متدولوژی های چابک

چابک چیست؟

- ”چابکی توانایی ایجاد و پاسخ به تغییرات به منظور کسب سود در محیط متلاطم حرفه می باشد. “
- چابکی و چابک بودن ارتباط تنگاتنگی با مسئله ی تغییر دارد.
- در محیط های توسعه نرم افزار چابک بودن بدان معناست که توانایی فرد برای پاسخگویی به تغییرات بیش از توانایی او برای برنامه ریزی و ارائه طرح می باشد.

چابکی در مقابل سبک وزنی

- سبک وزن بودن روش، بیشتر به کوچکی تیم پروژه و کاهش حجم کاری اشاره دارد.
- چابکی عدم قطعیت محیط و نیازمندیها را مورد اشاره قرار می دهد.
- می توان از رویکرد چابک در یک پروژه نسبتا بزرگ استفاده کرد.

چابکی و فردگرایی

- اساس کار روشهای چابک بر افراد و توانایی های آنها قرار گرفته است.
- از جمله روشهای فردگرا برنامه نویسی دو نفره می باشد.
- چابکی باعث افزایش نقش روز افزون برنامه نویسان در توسعه سیستم و بهره مندی هر چه بیشتر از تجربیات آنها در طراحی و تست سیستم می شود.

بیانیه چابک
(Agile Manifesto)

- در سال ۲۰۰۱ ، ۱۷ نفر از افرادی که در زمینه ی روشهای چابک فعالیت داشتند، به ابتکار Bob Martin دور هم جمع شدند تا علاوه بر تبادل نظر و بحث به استراحت و اسکی بپردازند.
- نتیجه ی این گردهمایی، بیانیه ای بود که به بیانیه ی چابک شهرت یافت

بیانیه چابک
(Agile Manifesto)

- چهار اصل روشهای چابک

۱. فردگرایی و تعامل برتر از فرآیندها و ابزارها.
۲. نرم افزار قابل اجرا برتر از مستندات مفهومی.
۳. همکاری با مشتریان برتر از مذاکرات قراردادگرا.
۴. پاسخ به تغییر برتر از دنباله روی از طرح.

فردگرایی و تعامل برتر از فرآیندها و ابزارها

- تکیه بر دانش افراد، توانایی و تعاملات بین آنها بجای ابزارها، فرایندها و طرحها.
- در محیطی که ابزارها پیشرفته هستند اما توسعه دهندگان مهارت کافی ندارند، توسعه نرم افزار با شکست مواجه می شود

نرم افزار قابل اجرا برتر از مستندات مفهومی

- توجه بیشتر به ساخت نرم افزار به نسبت مستندسازی.
- هدف اصلی توسعه نرم افزار است نه توسعه مستندات.
- مستندات راهنمای کاربران هستند نه محصول مورد نظر آنها.
- ۹۹ الی ۱۰۰ درصد مشتریان نرم افزاری کارآمد را بجای مستندی کامل انتخاب می کنند.

همکاری با مشتریان برتر از مذاکرات قراردادگرا

- مشکل همیشگی توسعه دهندگان عدم رسیدن به درک مشترک و همکاری نامناسب.
- داشتن قرارداد برای یک پروژه مهم است اما برای ارتباط با مشتری کافی نیست.

پاسخ به تغییر برتر از دنباله روی از طرح

- در طرح پروژه، تمهیداتی برای تغییرات احتمالی پیش بینی شود.
- تغییر یک واقعیت در توسعه ی نرم افزار است، واقعیتی که فرآیند توسعه بایستی آن را پشتیبانی کند.
  - خواسته های مشتری از نرم افزار تغییر می کند.
  - محیط حرفه ای که نرم افزار مرتبط به آن است، تغییر می کند.
  - تکنولوژی به مرور زمان تغییر می کند.

افسانه‌های بی‌اساس پیرامون متدولوژیهای چابک

افسانه‌های بی‌اساس پیرامون روشهای چابک

۱.متدولوژی چابک بی‌انضباط است.

۲.تیم چابک برنامه‌ریزی نمی‌کند.

۳.نرم‌افزارسازی چابک قابل پیش‌بینی نیست.

۴.نرم‌افزارسازی چابک مقیاس پذیرواندازه پذیر نیست.

