فصل چهارم

اصول راهنمای عملیات

اصول راهنمای فرآیند – ۱

- **اصل: *چابک باشید.*** این که آیا مدل فرایندی که انتخاب می کنید، قاعده مند یا چابک است، اصول اساسی توسعه چابک باید بر رویکرد شما حاکم باشد.
- **اصل۲: *در هر مرحله بر کیفیت تمرکز کنید.*** شرایط خروج هر فعالیت، عمل و وظیفه فرایند باید بر کیفیت محصول کاری که تولید شده است تمرکز کند.
- **اصل۳: *آماده برای انطباق باشید.*** فرایند یک تجربه مذهبی نیست و تعصب در آن جایی ندارد. در صورت لزوم، رویکرد خود را با توجه به محدودیت های تحمیل شده توسط مشکل، افراد و خود پروژه تطبیق دهید.
- **اصل۴: *یک تیم موثر ایجاد کنید.*** فرایند و عملیات مهندسی نرم افزار مهم هستند، اما مهم تر از آنها افراد است. یک تیم خودسازمانده ایجاد کنید که اعتماد و احترام متقابل داشته باشد.

اصول راهنمای فرآیند – ۲

- **اصل۵: *برای ارتباط و هماهنگی مکانیزمی ایجاد کنید.*** پروژه ها با شکست مواجه می شوند، زیرا اطلاعات مهم در شکاف می افتند و / یا ذینفعان قادر به هماهنگی تلاش های خود برای ایجاد یک محصول نهایی موفق نیستند.
- **اصل۶: *تغییر را مدیریت کنید.*** راهکار می تواند رسمی یا غیررسمی باشد، اما باید مکانیزم هایی برای مدیریت روشی که تغییرات درخواست، ارزیابی، تایید و پیاده سازی می شوند، برقرار شده باشد.
- **اصل۷: *مخاطرات را ارزیابی کنید.*** بسیاری از چیزها می تواند به محض اینکه نرم افزار توسعه می یابد اشتباه شوند. ضروری است که برنامه های احتمالی را ایجاد کنید.
- **اصل۸: *محصولات کاری ایجاد کنید که برای دیگران ارزشمند است.*** فقط محصولات کاری را ایجاد کنید که برای سایر فرآیندها، اقدامات و یا وظایف ارزش ارائه کنند.

اصول راهنمای عملیات – ۱

- **اصل۱: *تفرقه بینداز و حکومت کن.* از دیدگاه فنی بیشتر، تحلیل و طراحی همیشه باید بر تجزیه نگرانی ها (SoC) تأکید داشته باشد.**
- **اصل۲: *کاربرد انتزاع را درک کنید.* در اصل، انتزاع ساده سازی یک عنصر پیچیده سیستم است که برای مفهوم ارتباطی در یک عبارت به کار رفته است.**
- **اصل۳: *برای انسجام بکوشید.* یک مفاد آشنا نرم افزار را برای استفاده آسان تر می کند.**
- **اصول۴: *بر انتقال اطلاعات تمرکز کنید.* توجه ویژه ای به تحلیل، طراحی، ساخت و آزمون واسطه ها داشته باشید.**

اصول راهنمای عملیات – ۲

- ***اصل۵: نرم افزاری بسازید که قابلیت پیمانه ای را به شکل مؤثرنشان می دهد.***
  تجزیه نگرانی ها (اصل اول) یک فلسفه برای نرم افزار ایجاد می کند. پیمانه ای بودن مکانیزمی را برای تحقق این فلسفه فراهم می کند.
- ***اصل۶: به الگوها دقت کنید.*** **Brad Appleton اشاره می کند که: "هدف الگوها در جامعه نرم افزاری، ایجاد بدنه ادبیاتی است تا به توسعه دهندگان نرم افزار کمـــک کند مشکلات تکـــراری که در سراسر توسعه نرم افزار اتفاق می افتد، حل کند.**
- ***اصل۷: در صورت امکان، مشکلات و راه حل آن را از چندین دیدگاه مختلف ارائه دهید.***
- ***اصل۸: به یاد داشته باشید که کسی نرم افزار را نگهداری خواهد کرد.***

