عنوان: **Pro Agile .NET Development With Scrum - قسمت اول**
نویسنده: غلامرضا ربال
تاریخ: ۱۳۹۴/۰۵/۱۷ ۹:۳۵
آدرس: www.dotnettips.info
گروه‌ها: Agile, Scrum

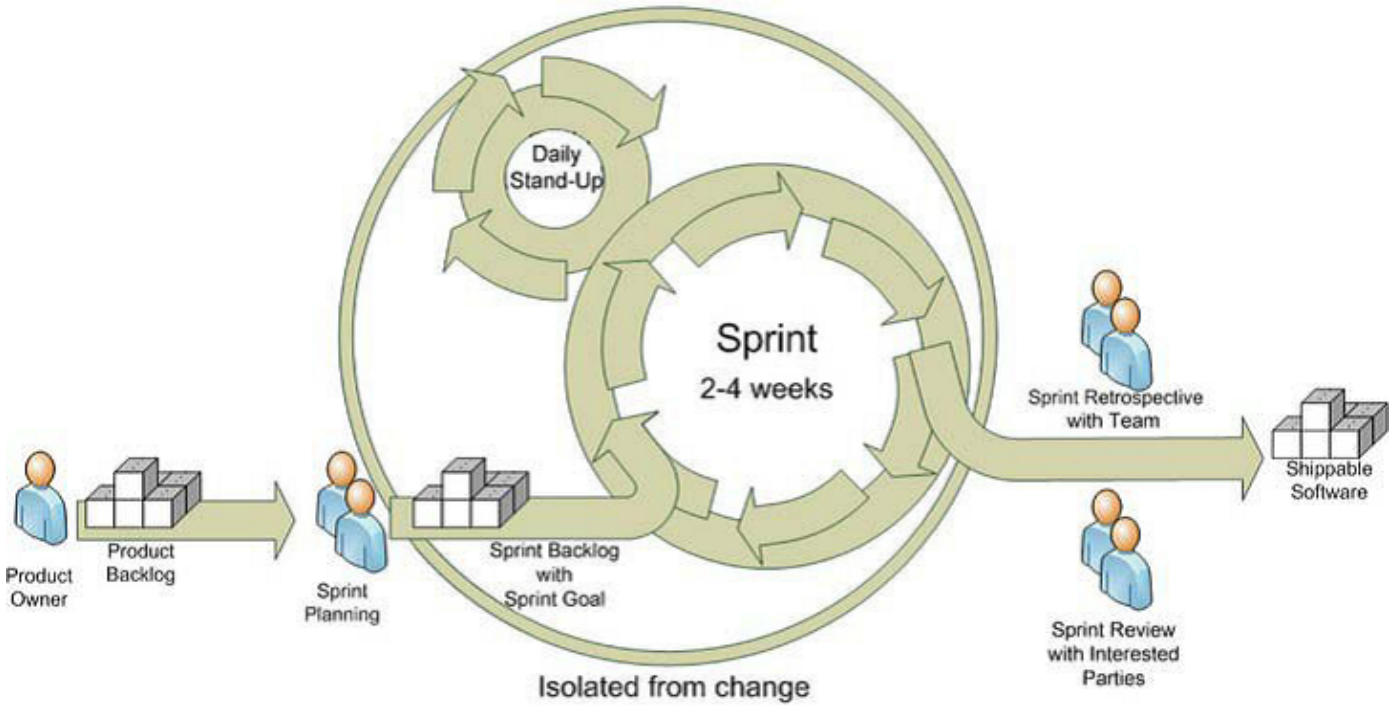
با همکاری آقایان سید مجتبی حسینی و محمد شریفی طی یک سری مقالات سریالی قصد داریم ترجمه آزادی از کتاب Pro Agile .NET Development With Scrum نوشته Jerrel Blankenship و Matthew Bussa ، داشته باشیم.
با توجه به اینکه در سایت جاری مطالب قسمت اول کتاب پوشش داده شده است، ما هم دوباره کاری نکرده و میتوانید از 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 دربردارندهٔ مجموعه‌ای از نقشها و فعالیتهایی است که همگی بر پشتیبانی از تیم مسؤول تولید محصول، تمرکز می‌کنند.
مطالعهٔ موردی بخش دوم این کتاب از شیوهٔ 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 توسعهٔ نرم‌افزار است که بصورت تنگاتنگی با اصول و بیانیهٔ چابک هم‌سو شده است. اسکرام از دنباله‌ای از بلاکهای زمانی به نام اسپرینت ساخته شده است که بر ارائهٔ محصولات کارآمد تمرکز می‌کند. یک اسپرینت نوعاً از دو تا چهار هفته به طول می‌انجامد و با هدف یا موضوعی که واضح کنندهٔ اسپرینت است، تعریف شده است.
اسپرینتها نسبت به تغییرات ایزوله شده‌اند و بدون هیچ اختلالی، تیم توسعه را بر ارائهٔ 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 ارائهٔ ویژگیهایی که تیم توسعه مد نظر قرار داده، نشود. جلسات سرپایی روزانه به تیم کمک می‌کند تا دربارهٔ 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 همهٔ نقشها و فعالیتها و خروجیهای اسکرام بوده که در ادامه، بیشتر دربارهٔ آنها خواهید خواند.



