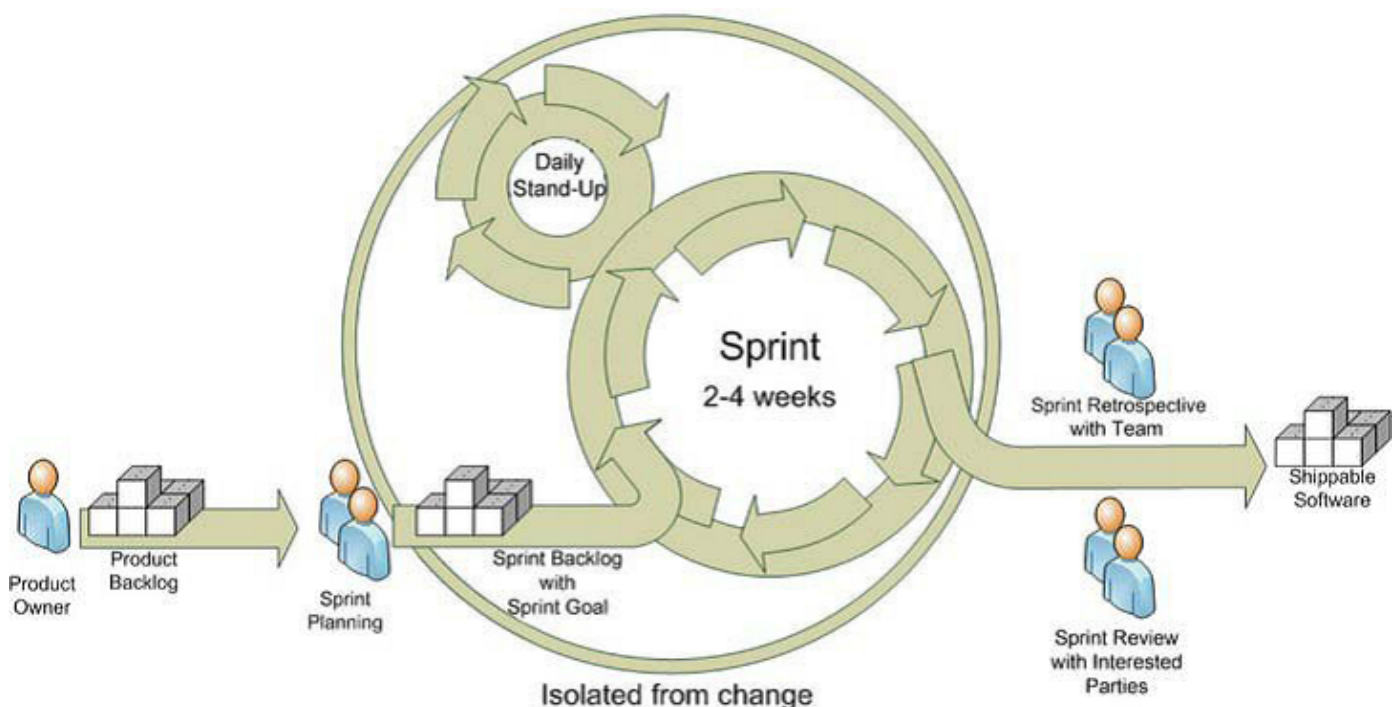
در این فصل با روشها و ماهیت تکرارپذیر اسکرام آشنا می‌شوید که استخوان‌بندی فرآیندی را تعریف می‌کند که دربردارنده مجموعه‌ای از نقشها و فعالیتهایی است که همگی بر پشتیبانی از تیم مسؤول تولید محصول، تمرکز می‌کنند. مطالعه موردی بخش دوم این کتاب از شیوه اسکرام به نحوی پیروی می‌کند که قادر به دیدن اجرایی عملی از تمام ویژگیهای کلیدی‌ای که در این فصل از آنها بحث می‌شود، خواهید بود و به شما کمک می‌کند تا مزیت‌های این شیوه را به خوبی درک کنید.

### اسکرام چیست؟

اسکرام رویکردی تکرارپذیر جهت توسعه نرم‌افزار است که بصورت تنگاتنگی با اصول و بیانیۀ چابک هم‌سو شده است. اسکرام از دنباله‌ای از بلاکهای زمانی به نام اسپرینت ساخته شده است که بر ارائه محصولات کارآمد تمرکز می‌کند. یک اسپرینت نوعاً از دو تا چهار هفته به طول می‌انجامد و با هدف یا موضوعی که واضح کننده اسپرینت است، تعریف شده است. اسپرینتها نسبت به تغییرات ایزوله شده‌اند و بدون هیچ اختلالی، تیم توسعه را بر ارائه محصولی کارآمد، متمرکز می‌سازند. کارها در Product Backlog (لیستی از کارهای کلی یک پروژه است که باید آن را بر اساس درجه اهمیت، دسته بندی نمود) اولویت‌بندی شده که توسط صاحب محصول مدیریت می‌شود. قبل از وقوع هر اسپرینت، یک ویژگی از Product Backlog انتخاب شده و تیم توافق می‌کند که در انتهای آن اسپرینت، آن ویژگی را ارائه کند.

برای آنکه همه چیز بخوبی پیش برود، یک نفر به عنوان ScrumMaster (که وظیفه نگهداری و حفظ فرآیند را برعهده دارد) تعیین می‌شود تا اطمینان حاصل شود که هیچ مانعی باعث جلوگیری از ارائه ویژگیهایی که تیم توسعه مد نظر قرار داده، نشود. جلسات سرپایی روزانه به تیم کمک می‌کند تا درباره هر مشکلی که مانع کار است، گفتگو کنند. مرور هر اسپرینت در انتهای آن به ارتقای فرآیند کمک می‌کند.

شکل 2-1 نمایش گرافیکی روش اسکرام است که حاوی همه نقشها و فعالیتهای اسکرام بوده که در ادامه، بیشتر درباره آنها خواهید خواند.



## شیوه‌های برنامه محور در مقابل شیوه‌های ارزش محور

هنگام ملاحظه تفاوت میان شیوه آبخاری و شیوه چابک، نیاز است تا به هسته مرکزی هر روش نگریم. یکی از شیوه‌ها از نقشه‌ای برگرفته شده که در ابتدای پروژه ایجاد شده است و شیوه دیگر از ارزشی برگرفته شده که شما به مشتری می‌دهید.

### شیوه آبخاری (برنامه محور)

به شیوه آبخاری می‌توان به منزله شیوه‌ای برنامه محور در توسعه نرم‌افزار نگریم. در گذشته، این شیوه توسعه بسیار مورد استفاده بود، نه به این دلیل که بهترین شیوه توسعه نرم‌افزار بود، بلکه به این دلیل که تنها شیوه شناخته شده بود. پروژه‌ای که شیوه آبخاری را به کار می‌برد با ریسک بسیار بالایی مواجه بود؛ به این دلیل که همه چیز در ابتدای پروژه طرح‌ریزی می‌شد. تمام نیازمندیها و جستجوها و تعیین بازه کاری قبل از آنکه حتی یک خط کد نوشته شود، جمع‌آوری می‌شد. مشتریان باید همه آنچه را که از سیستم انتظار داشتند، در ابتدای امر می‌دانستند. در زمانی که مشتریان دقیقاً نمی‌دانستند که چه می‌خواهند اما باید تمام جزئیات نیازهای خود را تعریف می‌کردند و در یک وهله باید جزئیات کار را تعیین می‌کردند و تا آخر نیز نمی‌توانستند آن را تغییر دهند؛ حتی اگر بعداً متوجه می‌شدند که نیازشان تغییر کرده است.

این رویکرد سرانجام پروژه را، حتی قبل از آنکه شروع شود، با شکست مواجه می‌کرد. کل فرآیند به سمت مشکلاتی هدایت می‌شد که تا پایان پروژه نیز پنهان می‌ماندند. زیرا مشتری همه نکات جزئی کار را مدنظر قرار نداده بود و راهی برای تغییرات مورد نیاز وجود نداشت. گاهی انجام تغییر مستلزم هزینه بسیار بالایی بود. در این گونه پروژه‌ها دامنه پروژه دچار تغییرات می‌شد؛ توسعه‌دهنده از مسائلی که مشتری درصدد حل آنها بود سردر نمی‌آورد و به همین ترتیب مشتری.

توسعه برنامه محور به مانند روند پرش حلقه‌ای است؛ شما ابتدا جستجو می‌کنید و یک مرتبه از میان آن حلقه پریده و وارد حلقه جمع‌آوری نیازمندیها می‌شوید و از آنجا وارد حلقه طراحی می‌شوید. از یک حلقه نمی‌توانید عبور کنید مگر اینکه از حلقه پیشین آن پریده باشید و با یک مرتبه، عبور از یک حلقه برگشتن به آن حلقه ممکن نیست. حتی اگر نیاز باشد چنین کاری انجام شود. ممکن نیست که اندکی از هر کاری را انجام داده و برای اطمینان از مسیر درست، قدری متوقف بمانید. فرآیند آبخاری فراهم کننده بستری نیست که در آن توسعه دهند بتواند به مشتری خود بگوید: «مایل هستم که کاری را که تاکنون انجام داده‌ام، به شما نشان دهم تا ببینید که آیا با آنچه شما می‌خواهید منطبق است یا خیر».

معمولاً در انتهای پروژه است که مشکلات بزرگی بروز پیدا می‌کنند که نسبتاً خیلی دیر است. این مورد منجر به آن می‌شود که چند تیم به کار وارد شده و افراد بیشتری در پروژه استفاده شوند؛ به این امید که پروژه سریعتر به اتمام برسد و البته چنین نتیجه‌ای به ندرت اتفاق می‌افتد. در نتیجه بخشهایی از پروژه باید کنار گذاشته شوند؛ یعنی یا حدود پروژه محدودتر شود، یا آزمودن آن حذف شود یا هر دو.