اصول ارتباط– ۱

- **اصل۱: *گوش دهید.* سعی کنید روی سخنان سخنران تمرکز کنید، نه روی پاسخ خود به آن صحبت ها.**
- **اصل۲: *قبل از برقراری ارتباط آماده شوید.* قبل از اینکه با دیگران ملاقات کنید برای درک آن وقت بگذارید.**
- **اصل۳: *شخصی باید فعالیت را تسهیل کند.* هر جلسه ارتباطی باید یک رهبر (تسهیل کننده) داشته باشد تا (۱)مکالمه را در مسیر سازنده ادامه دهد؛ (۲) هر درگیری که رخ می دهد، میانجی گری کند و (۳) اطمینان یابد اصول دیگر دنبال می شود.**
- **اصل۴: *ارتباط چهره به چهره بهترین است.* اما معمولا وقتی بهتر عمل می کند که ارائه اطلاعات مرتبط دیگر، آماده باشد.**

اصول ارتباط– ۲

- **اصل ۵: *یادداشت بردارید و تصمیمات را مستند کنید.*** **کسی که در ارتباطات شـــــرکت می کند باید به عنوان ضبط کننده عمل کند و همه نکات و تصمیمات مهم را بنویسد.**
- **اصل۶: *برای همکاری تلاش کنید.*** **همکاری و هماهنگی زمانی رخ می دهد که دانش جمعی اعضای تیم ترکیب شود...**
- **اصل ۷: *تمرکز کنید، بحث خود را پیمانه سازی کنید.*** **هرچه افراد بیشتری در ارتباط شرکت می کنند، بیشتر احتمال می رود که بحث از یک موضوع به موضوع دیگری برود.**
- ***اصل ۸: اگر چیزی واضح نیست، یک تصویر بکشید.***
- ***اصل ۹: (الف) هنگامی که شما با چیزی موافق هستید، از آن عبور کنید؛ (ب) اگر نمی توانید درباره چیزی توافق کنید، از آن عبور کنید؛ (پ) اگر یک ویژگی یا عملکرد واضح نیست و در حال حاضر نمیتوان آن را روشن کرد، از آن عبور کنید.***
- ***اصل ۱۰: مذاکره یک مسابقه یا یک بازی نیست. بهترین حالت زمانی است که هر دو طرف برنده شوند.***

اصول برنامه ریزی– ۱

- **اصل ۱: *دامنه پروژه را درک کنید.* اگر ندانید کجا هستید و به کجا می روید، استفاده از نقشه راه غیرممکن است. دامنه، مقصدی را برای تیم نرم افزاری فراهم می کند.**
- **اصل ۲: *مشتری را در فعالیت برنامه ریزی درگیر کنید.* مشتری اولویت را تعریف می کند و محدودیت های پروژه را ایجاد می کند.**
- **اصل ۳: *تکراری بودن برنامه ریزی را به رسمیت بشناسید.* طرح پروژه هرگز بر سنگ حکاکی نمی شود. با آغاز کار، ممکن است همه چیز تغییر کند.**
- **اصل ۴: *بر اساس آنچه می دانید، تخمین بزنید.* هدف از تخمین، نشان دادن تلاش، هزینه و مدت زمان کار بر اساس درک فعلی تیم از کاری که می خواهد انجام شود است.**