۵.نرم‌افزارسازی چابک هم یک مد زودگذر است.

متدولوژی چابک بی‌انضباط است

- برخی می گویند چابک یعنی "اول کد کن، بعد درستش کن"
- یکپارچگی مستمر، برنامه‌نویسی آزمون‌گرا، بازساخت (Refactoring) و حتی جفت برنامه نویسی به انضباط بالایی نیاز دارد.
- بسیاری از رویه‌های تعریف شده در متدولوژی‌های چابک برای موفقیت پروژه نه تنها به انضباطی انفرادی، بلکه به تعهدی جمعی بستگی تام دارد.

تیم چابک برنامه‌ریزی نمی‌کند

- برنامه ریزی تفصیلی و وظیفه محور خیلی زود از واقعیت‌های فنی و کاری فاصله می‌گیرند.
- برنامه ریزی در تیم های چابک نه تنها در ابتدای پروژه بلکه در سراسر طول پروژه گسترده شده است.
- شیوه تکراری و افزایشی چابک شامل برنامه‌ریزی پروژه نیز می‌شود.

نرم‌افزارسازی چابک قابل پیش‌بینی نیست

- رویکرد سنتی با استفاده از روشهای جزئی نگر و فعالیت محور، میزان انحراف پروژه ازهر فعالیت را می سنجد و خروجیها را پیش بینی می کند.
- این روش تنها "باوری از پیش بینی پذیری" را بدست می دهد.
- این روش ها طی چندین دهه به ندرت پیشرفت کرده اند زیرا پیچیدگی آنها زیاد است.