### شیوه اسکرام (ارزش محور)

اسکرام به عنوان شیوه‌ای ارزش محور در توسعه نرم‌افزار مورد توجه قرار می‌گیرد. اسکرام به چند دلیل تغییر چشم‌گیری نسبت به شیوه آبخاری داشته است. اسکرام به جای آنکه در ابتدا به جمع‌آوری نیازمندیهای مورد نیاز برای هر ویژگی مد نظر پروژه بپردازد و به جای آنکه همه طراحی‌های خود را مبتنی بر این نیازمندیها کامل کرده و سپس به کدنویسی برنامه مبتنی بر این طرحهای اول مشخص شده بپردازد؛ به توسعه تکرارپذیر و افزایشی می‌نگرد. اسکرام تماماً معطوف به مسیریابی جزئی در حل مسأله و ارزیابی مجدد آن مسأله پس از طی هر مسیر است. بلاکهای جزئی با عنوان اسپرینت

ویژگیهای جزئی

تیم‌های کوچک

بلاکهای زمانی کوچک بیانگر چگونگی کار بر روی حل مسأله توسط تیم توسعه است. به هر اسپرینت می‌توان به صورت یک پروژه آبخاری کوچک نگریم. زیرا در هر اسپرینت شما همه کارهایی را که به طور عادی در یک پروژه آبخاری انجام می‌دهید، اجرا می‌کنید با این تفاوت که فقط در مقیاسی کوچک‌تر آن را انجام می‌دهید. در هر اسپرینت، شما یک ویژگی را انتخاب کرده و نیازمندیهای آن ویژگی را جمع‌آوری کرده و به طراحی آن ویژگی مبتنی بر نیازمندیهای به دست آمده پرداخته و سپس کدنویسی کرده و آن خصیصه را با توجه به طراحی صورت گرفته، تست می‌کنید. شما در اسکرام برخلاف روش آبخاری، تلاش نمی‌کنید که همه چیز را پیشاپیش طراحی کنید. بلکه شما چیزی را انجام می‌دهید که نیاز است انجام شود. هدف هر اسپرینت انجام ارتقایی

(افزایشی) برای رسیدن به پروژه نهایی است؛ اما افزایشی که به طور بالقوه قابل ارائه است. حال چگونه می‌توان در هر اسپرینت تعداد زیادی پروژه‌های آبشاری را انجام داد، در حالی که قبلاً به سختی یک پروژه آبشاری قابل انجام بود؟ جواب، انجام اسپرینتهایی با ویژگیهای کوچک است. ویژگیهای جزئی، قطعاتی از پروژه هستند که تلاش می‌کنند مسأله خاصی را برای مشتری حل کنند. آنها درصدد این نیستند که کل برنامه را ایجاد کنند. ویژگیهای مدنظر یک پروژه به تکه‌های کوچکتری شکسته می‌شوند که هنوز قادر به تامین ارزش برای مشتری بوده و می‌توان آنها را به سرعت انجام داد. با هرچه بیشتر شدن این ویژگیهای کامل شده در پروژه، مشتری کم کم با نمای کامل برنامه مورد نظر مواجه شده و آن را ملاحظه می‌کند. همه‌ی این موارد توسط یک تیم کوچکی از توسعه‌دهندگان، تست‌کننده‌ها و طراحانی که صرفاً به انجام پروژه مشغول هستند، انجام می‌شود. این تیم، یک تیم با قابلیت‌هایی چندگانه است که هر عضو آن با انجام تمام کارهای تیم آشناست. هر عضوی از آن ممکن است که در همه چیز بهترین نباشد؛ اما هرکس می‌داند که چگونه یک کار ضروری را برای تکمیل پروژه انجام دهد. نگرستن به آنها به عنوان یک تیم SEAL که هر عضو آن می‌داند که چگونه هرچیز مورد نیاز را انجام دهد، اما برای هرکاری کارشناسان مخصوص آن کار وجود دارد.

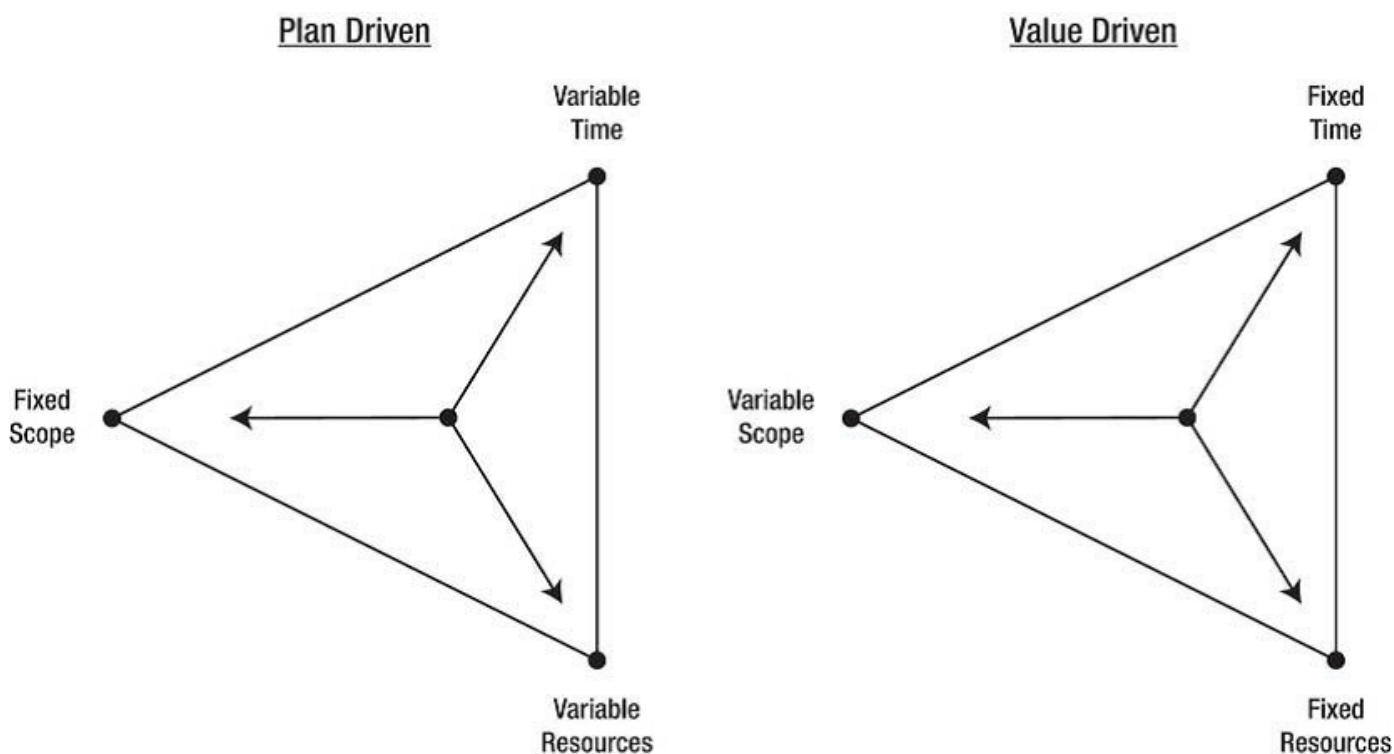
با انجام این کار در سطوح جزئی، مسائل این سطوح جزئی تا حدی شبیه مسائلی هستند که در انتهای پروژه در شیوه آبشاری رخ می‌دهند. در واقع اسکرام به گونه‌ای کار می‌کند که بتواند تا آنجا که ممکن است سریعاً مشکلات و مسائل را نشان دهد. مشکلات قابل پنهان شدن نیستند؛ چراکه پروژه به سطوحی کوچک و قابل مدیریت، تجزیه شده است. هنگامیکه مشکلی بروز پیدا می‌کند، تا وقت پیدا شدن راه حل و حل شدن آن، موجبات دردسر تیم را فراهم می‌کند و آنها نمی‌توانند از مسأله چشم‌پوشی کنند، چون برای همه قابل رؤیت است.

نکته بسیار مهمی را باید درباره اسکرام فهمید و آن اینکه اسکرام مشکلات را هرچه زودتر، به تیم نشان می‌دهد؛ اما آنها را حل نمی‌کند.

اسکرام نه تنها ویژگیهایی را برای نمایش به مشتری توسط تیمهای فروش و بازاریابی تولید می‌کند، بلکه راه‌حلهایی را نیز به مشتری ارائه می‌دهد. چنین امری با اولویت‌بندی خصوصیت‌ها مطابق نیاز و خواسته‌های مشتری، صورت می‌گیرد. اگر مشتری‌ای تصور کند که ویژگی A باید از ویژگی B بسیار مهم‌تر باشد و توسعه دهند، وقت زیادی را بر سر ویژگی B قبل از ویژگی A صرف کند، نمی‌تواند به نیاز مشتری به نحو مطلوبی، پاسخ‌گو باشد.

#### عوامل ثابت در مقابل عوامل متغیر

سه عامل یا قید کلیدی، برای هر پروژه نرم‌افزاری وجود دارند: زمان، منابع و محدوده پروژه. متأسفانه در یک زمان، هر سه عامل قابل جمع نیست. طبق شکل مثلی زیر، در هر زمان می‌توان بر روی تاثیرات دو عامل کار کرد و آن دو عامل اتفاقی را که رخ می‌دهد، بر سومی دیکته می‌کنند.



در مدل توسعه برنامه محور، حیطه و منابع پروژه، معمولاً عوامل ثابتند و زمان عامل متغیر است. در این حالت حیطه پروژه بر منابع و زمان حاکم است. این حالت تا زمانی خوب است که شما در میانه پروژه قرار دارید. اما به مرور، رشد حیطه پروژه، چهره نامطلوب کار را نمایان می‌سازد. در این هنگام محدوده پروژه، گسترش خواهد یافت در حالی که نه منابع و نه زمان، متناسب با چنین تغییری، قابل تغییر نیستند. در این هنگام شما افراد بیشتری را به پروژه وارد می‌کنید، به این امید که به نتیجه مناسبی در انتهای کار دست یابید.

در مدل توسعه ارزش محور، منابع و زمان در مثلث ثابتند. شما از ابعاد تیمتان و سرعت انجام کارشان در اسپرینتهای قبلی آگاهی دارید. در این حالت محدوده پروژه در مثلث فوق، عنصر متغیر می‌شود. به عبارت دیگر منابع پروژه و زمان، تعیین‌کننده محدوده پروژه هستند.

### محصولات اسکرام

اسکرام سه خروجی دارد:

**product backlog** : مجموعه‌ای اولویت بندی شده از نیازمندی‌های سطح بالای سیستمی که در نهایت بایستی تحویل داده شود.

**sprint backlog** : مواردی از product backlog که قرار است در یک sprint انجام شوند.

**نمودار burn-down** : هدف نمودار burn-down، نمایش روند پیشرفت پروژه به صورت نموداری به اعضای تیم توسعه است که حاوی اطلاعاتی درباره کل زمان انجام کار، زمان تخمین زده شده، مقدار کارانجام شده و عقب‌ماندگی‌های پروژه است.

این خروجی‌ها، محصولات فعالیت‌های اسکرام هستند و به تیم در جهت‌یابی و شفافیت کار کمک می‌کنند. افزون بر این خروجی‌های اصلی خروجی‌های فرعی‌ای نیز از قبیل معیار پذیرش (الزاماتی که باید در حل یک مسئله برآورده شود تا بتوان آن را کامل شده تلقی کرد) وجود دارد.