شیوه‌های برنامه محور در مقابل شیوه‌های ارزش محور

هنگام ملاحظهٔ تفاوت میان شیوهٔ آبشاری و شیوهٔ چابک، نیاز است تا به هستهٔ مرکزی هر روش نگریست. یکی از شیوه‌ها از نقشه‌ای برگرفته شده که در ابتدای پروژه ایجاد شده است و شیوهٔ دیگر از ارزشی برگرفته شده که شما به مشتری می‌دهید.

شیوهٔ آبشاری (برنامه محور)

به شیوهٔ آبشاری می‌توان به منزلهٔ شیوه‌ای برنامه محور در توسعهٔ نرم‌افزار نگریست. در گذشته، این شیوهٔ توسعه بسیار مورد استفاده بود، نه به این دلیل که بهترین شیوهٔ توسعهٔ نرم‌افزار بود، بلکه به این دلیل که تنها شیوهٔ شناخته شده بود.

پروژه‌ای که شیوهٔ آبشاری 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 بازهٔ کاری قبل از آنکه حتی یک خط کد نوشته شود، جمع‌آوری می‌شد. مشتریان باید همهٔ 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 همهٔ نکات جزئی کار را مدنظر قرار نداده بود و راهی برای تغییرات مورد نیاز وجود نداشت. گاهی انجام تغییر مستلزم هزینهٔ بسیار بالایی بود. در این گونه پروژه‌ها دامنهٔ پروژه دچار تغییرات می‌شد؛ توسعه‌دهنده از مسائلی که مشتری درصدد حل آنها بود سردرنمی‌آورد و به همین ترتیب مشتری.

توسعهٔ برنامه محور به مانند روند پرش حلقه‌ای است: شما ابتدا جستجو می‌کنید و یک مرتبه از میان آن حلقه پریده و وارد حلقهٔ جمع‌آوری نیازمندیها می‌شوید و از آنجا وارد حلقهٔ طراحی می‌شوید. از یک حلقه نمی‌توانید عبور کنید مگر اینکه از حلقهٔ 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 آبشاری فراهم کنندهٔ 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 هردو.