اصول برنامه ریزی– ۲

- **اصل ۵: *در حالی که برنامه ریزی می کنید، مخاطره را در نظر بگیرید.* اگر مخاطراتی را شناسایی کرده اید که دارای تأثیر زیاد و احتمال بالا هستند، برنامه ریزی احتمالی ضروری است.**
- **اصل ۶: *واقع بین باشید. افراد ۱۰۰ درصد روز را کار نمی کنند.***
- **اصل ۷: *همانطور که برنامه را تعریف می کنید، granularity را تنظیم کنید.* granularity به سطح جزئیاتی اشاره دارد که به عنوان طرح پروژه توسعه داده شده معرفی می شود.**
- **اصل ۸: *تعریف کنید که چطور می خواهید کیفیت را تضمین کنید.* طرح باید مشخص کند که چگونه تیم نرم افزاری قصد دارد کیفیت را تضمین کند.**
- **اصل ۹: *توضیح دهید که چگونه می خواهید تغییر را اصلاح کنید.* حتی بهترین برنامه ریزی می تواند با تغییر کنترل نشده از بین برود.**
- **اصل ۱۰: *برنامه را به طور مرتب پیگیری کنید و تنظیمات را در صورت نیاز انجام دهید.* پروژه های نرم افزاری یک بار زمانی از زمان بندی عقب می افتند.**

اصول مدل سازی

- **در کار مهندسی نرم افزار، دو کلاس مدل می تواند ایجاد شود:**
  - **مدل های نیازمندی ها (یا مدل های تحلیل) نیازهای مشتری با ترسیم نرم افزار در سه حوزه مختلف نشان می دهند: دامنه اطلاعات، دامنه عملکرد و دامنه رفتار.**
  - **مدل های طراحی نشان دهنده ویژگی های نرم افزار است که به تمرین کنندگان کمک می کند تا آنها را به طور موثر بسازند: معماری، رابط کاربر و جزئیات سطح مولفه.**

اصول مدل سازی نیازمندی ها

- اصل ۱: دامنه اطلاعات یک مشکل باید نمایان و درک شود.
- اصل ۲: توابعی که نرم افزار انجام می شود باید تعریف شود.
- اصل ۳: رفتار نرم افزار (در نتیجه وقایع خارجی) باید نشان داده شود.
- اصل ۴: مدل هایی که اطلاعات، عملکرد و رفتار را تصویر می کنند باید به نحوی تقسیم شوند که جزئیات را به شکل لایه ای (یا سلسله مراتبی) به نمایش در آورند.
- اصل ۵: کار تحلیل باید از اطلاعات ضروری به سمت جزئیات پیاده سازی حرکت کند.

اصول مدل سازی طراحی

- اصل ۱: طراحی باید تا مدل نیازمندی ها قابل پیگیری باشد.
- اصل ۲: همیشه در نظر بگیرید معماری سیستم ساخته می شود.
- اصل ۳: طراحی داده ها به اندازه طراحی توابع پردازش مهم است.
- اصل ۵: طراحی رابط کاربری باید با نیازهای کاربر نهایی منطبق شود. با این حال، در هر مورد، باید بر سهولت استفاده تاکید کرد.
- اصل ۶: طراحی سطح مولفه باید مستقل از عملکرد باشد.
- اصل ۷: اجزاء باید به صورت آزاد به یکدیگر و محیط خارجی وصل شوند.
- اصل ۸: ارائه(مدل) طراحی باید به راحتی قابل فهم باشد.
- اصل ۹: طراحی باید به طور تکراری توسعه یابد. با هر تکرار، طراح باید برای سادگی بیشتر تلاش کند.

اصول مدل سازی چابک

- اصل ۱: هدف اصلی تیم نرم افزاری ساخت نرم افزار است، نه ایجاد مدل.
- اصول ۲: سبک سفر کنید – مدل های بیشتری از آنچه نیاز دارید، ایجاد نکنید.
- اصول ۳: تلاش کنید تا ساده ترین مدل که مشکل یا نرم افزار را توصیف می کند، تولید کنید.
- اصول ۴: مدل ها را طوری بسازید که امکان تغییر داشته باشند.
- اصل ۵: بتوانید یک هدف صریح برای هر مدل که ایجاد شده است، بیان کنید.
- اصل ۶: مدل هایی که توسعه می دهید را با سیستم های در دست منطبق کنید.
- اصل ۷: سعی کنید مدل های مفید ایجاد کنید، اما ساختن مدل های کامل را فراموش کنید.
- اصل ۸: در مورد نحو مدل، متعصب نباشید. اگر با محتوا به شکل موفقیت آمیز ارتباط برقرار می کند، ارائه ثانویه است.
- اصل ۹: اگر غرایز شما می گویند مدل درست نیست حتی اگر روی کاغذ درست به نظر می رسد، دلیلی برای نگرانی دارید.
- اصل ۱۰: در اسرع وقت بازخورد بگیرید.