### Product Backlog

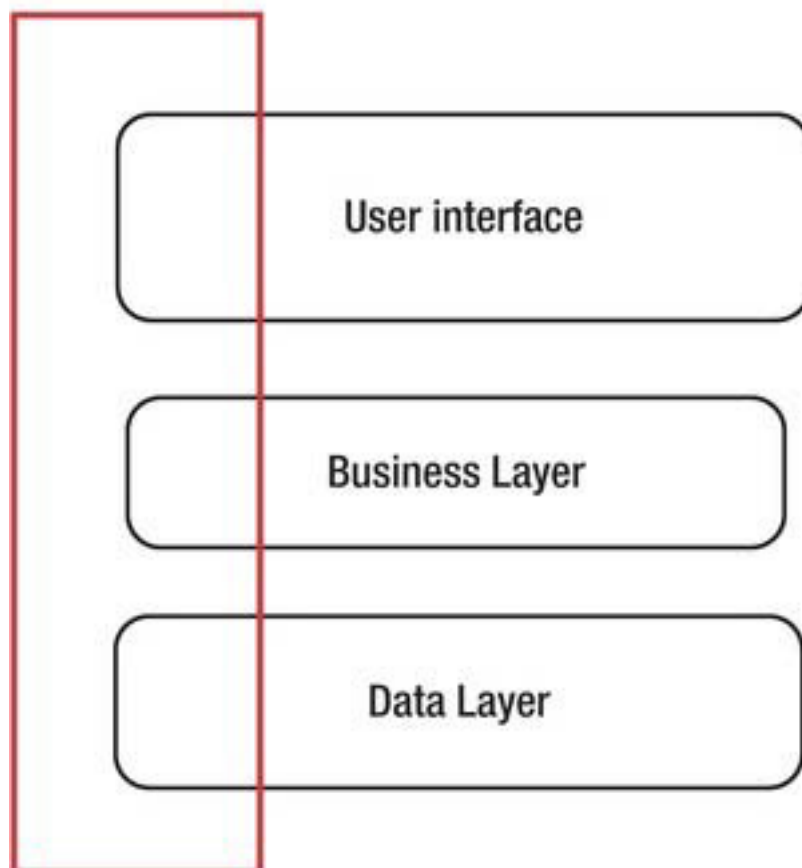
product backlog لیستی از همه کارهای باقی‌مانده در یک پروژه است که باید انجام شوند. این لیست نمایانگر نیازمندیها و خواسته‌های مشتری است. در قلب این لیست «داستان کاربر (user story)» یعنی مؤلفه کلیدی اسکرام قرار دارد. این مؤلفه تعیین‌کننده ملاک افزایش ارزش در نزد مشتری بوده و آن چیزی است که توسعه دهنده تلاش می‌کند، ارائه نماید و توسط صاحب محصول (product owner) (یعنی کسی که نسبت به افزودن یا حذف داستان کاربر (user story)ها به لیست، پاسخگو است) مدیریت می‌شود. product backlog به طور دائم توسط صاحب محصول و مشتری اولویت‌بندی می‌شود. این اولویت‌بندی دائمی امری کلیدی برای اسکرام است. این امر تضمین می‌کند که داستان کاربر (user story) که تعیین‌کننده بیشترین ارزش برای مشتریست، در صدر product backlog قرار گرفته باشد. با افزوده شدن یک داستان کاربر (user story) این مورد با سایر داستان‌های کاربر (user story)های پیشین مقایسه شده تا مشخص شود که در چه سطح ارزشی‌ای از نظر مشتری قرار دارد. در طول یک اسپرینت، داستان کاربر (user story)ها را می‌توان به اسپرینت اضافه کرد. اما تا کامل شدن اسپرینت جاری، به تیم توسعه نشان داده نمی‌شود.

### User Stories

همانطور که خاطر نشان شد، product backlog چیزی بیش از یک لیست اولویت‌بندی شده از داستان‌های کاربر (user story) نیست. یک داستان کاربر (user story)، یک کارت است که ارزش اضافه‌ای را برای مشتری توصیف می‌کند. داستان کاربر (user story) برای توسعه‌دهنده به منظور بیان ارزشی اضافه نوشته می‌شود. نکته کلیدی یک داستان کاربر (user story) خوب، این است که داستان کاربر (user story) بخشی عمودی از لیست است و بخش افقی ویژگی‌ای است که فقط به یک سطح، مانند سطح بانک اطلاعات یا سطح رابط کاربری اثر می‌گذارد. به عبارت دیگر قطعه عمودی تمام سطوح را آن گونه که در شکل 2-3 نشان داده شده، متأثر می‌سازد. این کوچکترین مقدار کاری است که تمام سطوح یک محصول را تحت تأثیر قرار داده و برای مشتری ارزش ایجاد می‌کند. با نوشتن داستان‌های کاربر (user story) به گونه‌ای که در بخشهای عمودی جایز است، می‌توان قابلیت پایه‌ای را در اولین داستان کاربر (user story) ایجاد کرده و سپس به سادگی قابلیت به این ویژگی به عنوان نیازهای مشتری اضافه کرد.



## Vertical Slice



یک شیوه اطمینان از اینکه داستان کاربر (user story) فایده قطعه عمودی بودن در یک سیستم را داراست این است که مطمئن شویم با «INVEST» منطبق است. عبارت «INVEST» عبارت است از مخفف: مستقل (Independent): باید خودبسنده باشد و به سایر داستانها وابسته نباشد.

قابل مذاکره (Negotiable): داستانهای کاربری که بخشی از یک اسپرینت هستند همیشه قابل تغییر و بازنویسی هستند.

با ارزش (Valuable): یک داستان کاربر باید به کاربر نهایی، ارزشی را ارائه دهد.

قابل برآورد (Estimable): همیشه باید بتوان اندازه داستان کاربر را تخمین زد.

اندازه مناسب (Sized appropriately): داستانهای کاربر نباید آن قدر بزرگ باشند که تبدیلشان به یک طرح یا وظیفه یا امر اولویت بندی شده با درجه مشخصی ممکن نباشد.

قابل آزمون (Testable): داستان کاربر یا توصیفات مربوط به آن باید اطلاعات ضروری برای آزمودن آن را فراهم کنند.

### تعیین اندازه backlog

تعیین اندازه product backlog عبارت است از اندازه گیری سرعتی که تیم اسکرام میتواند مؤلفه های آن را ارائه کند. افراد در تخمین کار خوب عمل نمی کنند. همگی می دانیم که در تخمین دقیق اینکه چقدر طول می کشد تا یک کار را به طور کامل انجام دهیم، تا چه اندازه بد عمل می کنیم. تا کنون چند مرتبه این اتفاق افتاده است که از کسی بشنویم یا به خودمان بگوییم که 80 درصد کار را انجام داده ام و 20 درصد باقی مانده آن در یک ساعت انجام خواهد شد. اما هنوز بعد از دو روز انجام نشده است. افراد به طور

طبیعی بد تخمین می‌زنند.

ما ممکن است در تخمین زدن خوب نباشیم؛ اما در مقایسه کردن اشیاء با یکدیگر عالی هستیم. به عنوان مثال قادریم که با نگاه انداختن به دو دستور پخت غذا تشخیص دهیم که کدام یک پیچیده‌تر از دیگری است؛ بدون آنکه تخصصی در آشپزی داشته باشیم. به دو چیز نگاه می‌کنیم و تشخیص می‌دهیم که کدام یک بزرگتر از دیگری است. تخمین اندازه backlog تماماً یعنی تصمیم‌گیری درباره پیچیدگی و مقدار کار لازم، نه اینکه چقدر طول می‌کشد تا این کار انجام شود. تخمین اندازه با تخمین برابر نیست. ممکن است بپرسید که چگونه می‌توان زمان انجام برخی چیزها را اندازه گرفت؟ مدیری را در نظر بگیرید که می‌خواهد بداند چقدر طول می‌کشد تیم شما یک widget را تولید کند. شما می‌توانید تخمین زمان کامل شدن widget را از پیچیدگی widget تخمین بزنید. شما می‌توانید وقتی که تیم از یک اسپرینت فارغ شد، به آن اسپرینت نگاه کرده و محاسبه کنید که چقدر طول می‌کشد تا کار کامل شود. فقط پیچیدگی یک وظیفه‌ی مورد توجه تیم است.

اجازه دهید برای توضیح بهتر چگونگی تخمین مقدار کار مورد نیاز برای کامل شدن کار، این مسأله را با رنگ‌آمیزی خانه‌تان مقایسه کنیم. شما به فروشگاه رنگ فروشی رفته و چند سطل رنگ را برای رنگ‌آمیزی خانه می‌خرید. سپس از سه پیمانکار می‌خواهید که انجام این کار را برای شما تخمین بزنند. اولین پیمانکار به خانه‌ی شما آمده و دور خانه قدم زده و به سطلهای رنگی که خریده‌اید نگاه کرده و می‌گوید که وی با یک نردبان زنگ زده و برس‌های دستی و پسرکی لاغراندام به عنوان دستیارش، این کار را در ظرف دو روز انجام می‌دهد.

دومین پیمانکار دور خانه قدم زده و به سطلهای نگریسته و می‌گوید که به تازگی نردبان و برس‌هایی خریداری کرده است و تیم محلی فوتبال در آخر هفته به وی کمک خواهند کرد. با این دستیاران و تجهیزات جدید، انجام این کار فقط یک روز به طول می‌انجامد.

سومین پیمانکار دور خانه قدم زده و به رنگها نگریسته و می‌گوید که وی صاحب یک دستگاه مکانیکی رنگ‌آمیزی است که باعث می‌شود انجام این کار حدود یک ساعت وقت بگیرد.

شما درباره این ماجرا و سه نوع تخمین از رنگ‌آمیزی خانه، چگونه فکر می‌کنید در صورتی که در هیچ کدام از این سه وضعیت، نه ابعاد خانه تغییری کرده است و نه مقدار رنگی که شما خریداری کرده‌اید. نکته این داستان در این است که بهترین چیزی که شما می‌توانید انجام دهید تخمین مدت زمان انجام کار نیست؛ بلکه به جای آن باید مقدار تلاشی را که منجر به اتمام کار خواهد شد، تخمین زد. با تخمین مقدار کار می‌توان مدت زمان انجام کار را به دست آورد.

### Sprint Backlog

sprint backlog لیستی از همه کارهای باقی‌مانده در یک اسپرینت است و باید توسط تیم انجام شود. sprint backlog زیرمجموعه‌ای از product backlog است. product backlog همه داستانهای کاربران را که برای product مانده لیست می‌کند؛ اما sprint backlog حاوی همه داستانها و وظایف باقی‌مانده برای اسپرینت است. نوعاً هنگامی که یک داستان کاربر برای یک اسپرینت انتخاب می‌شود، تیم، آن داستان کاربر را به تعدادی وظیفه تقسیم می‌کند.

یک وظیفه، تکه کوچکی از داستان کاربر است که توسط هر عضو تیم قابل انجام است. مثلاً وظایفی از قبیل اجرای تغییرات بر روی بانک اطلاعاتی مورد نیاز یک داستان کاربر یا وظیفه اجرای UI برای داستان کاربر. وظایفی که بر روی تابلوی وظایف - که با عنوان Kanban (معادل ژاپنی بیلبرد) نیز شناخته می‌شود- برای همه تیم قابل رؤیت است. سایر مؤلفه‌های روی این تخته به همان ترتیب، حاوی اطلاعاتی درباره قرار ملاقاتهای جمع‌آوری نیازمندیها، کنترل‌های بازبینی، تحقیقات، آزمون، طراحی و مراحل کد نویسی هستند. شکل 2-4 یک مثال را نشان می‌دهد.