شیوهٔ اسکرام (ارزش محور)

اسکرام به عنوان شیوه‌ای ارزش محور در توسعهٔ نرم‌افزار مورد توجه قرار می‌گیرد. اسکرام به چند دلیل تغییر چشم‌گیری نسبت به شیوهٔ آبشاری داشته‌است. اسکرام به جای آنکه در ابتدا به جمع‌آوری نیازمندیهای مورد نیاز برای هر ویژگی مد نظر پروژه بپردازد و به جای آنکه همهٔ طراحی‌های خود را مبتنی بر این نیازمندیها کامل کرده و سپس به کدنویسی برنامه مبتنی بر این طرحهای از اول مشخص شده بپردازد؛ به توسعهٔ تکرارپذیر و افزایشی می‌نگرد.

اسکرام تماماً معطوف به مسیرهایی 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 پروژهٔ آبشاری کوچک نگریست. زیرا در هر اسپرینت شما همهٔ کارهایی را که به طور عادی در یک پروژهٔ آبشاری 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 دست‌آمده پرداخته و سپس کدنویسی کرده و آن خصیصه را با توجه به طراحی صورت گرفته، تست می‌کنید. شما در اسکرام برخلاف روش آبشاری، 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 پروژهٔ 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 پروژهٔ آبشاری قابل انجام بود؟ جواب، انجام اسپرینتهایی با ویژگیهای کوچک است. ویژگیهای جزئی، قطعاتی از پروژه هستند که تلاش می‌کنند مسألهٔ 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. نگریستن 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.

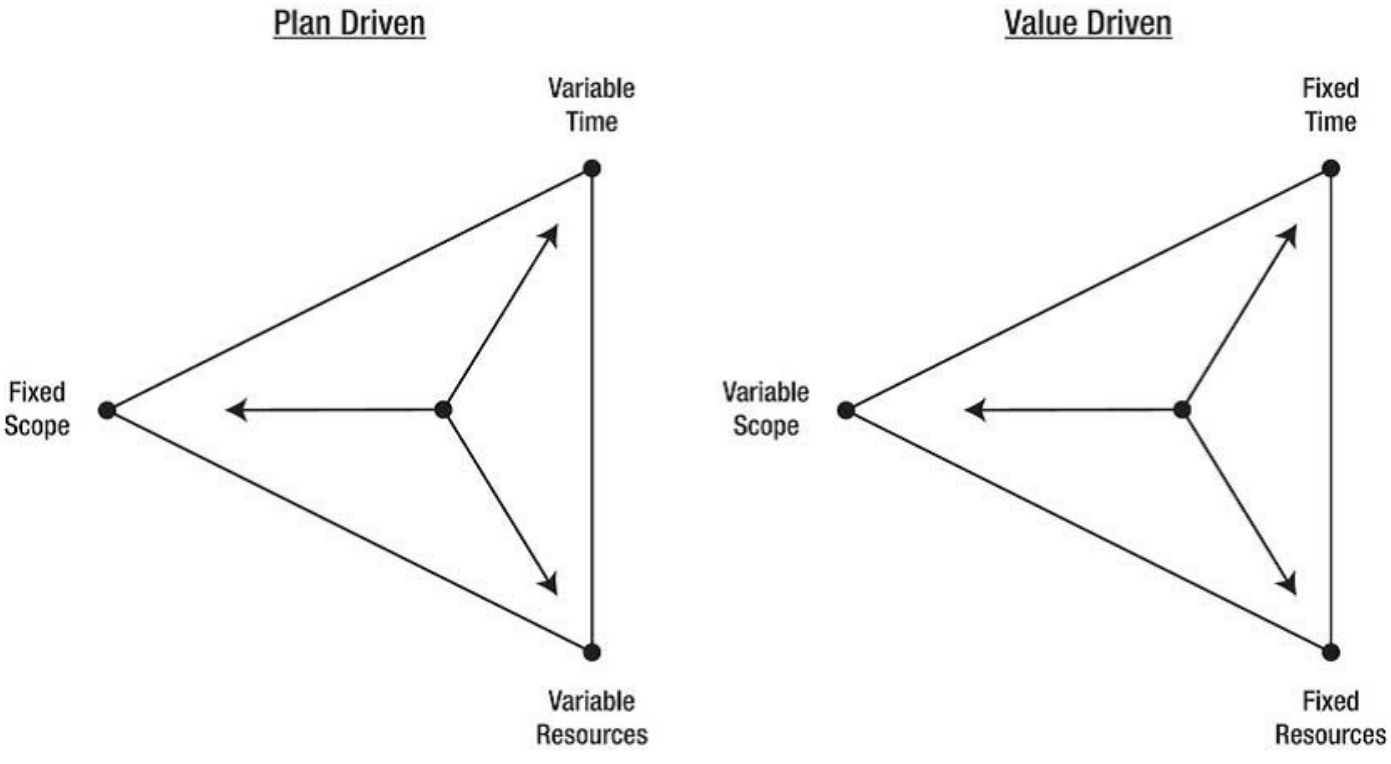
با انجام این کار در سطوح جزئی، مسائل این سطوح جزئی تا حدی شبیه مسائلی هستند که در انتهای پروژه در شیوهٔ آبشاری رخ می‌دهند. در واقع اسکرام به گونه‌ای کار می‌کند که بتواند تا آنجا که ممکن است سریعاً مشکلات و مسائل را نشان دهد. مشکلات قابل پنهان شدن نیستند؛ چراکه پروژه به سطوحی کوچک و قابل مدیریت، تجزیه 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.