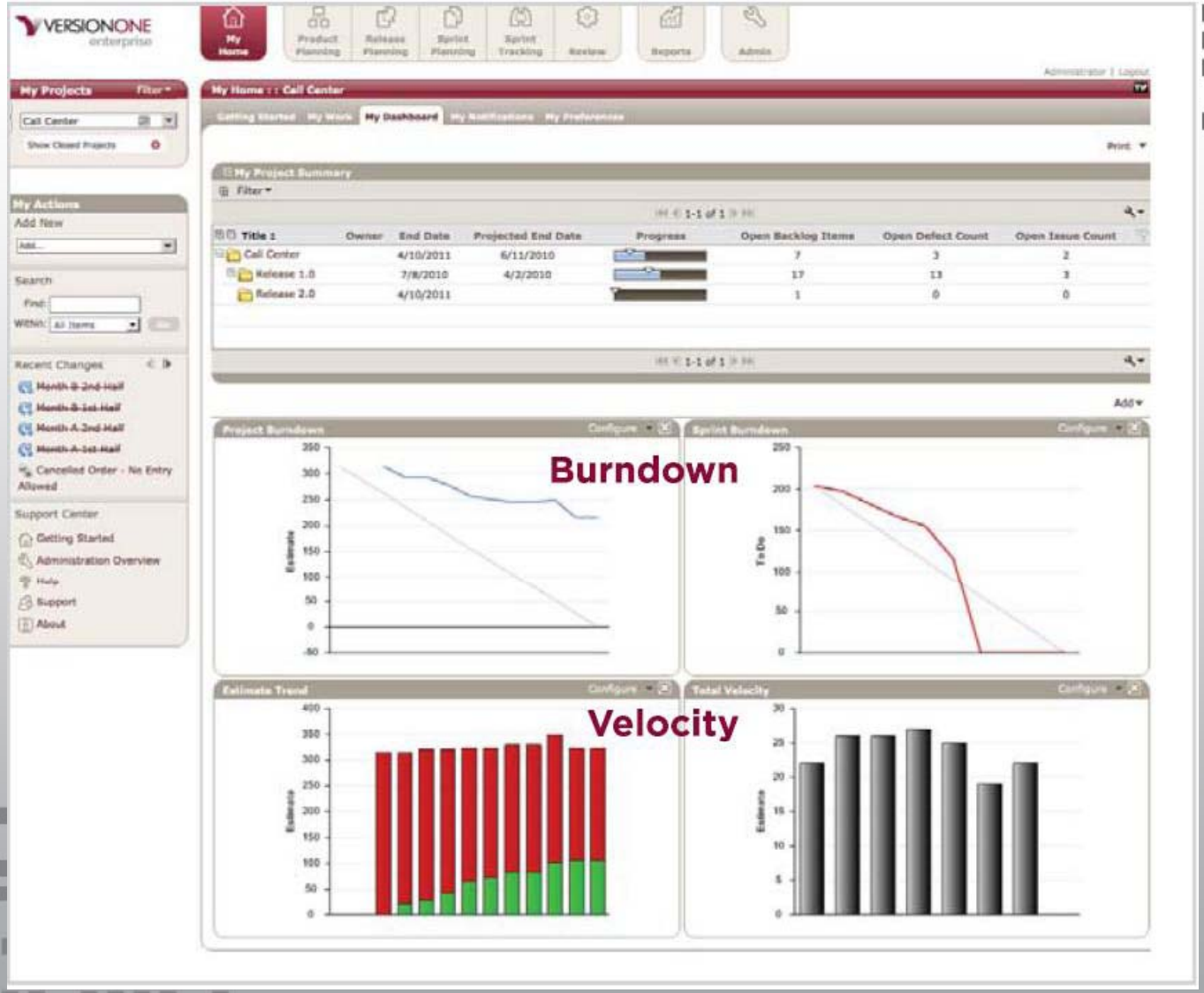
نرم‌افزارسازی چابک قابل پیش‌بینی نیست

- تیم‌های چابک، از روش سطح بالا و ویژگی‌محوری استفاده می کنند.
- برنامه‌ریزی جزییات با پیشرفت پروژه و مشخص شدن مشخصات تفصیلی آن به مرور اضافه شده و منجر به برنامه ریزی مداوم و تکراری می‌شود.
- این روش پیچیدگی های ارث بری و عدم قطعیت پیاده سازی نرم افزارها را پوشش می دهد.

نرم‌افزارسازی چابک قابل پیش‌بینی نیست

- استفاده از اطلاعات قبلی پروژه در خصوص هر ویژگی باعث بالا رفتن مشهود کیفیت برنامه ها می شود.
- نمودارهای PERT و GANT با نمودارهای ساده تری از جمله Velocity و Burndown جایگزین شده اند.
- به جای برنامه‌ریزی سالانه برای باقی‌مانده پروژه، به طور مستمر برنامه‌ریزی، تخمین و اولویت‌بندی می‌کنند.
- قدرت برنامه‌ریزی تیم بهبود می یابد.

Velocity & Burndown Chart



رویکرد چابک مقیاس پذیرواندازه پذیر نیست

- تولید نرم‌افزار ذاتا دارای محدودیتهایی در مقیاس پذیری (Scalability)است و این محدود به متدولوژی ها نیست.
- هر چه پروژه بزرگ تر باشد، احتمال شکستش هم بیش‌تر است و هر چه تعداد افراد دخیل در آن بیش‌تر باشد، ریسک های ارتباطی هم بیشتر است.
- رویکرد چابک پروژه‌های کوچکتر، تکرارهای کوتاهتر و تیم‌های کوچکتری را پیشنهاد می‌کند.

نرم‌افزارسازی چابک هم یک مد زودگذر است

- مدهای تکنیکی بصورت کلی همراه با تبلیغات زیاد و پر سروصدا می باشند و طی مدت ۱ الی ۲ سال فراموش می شوند.
- رویکرد چابک نه تنها فراموش نشده بلکه در کمتر از ۵ سال به رویکرد ترجیحی سازمانها بدل شده است.
- با استفاده از این روش می‌توان (Ttime-to-Mmarket) را کاهش داد، کیفیت را بهبود بخشید و روابطشان را با ذینفعان بازار مستحکم‌تر کرد.

نرم‌افزارسازی چابک هم یک مد زودگذر است

- بر اساس تحقیقی از شرکت VersionOne سه دلیل عمده انتقال از رویکرد سنتی تولید نرم‌افزار به رویکرد چابک عبارت است:
  ۱. سرعت بخشیدن به عرضه نرم‌افزار
  ۲. هم‌سو کردن نیازهای کسب‌وکار با نیازهای فناوری اطلاعات
  ۳. بهبود شفافیت فرآیند تولید نرم‌افزار
- این دلایل نشان می دهد رویکرد چابک مدی زودگذر نیست.

منابع

- VersionOne Web site
- 5 Myths Of Agile By Robert Holler