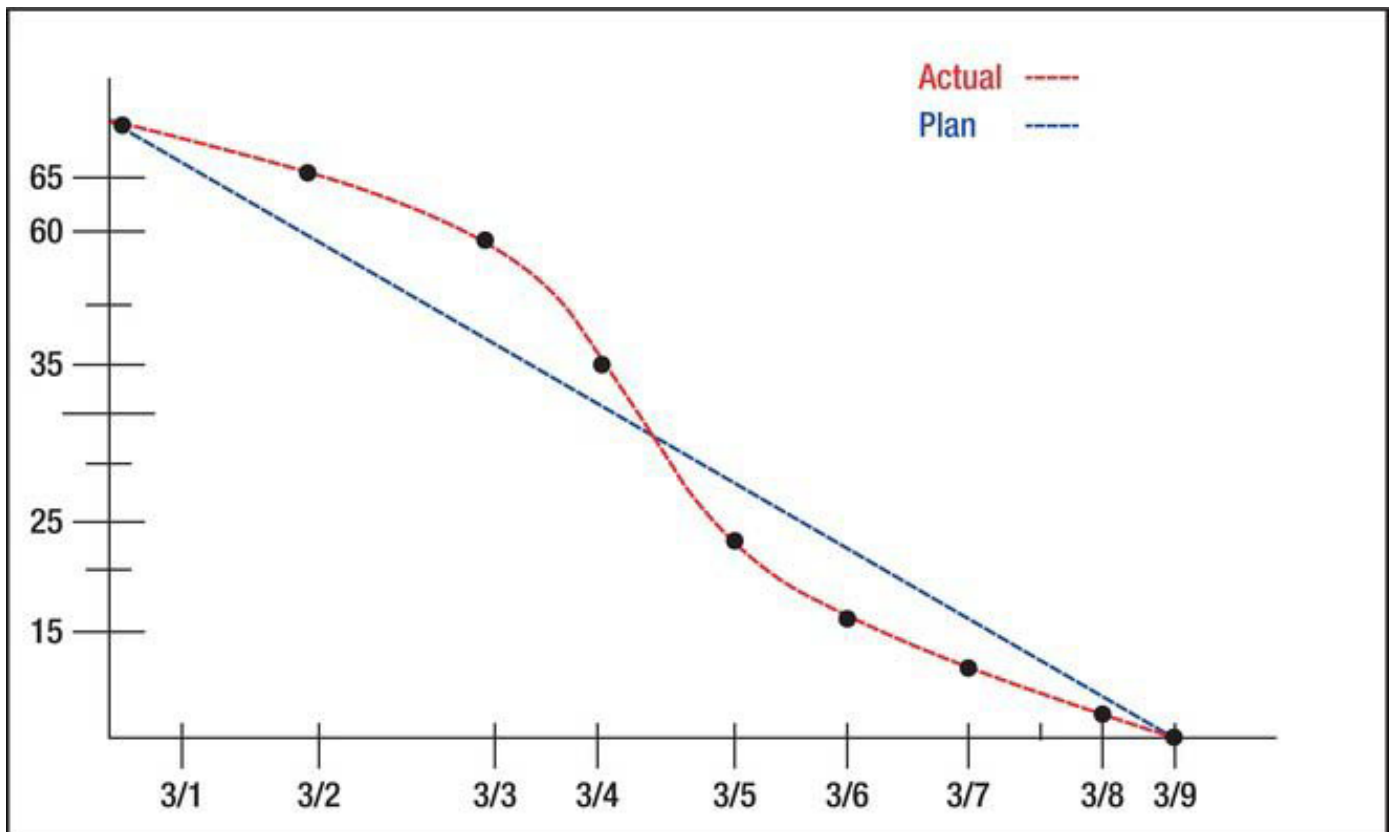
اعضای تیم یک کارت از تخته برداشته و در طول اسپرینت اقدام به انجام وظیفه‌ای که روی کارت توصیف شده، می‌نمایند. در خلال مدتی که تیم بر روی وظایف کار می‌کند، سایر وظایف بروز پیدا کرده و تخمینهای اصلی مجدداً تنظیم می‌شوند. همه اعضای تیم، در قبال به روز رسانی تابلو بر طبق اطلاعات جدید مقید خواهند بود.

| Backlog   | Currently working on  | Currently in QA                                     | Done   |
|---|---|---|--|
| <div>implement database schema for insert feature</div> <div>abilit to edit a record in the system</div> <div>abilit to delete a record from the system</div> | <div>right unit tests for userstory</div> <div>implement new object in business layer</div> | <div>ability to insert a new record in system</div> | <div>log in screen</div> <div>implement new object in business layer</div> <div>right unit tests for userstory</div> |

sprint backlog اطلاعات مورد نیاز نمودار burn-down را فراهم می‌کند. در پایان هر اسپرینت، sprint backlog خالی می‌شود. هر آیتم باقی مانده‌ای در backlog به product backlog برگردانده شده و مجدداً در کنار سایر داستانهای کاربری موجود در product backlog به‌علاوه داستانهای کاربری تازه وارد شده، اولویت‌بندی می‌شود.

#### Burn-down chart

نمودار burn-down شیوه‌ای بصری برای دنبال کردن چگونگی پیش‌روی یک اسپرینت است. این نمودار کار باقیمانده اسپرینت را در هر روز، به صورت گرافیکی همانند شکل 2-5 نشان می‌دهد. معمولاً این نمودار در یک محیط عمومی نمایش داده می‌شود تا هرکسی بتواند آن را ببیند. این کار به ارتباطات میان اعضای تیم و هرکس دیگری در سازمان کمک می‌کند. این نمودار همچنین می‌تواند به عنوان نشانگر وجود یک مسأله در اسپرینت عمل کند که تیم ممکن است بخاطر آن نتوانند به تعهد خود عمل کنند.



## معيار پذيرش

اگرچه sprint backlog ، product backlog و نمودار burn-down بخشهای اصلی اسکرام هستند، معيار پذيرش خروجی جانبی بسیار مهمی از فرآیند اسکرام است. بدون معيار پذيرش خوب، یک پروژه محکوم به شکست است.

معيار پذيرش ضرورتاً شفاف کننده داستان است. چنین معیاری مجموعه‌ای از گامهای مختلف را در اختیار توسعه دهنده می‌گذارد که پیش از آنکه کار تمام شده تلقی شود، باید انجام دهد. معيار پذيرش، توسط صاحب محصول (product owner) به کمک مشتری ایجاد می‌شود. این معيار انتظار از داستان کاربر را تنظیم می‌کند. استفاده از این معيار درجای خود نقطه شروع خوبی برای نوشتن تستهای خودکار یا حتی توسعه آزمون محور توسط توسعه دهنده است. بدین طریق، توسعه دهنده چیزی را تولید می‌کند که مشتری بدان نیاز داشته و آن را می‌خواهد.

دیگر مزیت معيار پذيرش وقتی آشکار می‌شود که یک ویژگی در طول یک اسپرینت کامل نشده و نیاز است تا از اسپرینت‌ها خارج شود. در چنین موردی تیم می‌تواند معيار پذيرش را به عنوان ابزاری به کار گیرد تا بفهمد که داستان کاربر چگونه به قطعات کوچکتری تقسیم شود تا کماکان ارزشی را برای مشتری فراهم کرده تا بتواند در یک اسپرینت کامل شود.

## نقش‌های اسکرام

اسکرام بین افرادی که نسبت به پروژه متعهد هستند و افرادی که فقط ذینفع محسوب میشوند، تمایز قابل توجهی قائل است. مشهورترین روش برای توضیح این مفهوم تعریف حکایت "Pig & Chicken" میباشد؛ یک خوک و یک جوجه در حال قدم زدن بودند که، یک دفعه جوجه به خوک گفت که: "چرا یک رستوران افتتاح نکنیم؟" خوک هم نگاهی به جوجه کرد و گفت: "ایده خوبی است، اسم آن را چه بنامیم؟" جوجه کمی در مورد این مسئله فکر کرد و گفت: "چرا اسمش را 'گوشت ران خوک و تخم مرغ‌ها' نگذاریم؟"

خوک جواب داد: "فکر نمیکنم جالب باشد چون من متعهد خواهم بود به کار، ولی تو فقط درگیر کار خواهی بود." بنابراین Pigs همان افراد متعهد به پروژه هستند که وظیفه ساخت، تست، گسترش و توزیع را ایفا میکنند. Chickens در طرف دیگر همان افرادی هستند که کمتر به پروژه تعهد دارند. این افراد همان stakeholderها و یا ذینفعانی هستند که از پروژه منفعت

می‌برند، اما در مقابل تحویل پروژه مسئول و پاسخگو نیستند.

### Pig Roles

نقش‌های عنوان شده در زیر جز نقش‌های Pig هستند که تیم اسکرام را نیز تشکیل می‌دهند:

Scrum Master

Product Owner

Delivery Team

Scrum Master

اگر تیم را موتور پروژه‌ی اسکرام در نظر بگیریم، اسکرام مستر روغنی است که موتور را در حال اجرا نگه می‌دارد. او مسئول این است که مطمئن شود فرآیند اسکرام تفهیم شده و دنبال می‌شود. اسکرام مستر تسهیل‌کننده‌ی جلسات تیم و حذف موانعی است که امکان دارد تیم در دوره‌ای از انجام کار خود با آن مواجه شد. او مطمئن خواهد بود که هیچ مانعی به عنوان بازدارنده از رسیدن به اهداف تیم در مقابل آنها وجود ندارد و تیم را از حواس پرتی‌های خارجی ایزوله نگه می‌دارد تا مطمئن شود اعضای تیم دقیقاً کاری را به آنها سپرده شده است انجام می‌دهند. اسکرام مستر با بخش‌های مختلف تیم، از صاحبان محصول گرفته تا تست‌کنندگان و ذینفعان کسب و کار در تعامل است، تا مطمئن شود که تمام اعضای تیم برای پروژه مفید هستند و تمام دست‌آوردهای مشترک در اسپرینت را به اشتراک می‌گذارند. اسکرام مستر را یک مدیر پروژه معمول فرض نکنید؛ چون نقشی که او ایفا می‌کند بیشتر از نقش یک مدیر پروژه است. مشخصه کلیدی اسکرام مستر "رهبر خدمتگزار است". او رئیس تیم نیست ولی به تیم کمک می‌کند تا به چیزی که نیاز دارند در این اسپرینت دست یابند. اسکرام مستر به تیم کمک می‌کند تا کار را به سمتی که خروجی با ارزشی برای مشتری دارد، متمایل کنند. زمانی که مسئله‌ای در داخل تیم بوجود آید، به اسکرام مستر انتقال داده می‌شود تا این تضاد را مدیریت کند. مواقعی هم وجود دارند که اسکرام مستر در نقش فرمانروا، ایفای نقش می‌کند. وقتی که یکی از مسئولیت‌های اسکرام مستر مطمئن شدن از این است که تمام روشهای اسکرام توسط اعضای تیم دنبال می‌شوند، هر مسئله و حمله‌ای در برابر چارچوب اسکرام باید توسط اسکرام مستر رفع شود. این شانس خوش و اینچنین اتفاقی به ندرت خواهد افتاد.

### Product Owner

صاحب پروژه یا محصول، مسئول مشخص کردن مشتری و افزایش مقدار کاری است که تیم انجام می‌دهد. او با مشتریان برای مشخص کردن اینکه خواسته آنها چیست، جلسه تشکیل داده و این نیازها را اولویت بندی می‌کند و تیم هم بر روی آیتم‌هایی با بیشترین ارزش برای مشتری، کار خواهد کرد. صاحب محصول همچنین product backlog را مدیریت کرده و تنها شخصی است که میتواند user story را برای انتقال به اسپرینت، اولویت بندی کند. مسئولیت‌های صاحب محصول در طول اسپرینت از سری مسئولیت‌های نقش Pig به سری مسئولیت‌های نقش Chicken تغییر می‌کند. یکی دیگر از نقش‌های حیاتی او این است که به عنوان نماینده مشتری، برای تیم است. صاحب محصول خیلی شبیه به اسکرام مستر می‌باشد ولی در این بین یک تفاوت اصلی در طبیعت نقش‌های آنها وجود دارد: اسکرام مستر به دنبال بهترین جذابیت‌های مورد علاقه تیم در یک اسپرینت بوده؛ در حالی که صاحب محصول به دنبال بهترین جذابیت‌های مطلوب مشتری در یک اسپرینت است. در یک تیم اسکرام، صاحب محصول، نقشی است که نمیتوان آن را دست کم گرفت. اگر صاحب محصول کسی باشد که نتواند به طور دقیق نیازهای مشتری را به تصویر بکشد، در نتیجه پروژه شکست خواهد خورد.