نکتهٔ بسیارمهمی را باید دربارهٔ 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 دهنده، 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 پروژهٔ نرم‌افزاری وجود دارند: زمان، منابع و محدودهٔ پروژه. متأسفانه در یک زمان، هر سه عامل قابل جمع نیست. طبق شکل مثلثی 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 توسعهٔ برنامه محور، حیطه و منابع پروژه، معمولاً عوامل ثابتند و زمان عامل متغیر است. در این حالت حیطهٔ پروژه بر منابع و زمان حاکم است. این حالت تا زمانی خوب است که شما در میانهٔ پروژه قراردارید. اما به مرور، رشد حیطهٔ پروژه، چهره نامطلوب کار را نمایان می‌سازد. در این هنگام محدودهٔ 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 نتیجهٔ مناسبی در انتهای کار دست یابید.
در مدل توسعهٔ ارزش محور، منابع و زمان در مثلث ثابتند. شما از ابعاد تیمتان و سرعت انجام کارشان در اسپرینتهای قبلی آگاهید. در این حالت محدودهٔ پروژه در مثلث فوق، عنصر متغیر می‌شود. به عبارت دیگر منابع پروژه و زمان، تعیین‌کنندهٔ محدودهٔ 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 دربارهٔ کل زمان انجام کار، زمان تخمین‌زده شده، مقدار کارانجام شده و عقب‌ماندگی‌های پروژه است.

