

فهرست مطالب

12	1. مهندسی نرم افزار.....
12	1.1. خلاصه فصل.....
13	1.2. مقدمه.....
13	1.3. مفاهیم نرم افزار و مهندسی نرم افزار.....
17	1.4. چرخه حیات نرم افزار.....
19	1.5. چرخه حیات ساخت یافته محصول نرم افزاری.....
20	1.5.1. مزایای چرخه حیات ساخت یافته.....
21	1.5.2. فواید چرخه حیات ساخت یافته.....
22	1.6. نتیجه گیری.....
23	1.7. سوالات متداول.....
23	1.8. منابعی برای مطالعه بیشتر.....
26	2. متدولوژی های تولید نرم افزار.....
26	2.1. خلاصه فصل.....
27	2.2. مقدمه.....
27	2.3. متدولوژی های نرم افزاری.....
28	2.3.1. مدل ترتیبی خطی.....
32	2.3.2. مدل پیش نمونه سازی.....
33	2.3.3. مدل های تکامل یافته.....
34	2.3.4. مدل افزایشی.....
36	2.3.5. مدل توسعه سریع کاربردها.....
39	2.3.6. الگوی توسعه تجزیه تحلیل سیستم های ساخت یافته.....
42	2.3.7. مدل حلزونی.....
44	2.3.8. روش مارپیچی برنده برنده.....
46	2.3.9. مدل مونتاژ مؤلفه ها.....
47	2.3.10. مدل توسعه همروند.....
48	2.3.11. مدل روش های رسمی.....
49	2.3.12. تکنیک های نسل چهارم.....
50	2.3.13. متدولوژی XP.....
54	2.3.14. مدل CMM.....
55	2.4. کاربردهای عملی متدولوژی های تولید نرم افزار.....

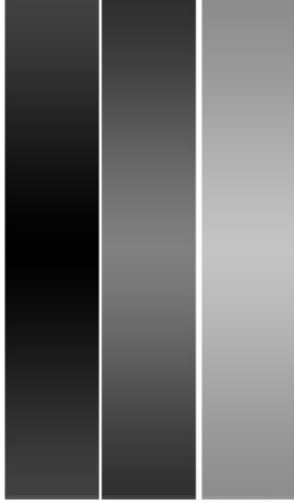
- ۲.۵. نتیجه گیری..... 57
- ۲.۶. سوالات متداول..... 58
- ۲.۷. منابعی برای مطالعه بیشتر..... 58
۳. مستندات محصول نرم افزاری..... 60
- ۳.۱. خلاصه فصل..... 60
- ۳.۲. مقدمه..... 61
- ۳.۳. فرآیند مستندسازی و طبقه بندی انواع مستندات..... 62
- ۳.۳.۱. مستند چشم انداز پروژه 63
- ۳.۳.۲. مستند توصیف مشخصات فنی محصول 68
- ۳.۳.۳. مستند تحلیل 69
- ۳.۳.۴. مستند خصوصیات طراحی 69
- ۳.۳.۵. مستند توصیف معماری 70
- ۳.۳.۶. مستند نمای واسط کاربر 70
- ۳.۴. راهکار عملی تهیه مستندات مورد نیاز..... 71
- ۳.۵. نتیجه گیری..... 73
- ۳.۶. سوالات متداول..... 73
- ۳.۷. منابعی برای مطالعه بیشتر..... 74
۴. دیاگرام جریان داده نرم افزار..... 76
- ۴.۱. خلاصه فصل..... 76
- ۴.۲. مقدمه..... 77
- ۴.۳. نمودارهای جریان داده و اجزاء آن..... 78
- ۴.۴. انواع نمودارهای جریان داده..... 80
- ۴.۴.۱. نمودار جریان داده فیزیکی 81
- ۴.۴.۲. نمودار جریان داده منطقی 81
- ۴.۵. رسم نمودارهای جریان داده فیزیکی و منطقی..... 82
- ۴.۵.۱. نمودار سطح صفر 83
- ۴.۵.۲. نمودار جریان داده سطح 1 84
- ۴.۵.۳. نمودار جریان داده سطح 2 85
- ۴.۵.۴. تبدیل نمودار جریان داده فیزیکی به منطقی 86
- ۴.۶. نتیجه گیری..... 90

91	۴.۷	سؤالات متداول
91	۴.۸	منابعی برای مطالعه بیشتر
94	۵	دیاگرام UML نرم افزار
94	۵.۱	خلاصه فصل
95	۵.۲	مقدمه
99	۵.۳	دیاگرام های زبان مدل سازی یکپارچه
101	۵.۳.۱	نمودار مورد کاربرد
103	۵.۳.۲	نمودار کلاس
105	۵.۳.۳	نمودار حالت
106	۵.۳.۴	نمودار توالی
107	۵.۳.۵	نمودار همکاری
108	۵.۳.۶	نمودار اجزاء
109	۵.۳.۷	نمودار استقرار
110	۵.۳.۸	نمودار فعالیت
111	۵.۳.۹	نمودار اشیاء
112	۵.۳.۱۰	نمودار بسته
112	۵.۴	نتیجه گیری
113	۵.۵	سؤالات متداول
113	۵.۶	منابعی برای مطالعه بیشتر
116	۶	متدولوژی RUP
116	۶.۱	خلاصه فصل
117	۶.۲	مقدمه
118	۶.۳	خصوصیات آریوپی
120	۶.۴	متدولوژی آریوپی
122	۶.۴.۱	مرحله آغازین
123	۶.۴.۲	مرحله تشریح
126	۶.۴.۳	مرحله ساخت
127	۶.۴.۴	مرحله انتقال
128	۶.۵	نظم های آریوپی
143	۶.۶	مستندات آریوپی
146	۶.۷	نتیجه گیری