صاحب محصول کلید اجرایی یک محصول است که ارزشی را برای مشتری و موفقیتی را برای تیم به ارمغان می‌آورد.

### Delivery Team

تیم تحویل، گروهی است از افراد که مسئول ارائه واقعی محصول هستند. این تیم معمولاً شامل دو تا ده نفر از افراد و همچنین ترکیبی از برنامه نویسان، تست‌کننده‌ها، طراحان محصول نهایی و اعضای از سایر نظام‌های ضروری، می‌باشد. تیم برای انتقال user story و وظایف مرتبط با آن به مرحله بعد بر روی تخته Kanban، تا مرحله‌ی اتمام، بر روی اسپرینت‌ها کار می‌کند. مشخصه کلیدی تیم تحویل این است که آنها به صورت یک واحد خود سازمانده می‌باشند. هیچ رهبری در جمع آنها وجود ندارد و همه به صورت گروهی تصمیم می‌گیرند که در هر اسپرینت به انجام چه چیزی میتوانند متعهد شوند. اعضای تیم بار دیگر تصمیم خواهند گرفت که چه ابزاری برای موفقیت پروژه نیاز دارند. چنین سطحی از استقلال در متدولوژی آبخاری بی‌سابقه است! تیم تحویل برای بهینه سازی انعطاف پذیری و بهره وری در نظر گرفته شده‌اند. تیم اسکرام ترکیبی از افرادی است با توانمندی‌های گوناگون که هر کدام باید با تمام چشم اندازه‌های محصول در مراتب مختلف آشنا

باشند. هریک از اعضای تیم به تنهایی در همه‌ی مباحث نرم افزار ماهر نیستند، اما هر یک از آنها دانش عمومی در همه مباحث را دارند و در قسمت کلی از مفاهیم محصول هم متخصص هستند. تیم تحویل به همراه اسکرام مستر و صاحب محصول، برای تکمیل user story ها و به سرانجام رساندن هر اسپرینت، باهم کار میکنند. اسکرام مستر با آمادگی به دنبال بهترین جذابیت‌های مورد علاقه تیم در یک اسپرینت است؛ در حالیکه صاحب محصول با آمادگی به دنبال بهترین جذابیت‌های مطلوب مشتری در یک اسپرینت است. با وجود این دو نقش، تیم میتواند محصولی که مشتری میخواهد را بسازد.

### فعالیت‌های اسکرام

شامل فعالیت‌هایی که در مرکز کانونی اسکرام و در سراسر طرح ریزی، بررسی و نشست‌ها و جلسات میباشد.

#### Sprint Planning

قبل از شروع هر sprint، جلسه طرح ریزی برای مشخص کردن اینکه کدام امکان و ویژگی در این sprint قرار بگیرد، برگزار میشود. ویژگی‌ها و امکانات از لیست pb ای (product backlog) که توسط صاحبان (یا صاحب) محصول اولویت بندی شده است، انتخاب خواهند شد. برای بار اول که این نشست و جلسه برای یک پروژه برگزار شود، pb ساخته میشود. شما می‌توانید این قسمت را sprint 0 در نظر بگیرید. user stories (گزارشات کاربر) انتخاب شده توسط صاحب محصول، برای قرار گرفتن در sprint، به تیم داده میشود و آنها از طریق یک ابزار کاری بنام Planning Poker، گزارشات مذکور را برای نشان دادن پیچیدگی یک گزارش وابسته به گزارشات دیگر در گروه گزارشات، تغییر اندازه و تغییر حجم میدهند. بار دیگر user story هایی که به اندازه هستند، توسط تیم به وظیفه‌هایی قابل نسبت دادن به یک فرد تبدیل میشوند و یک زمان تخمینی که نشان دهنده‌ی زمان اتمام برای هر وظیفه است، برای هر وظیفه در نظر گرفته میشود. بار دیگر که تمام این کارها انجام شد، اعضای تیم به لیست کامل کارهایی که برای sprint در نظر گرفته شده‌اند، نگاه خواهند کرد و اگر بتوانند تا اتمام sprint کار را تمام کنند، تصمیم خواهند گرفت که آن را انجام دهند. این تصمیم گیری به صورت زیر است:

به وسیله 5 انگشت قرار است نظرات خود را ارائه دهند؛ به طوری که اگر عضوی دست خود را با یک انگشت بالا ببرد، بدین معنی است که او در طرح پیشنهادی خیلی تردید دارد و اگر دستی با 5 انگشت بالا رود، به این معنی است که این عضو، به شدت به طرح پیشنهادی مطمئن است. اگر هیچ دستی با تعداد انگشت 1 یا 2 از بین دست‌های بالا رفته دیده نشود، لذا تیم به انجام آن کار در sprint جاری متعهد خواهد شد. ولی اگر دستی با تعداد انگشت 1 یا 2 از بین دست‌های بالا رفته دیده شود، در آن صورت اعضای تیم در مورد دلیل رای عضوی که رای با ارزش 1 یا 2 داده است، بحث خواهند کرد و با دیگر اعضای تیم برای تحویل گزارشات و وظایف موجود در اسپرینت، متعهد میشوند.

sprint backlog از گزارشات کاربر و وظایفی که باید در sprint تکمیل شوند، ساخته شده است. تمام اعضای تیم در کنار اسکرام مستر و صاحبان محصول در نشست برنامه ریزی اسپرینت درگیر هستند. بار دیگر جلسه برنامه ریزی متشکل از اعضای تیم و بدون صاحبان محصول برای بحث در مورد طراحی سطح بالای سیستم برگزار خواهد شد.

#### planning poker

planning poker یک بازی است که اعضای تیم را تشویق میکند تا ارزیابی درستی در مورد پیچیدگی گزارش کاربری (user story) که در ارتباط با سایر گزارشات (stories) است، داشته باشند. ابزارهای مورد نیاز برای این بازی خیلی ساده هستند: شما میتوانید از دست خود استفاده کنید؛ یا حتی میتوانید مجموعه کارت‌های Planning Poker را برای انجام بازی، خریداری کنید. برای انجام این بازی، صاحب محصول، گزارش کاربر (user story) را خوانده و برای تیم توضیح خواهد داد. تیم برای پرسیدن سوال در باره این گزارش کاربر، آزاد است. وقتی که تمام سوالات پاسخ داده شدند، اسکرام مستر از اعضای تیم خواهد خواست که یک عدد را به صورت خصوصی که به بهترین شکل پیچیدگی user story را ارائه میکند، تعیین کنید. توجه داشته باشید برای اینکه این انتخاب به صورت سهوی تحت تاثیر انتخاب سایر اعضا نباشد، باید برای دیگران آشکار نشود. بار دیگر اسکرام مستر از همه میخواهد تا شماره‌های خود را برای همه آشکار کنند. اگر تمام اعضای تیم، یک شماره یکسان را تعیین کرده باشند، آن شماره به user story مورد نظر نسبت داده شده و همه سراغ user story بعدی میروند.

اگر شماره‌ها باهم یکسان نباشند، عضوی با بیشترین و کمترین شماره، انتخاب شده و از آنها خواسته میشود دلیل تعیین شماره خود را شرح دهند. بعد از بحث، یک راند دیگر از بازی بین اعضایی که یک شماره را برای user story انتخاب کرده‌اند، انجام میشود. این کار تا زمانی که تیم در یک شماره اتفاق نظر داشته باشند، ادامه خواهد داشت. به طور متوسط برای رسیدن به یک شماره یکسان، بیشتر از 3 راند طول نخواهد کشید. اگر بعد از 3 راند باز هم به شماره‌ای که همه با آن موافق هستند، دست نیابند، ما به اسکرام مستر پیشنهاد میکنیم که میانگین را انتخاب کرده و سراغ user story بعدی بروند.

#### Daily Stand Ups

در طول یک اسپرینت، تیم، اسکرام مستر و صاحب محصول، برای حضور در جلسات روزانه که یکبار در هر روز و در یک مکان و زمان یکسان برای بحث در مورد موضوع‌هایی که موجب مانع از اتمام کار میشوند، متعهد میشوند. در جلساتی که برگزار میشود، همه به صورت ایستاده بوده و زمان آن بیشتر از 15 دقیقه طول نخواهد کشید. هرکسی که به نوعی ذینفع در پروژه هستند، برای

حضور در جلسات دعوت میشوند. هر چند فقط افرادی که رده بندی شده‌اند، اجازه صحبت در این جلسات را خواهند داشت. در این جلسات، هر عضو تیم به 3 سوال زیر پاسخ خواهد داد:

شما چه چیزی را از دیروز تا حالا انجام داده‌اید؟

شما چه برنامه‌ای برای امروز دارید؟

آیا شما مشکل دیگری که مانع رسیدن به هدف‌تان باشد، ندارید؟ چه جریانی باعث ایجاد این موانع شده‌اند؟ آیا میتوان مانع را حذف کرد یا باید تشدید شود؟

### Sprint Review

جلسه "بررسی اسپرینت" در پایان اسپرینت برگزار میشود. هدف از آن ارائه گزارشات کاربری (user stories) هست که در طول اسپرینت تکمیل شده‌اند. تیم، صاحب محصول و اسکرام مستر به همراه سایر ذینفعان، مخصوصا مدیران و مشتریان، در این جلسه حضور خواهند داشت. این بررسی شامل یک دموی غیررسمی از نرم افزار توسعه داده شده در اسپرینت، میباشد. این جلسه دموی محصول، فرصتی است برای مشتری تا بازخوردهای خود از محصول را به تیم توسعه انتقال دهند. هدف اصلی از این بازنگری، نمایش محصول با کارکرد واقعی است. این جلسه با اصل "بالاترین اولویت ما عبارت است از راضی کردن مشتری با تحویل سریع و مداوم نرم افزار با ارزش" چابک در یک راستا میباشد.

## داستان‌های کاربر

توسعه‌دهندگان، ویژگی‌های مورد نظر پروژه را با جمع‌آوری نیازمندی‌ها، در قالب داستانهای کاربر احصاء می‌کنند و به هرکدام متناسب با پیچیدگی‌اش امتیازی اختصاص می‌دهند. با لیستی از داستان‌های دارای ابعادی مشخص و بودجه و زمان مورد نیاز برای هرکدام، مشتریان قادر به این انتخابند که کدام ویژگی‌ها در تکرار (iteration) بعدی باقی بماند. مشخص کردن بودجه و زمان، یعنی تعیین حجم کاری که تیم توسعه برای انجام آن ویژگی، نیاز می‌داند. برآورد بودجه مورد نیاز تکرار اول به صورت تجربی خواهد بود و ممکن است این تخمین در ابتدا نادرست باشد؛ اما با شروع تکرار بعدی درست خواهد شد. در پایان هر تکرار، امتیازات به دست آمده از داستان‌های کامل شده را جمع کنید. مجموع این امتیازات، نشانگر سرعت شما خواهد بود. این سرعت شاخص خوبی جهت چگونگی بودجه‌بندی مرحله بعد است. هنگامیکه امتیازات جمع‌آوری شده به حد مطلوبی رسید، «سرعت پیشروی»، شاخص مناسب دیگری برای بودجه‌بندی است که عبارت است از متوسط سرعت سه تکرار آخر.