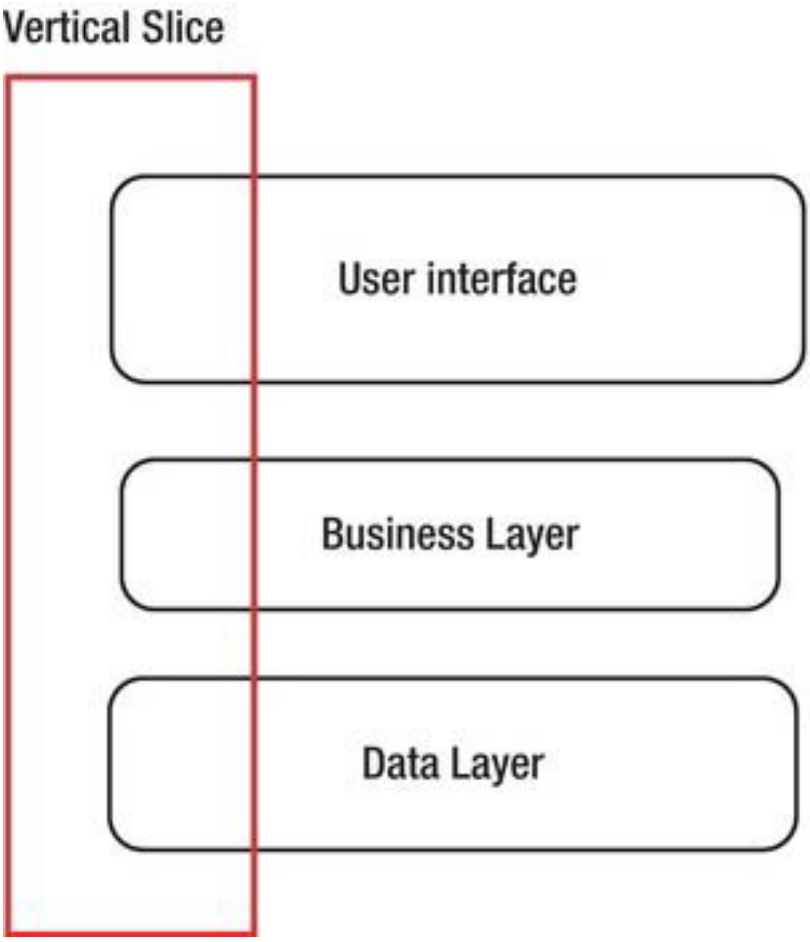
این خروجی‌ها، محصولات فعالیت‌های اسکرام هستند و به تیم در جهت‌یابی و شفافیت کار کمک می‌کنند. افزون بر این خروجی‌های اصلی خروجی‌های فرعی‌ای نیز از قبیل معیار پذیرش (الزاماتی که باید در حل یک مسأله برآورده شود تا بتوان آن را کامل شده تلقی کرد) وجود دارد.

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 مؤلفهٔ کلیدی اسکرام قرار دارد. این مؤلفه تعیین‌کنندهٔ 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 تعیین‌کنندهٔ 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 storyهای) پیشین مقایسه شده تا مشخص شود که در چه سطح ارزشی‌ای از نظر مشتری قراردارد. در طول یک اسپرینت، داستان کاربر (user storyها) را می‌توان به اسپرینت اضافه کرد. اما تا کامل شدن اسپرینت جاری، به تیم توسعه نشان داده نمی‌شود.

User Stories

همانطور که خاطرنشان شد، product backlog چیزی بیش از یک لیست اولویت‌بندی شده از داستان‌های کاربر (user storyها) 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. نکتهٔ 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 قطعهٔ عمودی تمام سطوح را آن گونه که در شکل 3-2 نشان داده شده، متاثر می‌سازد. این کوچکترین مقدار کاری است که تمام سطوح یک محصول را تحت تاثیر قرارداده و برای مشتری ارزش ایجاد می‌کند. با نوشتن داستان‌های کاربر (user storyها) به گونه‌ای که در بخشهای عمودی جایز است، می‌توان قابلیت پایه‌ای را در اولین داستان کاربر (user story) ایجاد کرده و سپس به سادگی قابلیتی به این ویژگی به عنوان نیازهای مشتری اضافه کرد.



یک شیوهٔ اطمینان از اینکه داستان کاربر (user story) فایدهٔ قطعهٔ عمودی بودن در یک سیستم را داراست این است که مطمئن شویم با «INVEST» منطبق است. «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 اندازهٔ داستان کاربر را تخمین زد.

اندازهٔ مناسب (Sized appropriately): داستانهای کاربر نباید آن قدر بزرگ باشند که تبدیلشان به یک طرح یا وظیفه یا امر اولویت‌بندی شده با درجهٔ مشخصی ممکن نباشد.