اصول ساخت

- فعالیت ساخت و ساز شامل مجموعه ای از وظایف کد نویسی و تست است که منجر به نرم افزار عملیاتی می شود که برای تحویل به مشتری یا کاربر نهایی آماده است.
- اصول و مفاهیم کدنویسی همتراز با سبک برنامه نویسی، زبان برنامه نویسی و روش های برنامه نویسی هستند.
- اصول و مفاهیم آزمون منجر به طراحی آزمون هایی می شوند که مرتبا کلاس های مختلف خطاها را با حداقل میزان زمان و تلاش کشف می کنند.

اصول آماده سازی

- **قبل از نوشتن یک خط کد مطمئن شوید:**
  - **مشکلی که سعی در حل آن دارید، درک کرده اید.**
  - **اصول و مفاهیم طراحی اولیه را درک کنید.**
  - **زبان برنامه نویسی را انتخاب کنید که نیازهای نرم افزاری که ساخته می شود و محیطی که در آن کار می کند، را برآورده می کند.**
  - **محیط برنامه نویسی را انتخاب کنید که ابزارهایی را فراهم می کند که کار شما را ساده تر خواهد کرد.**
  - **مجموعه ای از آزمون های واحدی ایجاد کنید که بعد از تکمیل کد مولفه، برای آن اعمال شود.**

اصول کد نویسی

- **با شروع نوشتن کد مطمئن شوید:**
  - **الگوریتم های خود را با دنبال نمودن عملیات برنامه نویسی ساخت یافته محدود کنید.**
  - **استفاده از برنامه نویسی جفتی را در نظر بگیرید**
  - **ساختارهای داده را انتخاب کنید که نیازهای طراحی را برآورده می کند.**
  - **معماری نرم افزار را درک کنیدو واسط هایی که با آن سازگار است، ایجاد کنید.**
  - **منطق شرطی را تا حد ممکن ساده کنید.**
  - **حلقه های تودرتو را طوری ایجاد کنید که به راحتی قابل آزمون باشند.**
  - **نام های متغیر معنا دار انتخاب کنید و دیگر استانداردهای برنامه نویسی محلی را دنبال کنید.**
  - **کدی بنویسید که خود مستند است**
  - **یک چینش ظاهری (به عنوان مثال، تو رفتگی و خطوط خالی) ایجاد کنید تا به درک آن کمک کند.**

اصول آزمون

- Al Davis موارد زیر را پیشنهاد می کند:
  - اصل ۱: تمام آزمون ها باید قابل ردیابی تا نیازمندی های مشتری باشند.
  - اصل ۲: آزمون ها باید قبل از شروع، برنامه ریزی شوند.
  - اصل ۳: اصل Pareto برای تست نرم افزار به کار می رود.
  - اصل ۴: تست باید "کوچک" شروع شود و پیشرفت در آزمون "زیاد" باشد.
  - اصل ۵: آزمون جامع ممکن نیست.

اصول استقرار

- **اصل ۱: *انتظارات مشتری از نرم افزار باید مدیریت شود.* اغلب، مشتری انتظاری بیش از آنچه تیم وعده داده دارد و بلافاصله ناامید می شود.**
- **اصل ۲: *یک بسته تحویل کامل باید مونتاژ و تست شود.***
- **اصل ۳: *یک سازماندهی پشتیبانی باید قبل از تحویل نرم افزار ایجاد شود.* کاربر نهایی انتظار پاسخگویی و اطلاعات دقیق در زمان بروز سوال یا مشکل دارد.**
- **اصل ۴: *باید به کاربران نهایی مفاد آموزشی مناسب ارائه شود.***
- **اصل ۵: *نرم افزار شامل خطا باید ابتدا درست شود، سپس تحویل داده شود.***