با این کار شما به دیدگاه مناسبی از فاز برنامه‌ریزی دست پیدا می‌کنید. حال اجازه دهید نگاه دقیق‌تری به شیوه‌های برنامه‌ریزی داشته باشیم.

برنامه‌ریزی (planning game) دو فاز دارد: فاز شناسایی و فاز برنامه‌ریزی. در فاز شناسایی، توسعه‌دهندگان و مشتریان را دور هم جمع می‌کنند تا درباره نیازمندیهای سیستم در حال طراحی، گفتگو کنند. به خاطر داشته باشید که این کار تا وقتی انجام می‌شود که به ویژگی‌هایی (features) کافی برای شروع انجام کار برسیم و البته واضح است که چنین لیستی از ویژگی‌های احصاء شده، هرچقدر هم که تلاش شود، کامل نخواهد بود. مشتریان اغلب اوقات، خواسته‌ی خود را یا نمی‌دانند یا نمی‌توانند به خوبی توضیح دهند. بنابراین معمولاً این لیست به مرور تغییر می‌کند. در ضمن آنکه برخی ویژگیها دقیق‌تر می‌شود، مواردی نیز ممکن است به لیست افزوده شوند یا حتی می‌توان برخی ویژگی‌های نامربوط را از لیست حذف کرد. در مرحله شناسایی، ویژگی‌ها به داستانهای کاربر تجزیه شد و ثبت می‌شوند.

یک داستان کاربر عبارت است از توصیفی کوتاه از یک ویژگی که نمایانگر یک واحد ارزش کسب و کار برای مشتری است. داستانهای کاربر از زبان کاربر بیان شده‌اند و قالب نوشتاری زیر را دارند:

به عنوان «نوع کاربر»، من می‌خواهم «یک فعل» تا «منفعتی برای کسب و کار»

یا به صورت:

به منظور «یک دلیل» به عنوان «نقش کاربر» من می‌خواهم «یک فعل»

داستانهای کاربر معمولاً در جلسه‌ی گفتگو با مشتری بر روی کارت‌های راهنما نوشته شده و در آن از واژگان و ادبیاتی استفاده می‌شود که برای مشتری قابل فهم باشد. ممکن است چنین بیانیدیشید که ثبت نیازمندی‌ها، خلاف مزیت‌های چابک‌سازی است؛ چرا که تولید نرم‌افزار کارآمد و چابک مبتنی بر مستندسازی گسترده و فراگیر خواهد بود. در واقع، داستان‌های کاربر به طور ساده فقط یادآورنده جزئیات بیشتری از گفتگوی انجام شده‌اند که به عمد به صورت کوتاه و دقیق نوشته شده‌اند. فهم دقیق‌تر جزئیات کار، مستلزم ارتباط بیشتر میان توسعه‌دهندگان و مشتری است. در واقع همسو با این اصل چابک که می‌گوید: «مؤثرترین و کارآمدترین شیوه انتقال اطلاعات در میان تیم توسعه و به خارج از آن، گفتگوی چهره به چهره است.»

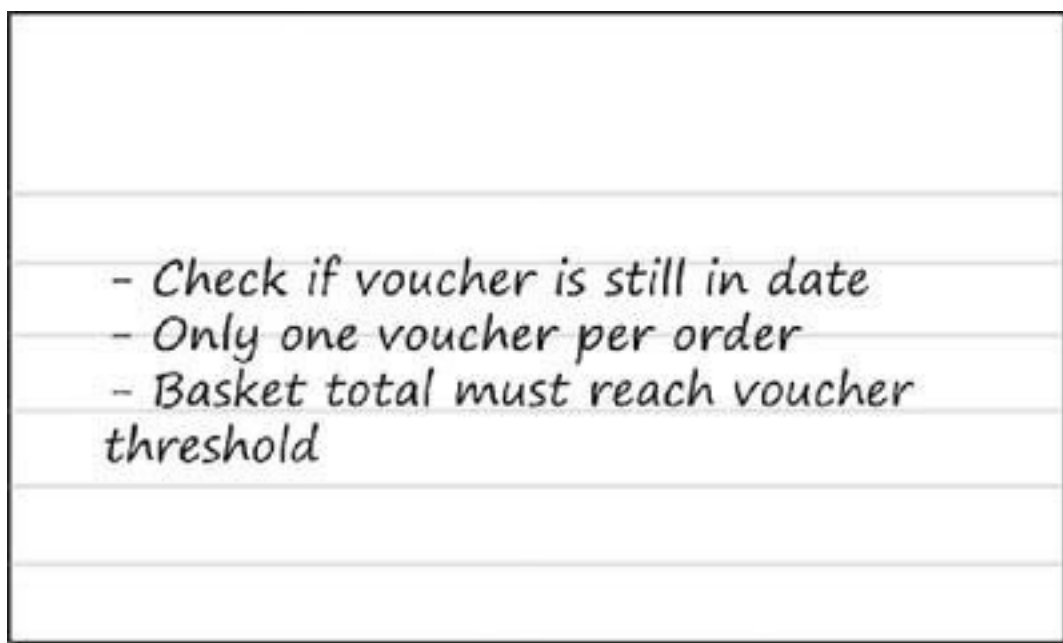
هنگام احصاء ویژگی‌های پروژه تحت عنوان داستان‌های کاربری، از اصول INVEST (که پیش‌تر گفته شد) جهت کنترل مناسب بودن این داستانها استفاده کنید. شکل 2-3 مثالی از یک داستان کاربر را که توصیف‌کننده ویژگی «افزودن یک بن تخفیف به سبد خرید»



است، نشان می‌دهد. «تخفیف گرفتن»، یک منفعت کسب و کار است برای عامل (actor) اصلی، یعنی مشتری. «یک بن تخفیف به سبد بیفزاید» نام فرآیند یا «use case» مربوط است.



از معیار پذیرش (acceptance criteria) نیز می‌توان در هنگام تولید داستان‌ها استفاده کرد. معیار پذیرش را می‌توان در پشت کارت داستان، آن طور که در شکل 3-3 نشان داده شده است، نوشت. استفاده از طرف مقابل کارت این اجازه را می‌دهد که اعضای تیم و مشتریان، اطلاعات خودشان را در یک جا جمع کنند.



معیار پذیرش همچنین به تشخیص جزئیات بیشتر یا شناسایی وابستگی‌ها کمک می‌کند. مثلاً در شکل 3-3 تعریف «in date» چیست و چه چیزی حدود یک بن تخفیف را مشخص می‌کند؟ معمولاً باید حداقل سه معیار پذیرش وجود داشته باشد. در فصل بعد در یک مطالعه موردی، مطالب بیشتری را دربارهٔ داستانهای کاربر خواهید آموخت.

هنگامیکه تیم و مشتریان حس کنند که حدود 75 درصد از ویژگی‌های اصلی احصاء شده است، توسعه‌دهندگان ابعاد داستان‌ها را تخمین زده و آنها را برای اولویت‌بندی توسط مشتری آماده می‌کنند.

## تخمین

شکی در آن نیست که تخمین‌زدن کار سختی است. تخمین‌زدن هم دانش است هم هنر. تخمین‌زدن در یک پروژه تازه شروع شده، بسیار سخت است زیرا مجهولات بسیاری در آن وجود دارد.

یکی از روش‌های تخمین گروهی، روش «Planning Poker» نام دارد. در این روش همه‌ی اعضای فنی تیم، متشکل از توسعه‌دهندگان نرم‌افزار، تحلیل‌گران، متخصصان امنیت و زیرساخت، مشارکت می‌کنند. نقش مشتری در این حالت پاسخ‌گویی به سؤالات احتمالی اعضای تیم است تا ایشان بهتر بتوانند تخمین بزنند.

شیوهٔ انجام کار به این صورت است که عضوی از تیم، یک داستان کاربر را برداشته و آن را برای تیم توضیح می‌دهد. تیم دربارهٔ آن ویژگی با مشتری گفتگو کرده تا جزئیات بیشتری را دریابد. وقتی که تیم به درک خوبی از آن رسید، رأی‌گیری آغاز می‌شود. هر عضو تیم با یک کارت، از مجموعه‌ای از کارتهایی با شماره‌های 0، 1، 2، 3، 5، 8، 13، 20، 40 و 100 رأی خود را اعلام می‌کند.

تیم باید از داستانی شروع کند که نسبتاً کوچک و ساده باشد. این داستان به عنوان مبنا انتخاب می‌شود. هر تخمین داستان کاربر، باید به نسبت این داستان کوچک انجام شود. اگر داستان مبنا به خوبی انتخاب نشود، بقیهٔ تخمین‌ها نادرست خواهد بود.

اگر همه‌ی اعضای تیم به یک صورت رأی دهند، آن رأی، تخمین آن داستان خواهد شد. اگر اختلاف آراء وجود داشت، ناظر یعنی کسی که رأی نمی‌دهد، از افرادی که بالاترین و پایین‌ترین امتیاز را داده‌اند، می‌خواهد که علل خود را توضیح دهند. سپس تیم مجدداً گفتگو کرده و دوباره رأی‌گیری می‌کند. طبق تجربه، خوب است که زمان معقولی، برای هر گفتگو در نظر گرفته شود.

اگر تخمین یک داستان به دلیل فقدان دانش فنی، بسیار سخت بود، مناسب است که این داستان کنار گذاشته شود و داستان دیگری برای برطرف کردن مشکل ناآشنایی با دانش فنی مورد نظر فراهم شود. بدین ترتیب تیم توسعه در موقعیت بهتری می‌تواند نسبت به داستان جدید تخمین بزند.

داستان‌هایی که بیش از یک هفته کار نیاز داشته باشند با عنوان داستانهای حماسی (epic stories) شناخته می‌شوند و معمولاً برای تخمین بسیار بزرگ هستند. در واقع، این داستان‌ها به چند داستان کوچک‌تر که قابل فهم‌تر و به آسانی قابل تخمین باشند، تجزیه می‌شوند. این بدان معناست که ایجاد یک داستان کاربر از تعداد انبوهی ویژگی موجب کاهش کارایی خواهد شد.