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

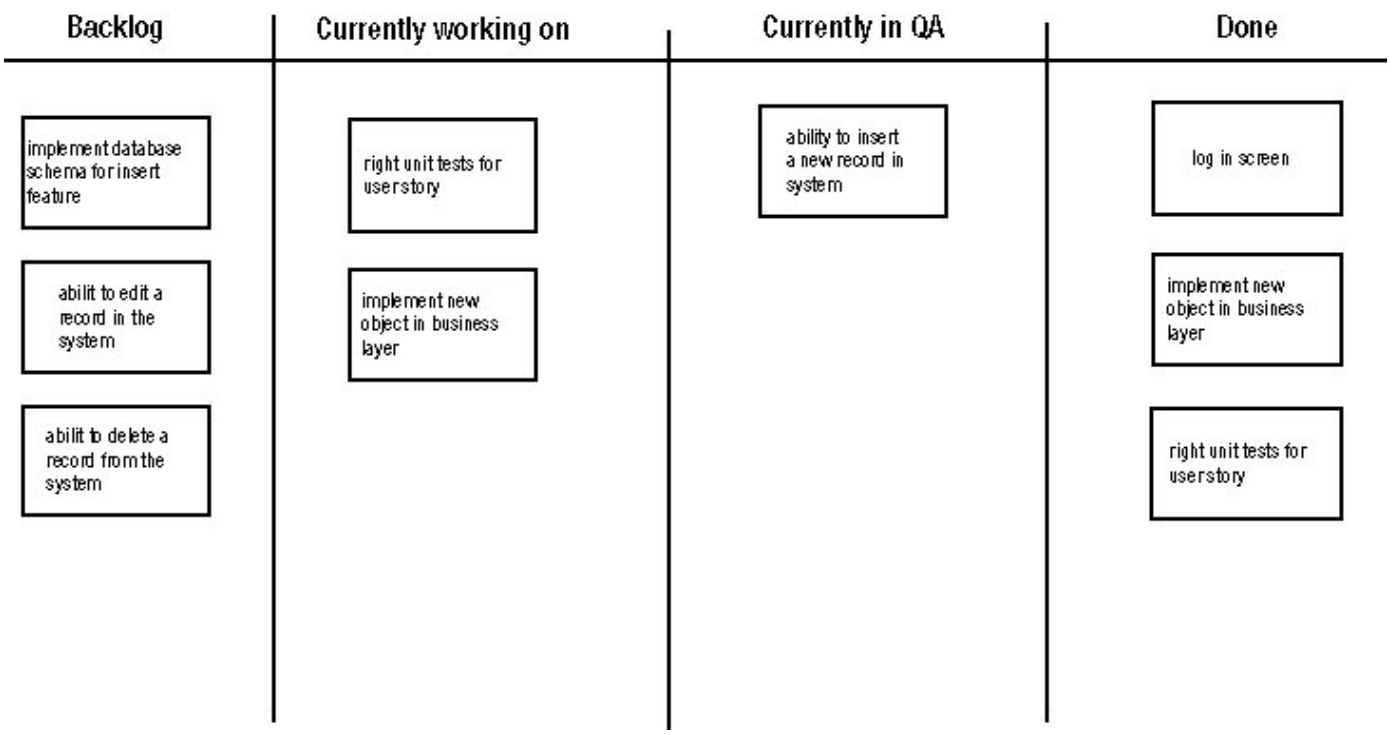
تعیین اندازهٔ backlog

تعیین اندازهٔ product backlog عبارت است از اندازه‌گیری سرعتی که تیم اسکرام می‌تواند مؤلفه‌های آن را ارائه کند. افراد در تخمین کار خوب عمل نمی‌کنند. همگی می‌دانیم که در تخمین دقیق اینکه چقدر طول می‌کشد تا یک کار را به طور کامل انجام دهیم، تا چه اندازه بد عمل می‌کنیم. تا کنون چند مرتبه این اتفاق افتاده است که از کسی بشنویم یا به خودمان بگوییم که 80 درصد کار را انجام داده‌ام و 20 درصد باقی‌ماندهٔ 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 کدام‌یک پیچیده‌تر از دیگری است؛ بدون آنکه تخصصی در آشپزی داشته باشیم. به دو چیز نگاه می‌کنیم و تشخیص می‌دهیم که کدام‌یک بزرگتر از دیگری است. تخمین اندازهٔ backlog تماماً یعنی تصمیم‌گیری دربارهٔ 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 سطلها 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 دربارهٔ 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. نکتهٔ 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