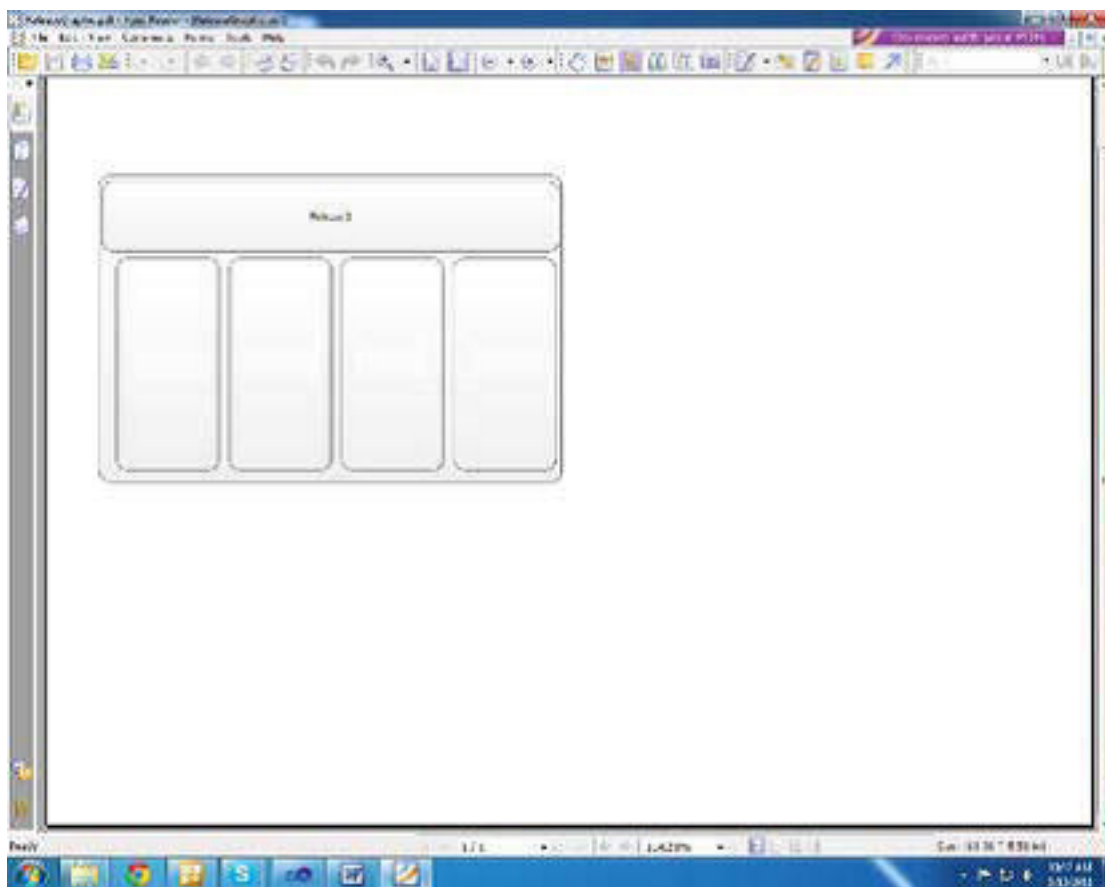
تخمین در تیمی که افراد آن تاکنون با همدیگر سابقهٔ همکاری نداشته باشند، خیلی پایین یا خیلی بالاست. اما با استمرار هر تکرار و تجربه و دانش بیشتر افراد، تخمین داستان‌ها بهتر می‌شود.

استفاده از ابزار Planning Poker مزایای بسیاری دربردارد. دقت تخمین بالا می‌رود؛ زیرا مسأله از منظر تخصص‌های گوناگون مورد بررسی قرار گرفته است. همچنین به تیم کمک می‌کند که هم رأی شوند و گفتگو میان اعضا را تسهیل می‌کند. پس از آنکه داستان‌ها تخمین زده شدند، مشتری و صاحب محصول با تیم توسعه در تولید چگونگی انتشار نسخه‌ها، همکاری می‌کنند.

## برنامه انتشار

اگرچه کدهای قابل ارسال، قابلیت انتشار در پایان هر تکرار را دارند، اما یک پروژه XP در چند سری منتشر شده است. یک نسخهٔ منتشرشده، متشکل از تعداد مناسبی داستان برای عرضهٔ ارزش کسب و کاری است که به کوچک نگه داشتن آن کمک می‌کند.

بسیار مناسب است که یک موضوع یا هدف خاص را در ضمن هر نسخه انتشار، مد نظر قرار داد تا کمک کند که هر نسخه انتشار بر برخی ارزشهای کسب و کاری متمرکز شده و آن را هدایت کند. معمولاً یک نسخه انتشار، متشکل از چهار تکرار است؛ همانطور که در شکل 3-4 نشان داده شده است.



در برنامه ریزی نسخه های انتشار، طول یک تکرار نیز تعیین می شود که معمولاً بین دو تا چهار هفته است. مطابق تجربه، اگر محیط کار شما دچار بی نظمی و اختلالات دائمی است، می توانید دوره تکرار را به یک هفته محدود کنید.

یکی از پروژه هایی که ما بر روی آن کار می کردیم، برنامه ای بود که نگهداری آن بسیار سخت و فوق العاده ناپایدار بود. مشتری مکرراً با تیم تماس گرفته و اشکالات بحران ساز و ایراداتی را که مخل برنامه بودند، گزارش می کرد. در ابتدای کار دوره، تکرار ما هفتگی بود. به همین دلیل چون حلقه بازخوردگیری مان کوچک بود، می توانستیم بر پایدارسازی پروژه در هر دوره کاری تمرکز کنیم. هنگامی که محصول به پایداری مناسب تری رسید و تماس های مشتری کم شد، قادر شدیم تا در هر دوره، دقت بیشتری بر روی مسائل به خرج دهیم.

اگر قصد دارید به صورت دقیق بر روی حلقه بازخورد متمرکز شوید، دوره ی تکرار یک هفته ای، مدل خوبی است. اما این مدل سربرای زیادی را به دلیل ضرورت تقسیم داستانهای کاربر باید به بخش های کوچک تری تا آن اندازه که در یک دوره تکمیل شوند، بر پروژه تحمیل می کند. در ادامه خواهیم گفت که هر تکرار شامل برنامه ملاقات و بازبینی نیز هست.

بعد از مدتی که تیم با فرآیند کار آشنا تر شد و نوبت به مشکلات با اولویت کم تر رسید، می توان دوره تکرار را دو هفته ای در نظر گرفت. اما اگر پروژه به گونه ای است که ویژگی های بزرگ تر را نمی توان به موارد کوچک تری که قابل انجام در دوره های یک هفته ای باشد، تجزیه کرد و تیم هنوز در حال یادگیری است، دوره های بلند مدت تر قابل پذیرش است.

مشتری با توجه به طول دوره تکرار و بودجه داستان آغازین، انتخاب می‌کند که کدام داستان در هنگام انتشار نسخه اول، در تکرار اول کامل شود.

این مشتری است که داستان‌ها را به گونه‌ای اولویت‌بندی می‌کند تا مشخص شود که کدام یک بیشترین ارزش کسب و کار را فراهم می‌کند. از آنجایی که مشتری مسئول داستانهای کاربر است، تیم باید به وی توضیح دهد که داستانهایی وجود دارند که صرفاً باید به جهت دلایل فنی ایجاد شوند.

معمولاً باید به داستانهای کاربری‌ای که مستلزم ریسک بالا بوده یا دربرگیرنده مجهولات زیادی باشند، بیش از یک یا دو تکرار اختصاص داد.

## برنامه تکرار

مشتری داستان‌هایی را که می‌خواهد در تکرار باشند، انتخاب می‌کند. برای هر داستان کاربر، مجموعه‌ای از معیارهای پذیرش، تعریف شده است. همان طور که متوجه شده‌اید ما در هر فاز، وقت بیشتر و بیشتری را صرف جمع‌آوری جزئیات هر داستان کاربر کرده و بصورت عمیق‌تری در آن غور می‌کنیم. این کار مفید است، زیرا اگر یک داستان کاربر ایجاد شده در ابتدای پروژه، ممکن است بعداً به عنوان داستانی کم اهمیت یا غیر مهم دیده‌شود و بدون آنکه وقت خاصی برای آن صرف شده باشد، کنار گذاشته شود. اما اگر در ابتدای کار وقت زیادی صرف دقیق‌تر کردن داستان‌های کاربر شود و بعداً بعضی از آنها کنار گذاشته شوند، در واقع وقت تلف شده است. بنابراین دقیق‌تر کردن یک داستان در جایی که مورد نیاز است، باید اتفاق بیفتد. در سطح برنامه تکرار، مجموعه‌ای از معیارهای پذیرش را برای هر داستان کاربر تعریف می‌کنیم. معیار پذیرش به توسعه‌دهنده کمک می‌کند تا بداند که یک داستان کاربر به طور کامل انجام می‌شود. این معیارها به صورت مؤلفه‌هایی از بافرض/هنگامی که/درنتیجه، نوشته می‌شود.

مثالهای زیر چگونگی انجام این کار را توصیف می‌کند:

### عنوان ویژگی : افزودن کالایی به سبد

به عنوان یک مشتری می‌خواهم بتوانم کالایی را به سبدم اضافه کنم؛ به نحوی که قادر باشم به خرید خود ادامه دهم.

### سناریو : سبد خالی

با فرض اینکه یک سبد خالی دارم، در نتیجه جمع تعداد کالایی که برای سفارش در سبد من وجود دارد، صفر است.

### سناریو : افزودن یک کالا به سبد

با فرض اینکه یک سبد خالی دارم هنگامی که کالایی با شناسه 1 به سبدم اضافه می‌کنم، در نتیجه جمع کالاهای قابل سفارش در سبد 1 می‌شود.

### سناریو : افزودن کالاهایی به سبد

با فرض اینکه یک سبد خالی دارم، هنگامی که کالایی با شناسه 1 و کالایی با شناسه 2 به سبدم اضافه می‌کنم، در نتیجه جمع کالاهای قابل سفارش در سبد 2 می‌شود.

### سناریو : دو بار افزودن یک کالا

با فرض اینکه یک سبد خالی دارم هنگامی که کالایی با شناسه 1 به سبدم اضافه می‌کنم و هنگامی که کالایی با شناسه 1 را مجدداً به سبدم اضافه می‌کنم، در نتیجه تعداد کالاهای با شناسه 1 در سبد من باید 2 باشد.

**سناریو : افزودن یک کالای تمام شده به سبد**

با فرض اینکه یک سبد خالی داریم و کالایی با شناسه 2 در انبار وجود نداشته باشد، هنگامی که من کالایی با شناسه 2 را به سبد خودم اضافه می‌کنم، در نتیجه جمع تعداد کالای قابل سفارش در سبد من باید 0 باشد و به کاربر، موجود نبودن آن کالا را هشدار دهد.

یک آزمون پذیرش (acceptance) به زبان متعارف در قوانین کسب و کار نوشته می‌شود. در مثال سبد خرید، این سؤال پیش می‌آید که چگونه می‌توان یک محصول را از سبد کالا، حذف کرد و اگر یک جنس اکنون در انبار نیست و کاربر پیام هشدار دریافت کرده است، در ادامه چه اتفاقی باید بیفتد؟ سناریوها به تیم در کشف ملزومات کسب و کار و تصریح آن‌ها کمک می‌کند.

این سناریوها توسط توسعه‌دهنده به عنوان نقطه شروع آزمونهای واحد در توسعه آزمون محور و رفتار محور استفاده می‌شود. سناریوها همچنین در آزمون معیارهای پذیرش به توسعه‌دهنده کمک کرده و توسعه‌دهنده و تست‌کننده را قادر می‌سازند که بر روی اتمام داستان اتفاق نظر داشته باشند.