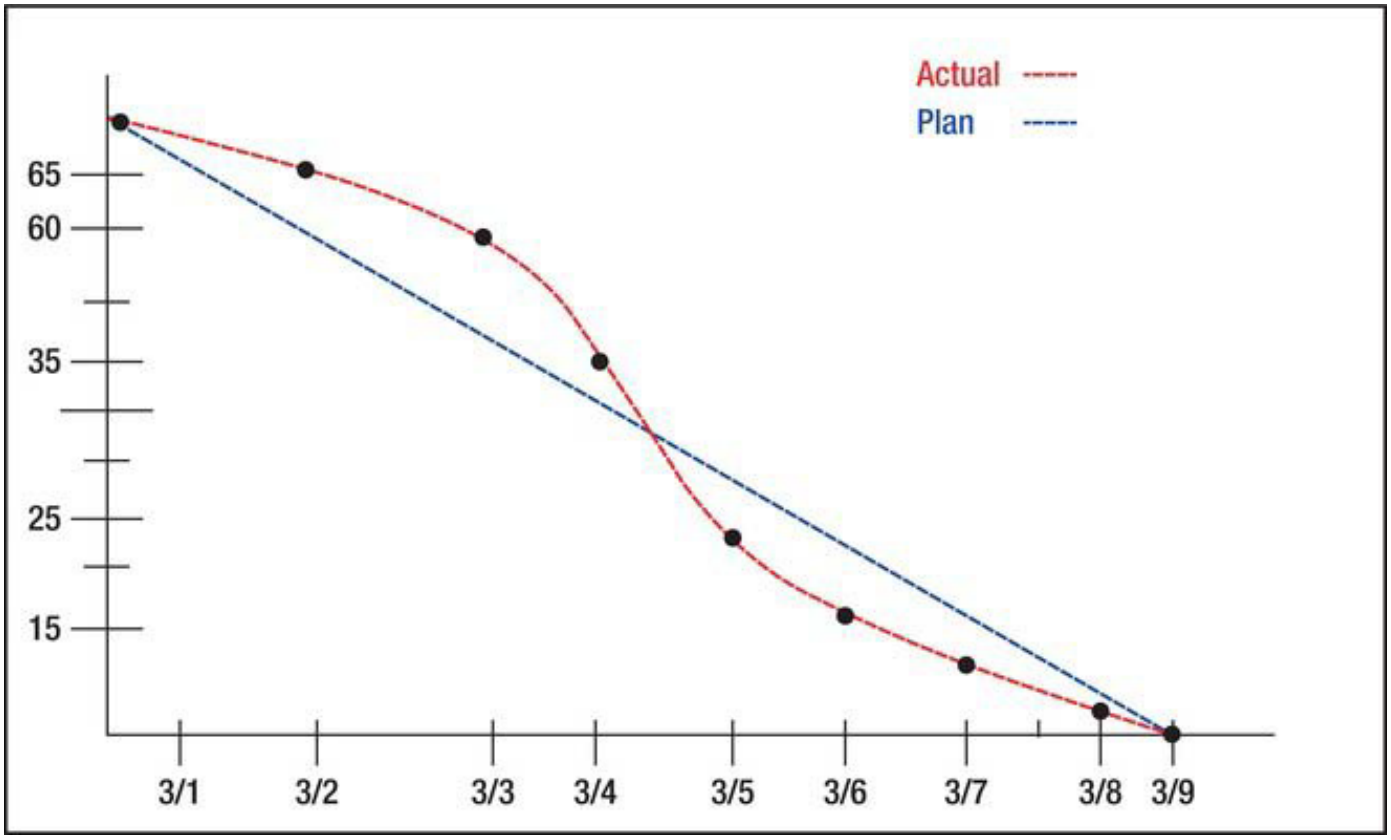
sprint backlog لیستی از همهٔ کارهای باقی‌مانده در یک اسپرینت است و باید توسط تیم انجام شود. sprint backlog زیرمجموعه‌ای از product backlog است. product backlog همهٔ داستانهای کاربران را که برای product مانده لیست می‌کند؛ اما sprint backlog حاوی همه‌ی داستان‌ها و وظایف باقی‌مانده برای اسپرینت است. نوعاً هنگامی که یک داستان کاربر برای یک اسپرینت انتخاب می‌شود، تیم، آن داستان کاربر را به تعدادی وظیفه تقسیم می‌کند.
یک وظیفه، تکهٔ 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 وظیفهٔ اجرای UI برای داستان کاربر. وظایفی که بر روی تابلوی وظایف – که با عنوان Kanban (معادل ژاپنی بیلبورد) نیز شناخته می‌شود- برای همهٔ تیم قابل رؤیت است. سایر مؤلفه‌های روی این تخته به همان ترتیب، حاوی اطلاعاتی دربارهٔ قرار ملاقاتهای جمع‌آوری نیازمندیها، کنترلهای بازبینی، تحقیقات، آزمون، طراحی و مراحل کد نویسی هستند. شکل 4-2 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. همهٔ اعضای تیم، در قبال به روز رسانی تابلو بر طبق اطلاعات جدید مقید خواهند بود.