بعد از آنکه سناریوهای معیار پذیرش تعیین شد، تیم توسعه، هر داستان را به تعدادی وظیفه تقسیم می‌کند و وظایف مرتبط به یک داستان، در تابلوی وظایف قرار گرفته و تیم توسعه تخمین‌های خود را در قالب یکی از واحدهای اندازه‌گیری، مثلاً نفرساعت اعلام می‌کند. شکل 3-5 یک تابلوی وظیفه را نمایش می‌دهد.

به عنوان مثال وظایف می‌توانند شامل ایجاد طرح یک بانک اطلاعاتی برای یک داستان یا یکپارچه‌سازی آن با بخشی موجود در سیستم باشند. وظایف شامل مؤلفه‌های فنی مانند تهیه گزارش از زیرسیستم‌ها یا چارچوب مدیریت استثنائات نیز می‌باشد. اغلب این‌گونه وظایف نادیده گرفته می‌شود. یک داستان کاربر با وظایف گوناگونی گره خورده است. مثلاً:

**داستان کاربر :** به عنوان یک کاربر می‌خواهم بتوانیم یک کاربر را مدیریت کنیم.

وظایف زیر از این داستان قابل استخراج است:

طرحی برای بانک اطلاعات جهت ذخیره‌سازی اطلاعات کاربر ایجاد کن.

یک کلاس کاربر، برای مدیریت کاربر از درون برنامه ایجاد کن.

هر عضو تیم می‌تواند بر روی هر وظیفه‌ای که بر روی تخته است، کار کند. هنگامیکه یک عضو گروه، وظیفه‌ای را برمی‌دارد، باید نشانی از خود روی کارت آن وظیفه قرار دهد (مثلاً حروف اوّل اسمش) تا بقیه افراد بدانند که وی بر روی آن وظیفه، مشغول به کار است. معمولاً اما نه همیشه، یک توسعه‌دهنده همه وظایف مربوط به یک داستان را برمی‌دارد. این کار بدین معناست که آن توسعه‌دهنده با پشتیبانی تیم، مسئول اتمام آن کار است.

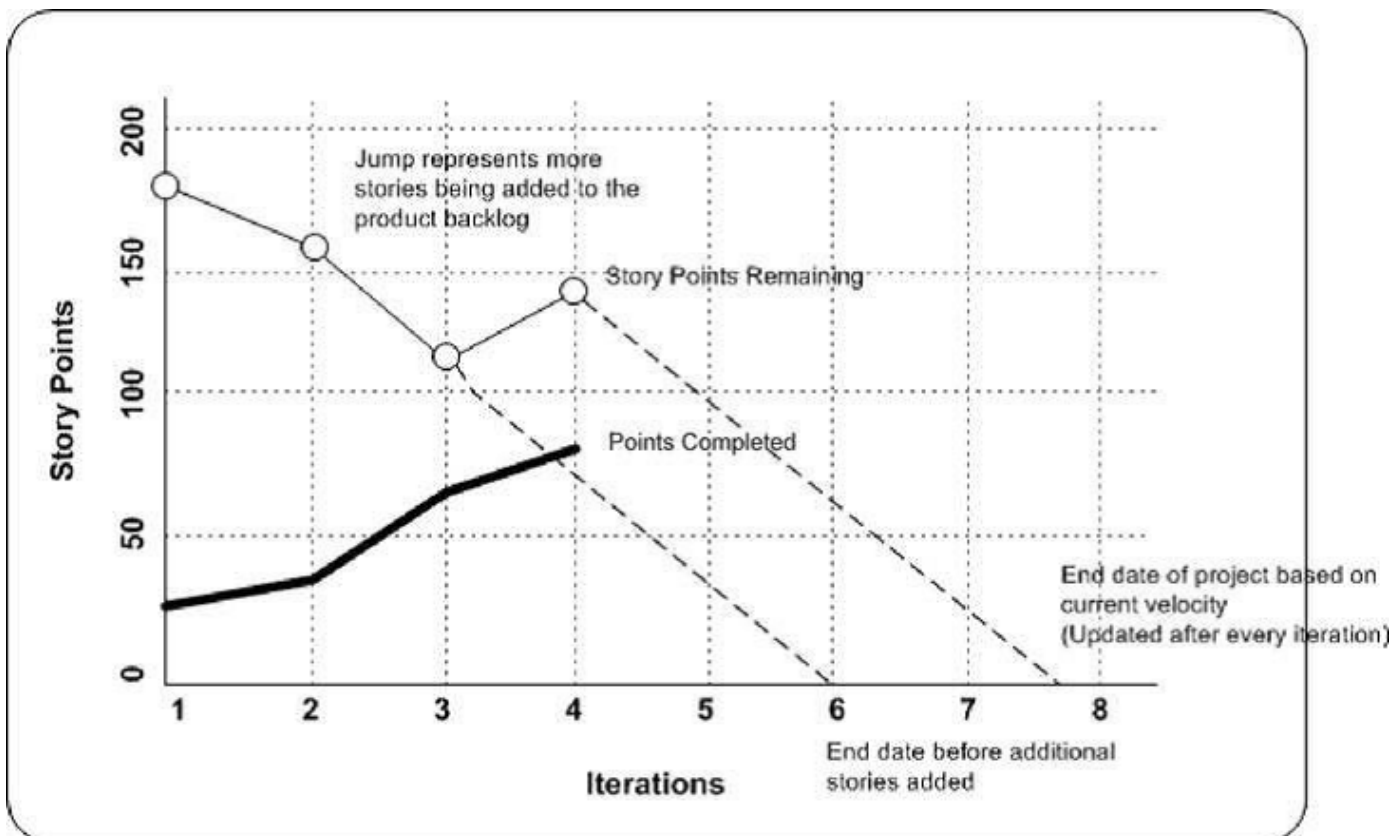
| Stories   | Task Back log  |  | In Process   | In Testing   | Done   |
|---|--|--|--|--|--|
| <b>Basket Discount Vouchers</b><br>In order to receive a discount on my order As a Customer I want to be able to add a voucher to my basket | Asadikljij ij I lujoua oijik<br>Uoiuoua jkondfghdij kjkdj dfgfdg n khskjdjhfkhskej safadkjh ikjh ilf juiz ikjklj sfjnsadfk fjioyjk k ; | Asadikljij ij I lujoua oijik<br>Uoiuoua jkondfghdij kjkdj dfgfdg n khskjdjhfkhskej safadkjh ikjh ilf juiz ikjklj sfjnsadfk fjioyjk k ; | Asadikljij ij I lujoua oijik<br>Uoiuoua jkondfghdij kjkdj dfgfdg n khskjdjhfkhskej safadkjh ikjh ilf juiz ikjklj sfjnsadfk fjioyjk k ; |  |  |
| <b>Loyalty Points</b><br>In order to receive free gifts As a Customer I want loyalty points for orders I place                              | Asadikljij ij I lujoua oijik<br>Uoiuoua jkondfghdij kjkdj dfgfdg n khskjdjhfkhskej safadkjh ikjh ilf juiz ikjklj sfjnsadfk fjioyjk k ; | Asadikljij ij I lujoua oijik<br>Uoiuoua jkondfghdij kjkdj dfgfdg n khskjdjhfkhskej safadkjh ikjh ilf juiz ikjklj sfjnsadfk fjioyjk k ; |  | Asadikljij ij I lujoua oijik<br>Uoiuoua jkondfghdij kjkdj dfgfdg n khskjdjhfkhskej safadkjh ikjh ilf juiz ikjklj sfjnsadfk fjioyjk k ; | Asadikljij ij I lujoua oijik<br>Uoiuoua jkondfghdij kjkdj dfgfdg n khskjdjhfkhskej safadkjh ikjh ilf juiz ikjklj sfjnsadfk fjioyjk k ; |
|   | Asadikljij ij I lujoua oijik<br>Uoiuoua jkondfghdij kjkdj dfgfdg n khskjdjhfkhskej safadkjh ikjh ilf juiz ikjklj sfjnsadfk fjioyjk k ; |  |  |  |  |

به محض اینکه یک تکرار آغاز شد، داستان‌هایی را که کار بر روی آنها شروع شده است، دیگر نمی‌توان تغییر داد. این مهم است که برنامه تکرار را در حین انجام آن، تغییر ندهید؛ زیرا این کار منجر به سوئیچینگ زمینه (context switching) می‌شود. برای توسعه‌دهندگان سوئیچینگ زمینه، هم به لحاظ زمانی و هم به لحاظ مالی، بسیار پرهزینه است.

به جای آنکه تلاش کنید در ضمن یک تکرار، تغییراتی را ایجاد کنید، مشخص کنید که آیا این کار اضافه، یا داستان اضافه را می‌توان تا تکرار بعدی به تعویق انداخت. مشتریان یا مدیران معمولاً می‌توانند چنین تعویقی را بپذیرند؛ زیرا این پذیرش مستلزم به تأخیر انداختن کار، مثلاً تا یک ماه دیگر نیست. اگر این کار جدید را که اضافه شده است، نمی‌توان به تعویق انداخت، باید ریسک خارج ساختن کدهای موجود و رفتن به سمت کدنویسی برای کارکرد جدید را به همراه تیم بررسی کرد. همچنین تیم باید بداند که اگر کار اضافه‌ای به یک دوره تکرار افزوده شد، بخشی از کارهای این دوره باید به تکرار بعدی موکول شوند. قاعده کلی این است که اگر چیزهای جدیدی به کار وارد شد و تعویق آن ممکن نبود، باید کارهایی با همان ابعاد یا بزرگتر از تکرار، خارج شود.

سرعت به ما نشان می‌دهد که تیم چه حجم کاری را در طول یک دوره کامل کرده است. از سرعت، در برنامه‌ریزی تکرارهای آتی استفاده می‌شود. برای درک چگونگی سرعت کار از نمودار burn-down استفاده می‌شود. یک نمودار burn-down (شکل 3-6) داستان‌های باقی‌مانده یک پروژه و داستانهای تکمیل شده را در یک تکرار نمایش می‌دهد. سرعت در پایان هر تکرار محاسبه می‌شود و تعریف آن عبارت است از تعداد داستانهای تکمیل شده در آخرین تکرار. بر اساس سرعت کنونی و تعداد داستانهای باقی‌مانده، می‌توان تخمین زد که چقدر طول می‌کشد تا همه‌ی داستانها تکمیل شود. همانند آنچه در شکل 3-6 با خط چین نمایش داده شده است.

نمودار burn-down ابزار خوبی برای فهم آن است که آیا تیم می‌تواند پروژه را در زمان مقتضی به پایان برساند یا خیر و اگر نمی‌تواند، مدیر چگونه باید نسبت به آن تصمیم‌گیری کند. آیا افراد بیشتری باید به پروژه وارد شوند؟ آیا باید از ویژگی‌های مدنظر پروژه کاهش داد، یا باید زمان پایان کار را تغییر داد؟



در طول یک تکرار، هر روز باید گفتگوهای سرپایی با حضور همه اعضای تیم انجام شود و مشکلاتی که ممکن است باعث به تأخیر افتادن ارائه کار شود، مورد بحث و بررسی قرار گیرد و همچنین تیم، لیست وظایف و تخته آن را به روز کرده تا پیشرفت یا موانع آن به وضوح قابل رؤیت باشند.

با تشکر از آقای [سید مجتبی حسینی](#)