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 بعلاوهٔ 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 باقیماندهٔ اسپرینت را در هر روز، به صورت گرافیکی همانند شکل 5-2 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.



معیار پذیرش

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

معیار پذیرش ضرورتاً شفاف کنندهٔ 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. معیار پذیرش، توسط صاحب محصول (product owner) به کمک مشتری ایجاد می‌شود. این معیار انتظار از داستان کاربر را تنظیم می‌کند. استفاده از این معیار درجای خود نقطهٔ شروع خوبی برای نوشتن تستهای خودکار یا حتی توسعهٔ آزمون محور توسط توسعه‌دهنده است. بدین طریق، توسعه‌دهنده چیزی را تولید می‌کند که مشتری بدان نیاز داشته و آن را می‌خواهد.

دیگر مزیت معیار پذیرش 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 معیار پذیرش 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 توجه‌ای 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 مرغ ها' 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 stackeholderها و یا ذینفعانی هستند که از پروژه منفعت

می‌برند، اما در مقابل تحویل پروژه مسئول و پسخگو 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 تسهیل کننده‌ی 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 دقیقا کاری را به آنها سپرده شده است انجام میدهند. اسکرام مستر با بخش‌های مختلف تیم، از صاحبان محصول گرفته تا تست کنندگان وذینفعان کسب و کار در تعامل است، تا مطمئن شود که تمام اعضای تیم برای پروژه مفید هستند و تمام دست آوردهای 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، هرمسئله 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 میکنند و تیم هم بر روی آیتم هایی با بیشترین ارزش برای مشتری، کار خواهد کرد. صاحب محصول همچنین product backlog را مدیریت کرده و تنها شخصی است که میتواند user storyها 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 نماینده‌ی 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 معمولا شامل دو تا ده نفر از افراد و همچنین ترکیبی از برنامه نویسان، تست کننده‌ها، طراحان محصول نهایی و اعضایی 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. مشخصه‌ی 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 آبشاری 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 هرکدام باید با تمام چشم اندازهای 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 storyها 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 بسارد.

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 0 sprint در نظر بگیرید. user stories (گزارشات کاربر) انتخاب شده توسط صاحب محصول، برای قرار گرفتن در print ، 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 درطرح 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 رایی 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 ارئه 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 زمانیکه 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.