۶.۸	سؤالات متداول.....	147
۶.۹	منابعی برای مطالعه بیشتر.....	147
۷	کیفیت نرم افزار.....	150
۷.۱	خلاصه فصل.....	150
۷.۲	مقدمه.....	151
۷.۳	سیستم های مدیریت کیفیت.....	152
۷.۴	کیفیت نرم افزار.....	153
۷.۵	ابعاد کیفیت نرم افزار.....	154
۷.۵.۱	ارتباط بین ابعاد کیفیت	154
۷.۶	هزینه کیفیت نرم افزار.....	155
۷.۷	مدیریت کیفیت نرم افزار.....	156
۷.۸	استانداردهای مدیریت کیفیت نرم افزار.....	157
۷.۸.۱	استاندارد مدیریت کیفیت نرم افزار	161
۷.۸.۲	استانداردسازی	161
۷.۸.۳	استاندارد SQuaRE	162
۷.۹	کیفیت در بررسی نیاز مشتریان نرم افزار.....	164
۷.۱۰	گسترش عملکرد کیفیت در نرم افزار.....	165
۷.۱۱	نتیجه گیری.....	165
۷.۱۲	سؤالات متداول.....	166
۷.۱۳	منابعی برای مطالعه بیشتر.....	166
۸	تضمین کیفیت نرم افزار.....	168
۸.۱	خلاصه فصل.....	168
۸.۲	مقدمه.....	169
۸.۳	کنترل کیفیت نرم افزار.....	170
۸.۴	تضمین کیفیت نرم افزار.....	170
۸.۵	اهداف تضمین کیفیت نرم افزار.....	171
۸.۶	وظایف SQA.....	172
۸.۷	پیاده سازی برنامه SQA.....	172
۸.۸	فعالیت های SQA.....	175
۸.۹	جایگاه SQA در پروژم.....	175

۸.۱۰	حرکت کیفی.....	176
۸.۱۱	کیفیت نرم افزار.....	177
۸.۱۲	مشکلات توسعه دهندگان نرم افزار.....	180
۸.۱۳	چالش های مدیر پروژه.....	181
۸.۱۳.۱	کارفرمایان و مشتریان	181
۸.۱۳.۲	کارکنان بخش نرم افزار	182
۸.۱۳.۳	برآوردهای زمانی و هزینه ای پروژه	183
۸.۱۳.۴	مدیریت و نیروی انسانی با کیفیت	184
۸.۱۴	نتیجه گیری.....	185
۸.۱۵	سؤالات متداول.....	187
۸.۱۶	منابعی برای مطالعه بیشتر.....	187
۹	تست و تضمین نرم افزار.....	190
۹.۱	خلاصه فصل.....	190
۹.۲	مقدمه.....	191
۹.۳	آزمون نرم افزار.....	192
۹.۴	تست نرم افزار در تئوری و عمل.....	194
۹.۴.۱	تفاوت تست و اشکال زدایی	195
۹.۴.۲	تست مبتنی بر نیازمندی و خطرات	196
۹.۵	تست و کیفیت نرم افزار.....	198
۹.۶	روش های تست.....	199
۹.۶.۱	تست ایستا	201
۹.۶.۲	تست پویا	202
۹.۷	تکنیک های پایه آزمون نرم افزار.....	202
۹.۷.۱	تست جعبه سفید	203
۹.۷.۲	تست جعبه-سیاه	208
۹.۸	استراتژی های چگونگی اجرای تست.....	211
۹.۹	انواع تست.....	212
۹.۱۰	مراحل تست.....	213
۹.۱۰.۱	تست واحد	214
۹.۱۰.۲	تست زیر سیستم	214
۹.۱۰.۳	تست ترکیب و جامعیت	215

- ۹.۱۰.۴. تست اعتبار سنجی 216
- ۹.۱۰.۵. تست سیستم 216
- ۹.۱۱. ابزارهای پشتیبانی خودکار برای تست 217
- ۹.۱۲. نتیجه گیری 219
- ۹.۱۳. سوالات متداول 220
- ۹.۱۴. منابعی برای مطالعه بیشتر 221
- ۱۰. محیط توسعه نرم افزار 224**
- ۱۰.۱. خلاصه فصل 224
- ۱۰.۲. مقدمه 225
- ۱۰.۳. محیط توسعه نرم افزار 227
- ۱۰.۳.۱. NET Framework 229
- ۱۰.۳.۲. Delphi 232
- ۱۰.۳.۳. Visual Basic 234
- ۱۰.۴. کدنویسی ایمن 235
- ۱۰.۵. پرهیز از اشتباهات و خطاهای برنامه سازی 237
- ۱۰.۶. نتیجه گیری 246
- ۱۰.۷. سوالات متداول 246
- ۱۰.۸. منابعی برای مطالعه بیشتر 246
- ضمائم 247**
- ضمیمه اول: مستند چرخه حیات نرم افزار 248
- ضمیمه دوم: مستند طراحی نرم افزار 249
- ضمیمه سوم: مستند طراحی بانک اطلاعاتی 251
- ضمیمه چهارم: استانداردهای تولید و توسعه نرم افزار 253
- ضمیمه پنجم: مهمترین مشاغل مرتبط مهندسی نرم افزار 256
- ضمیمه ششم: چک لیستهای مهم نظارتی پروژه های نرم افزاری 259



فصل اول

مهندسی نرم افزار

در این فصل

با مفهوم نرم افزار، بحران نرم افزار و جایگاه حیاتی مهندسی نرم افزار در جامعه دانش محور امروز آشنا شده و به معرفی انواع سطوح بلوغ نرم افزار، چرخه حیات نرم افزار و چرخه حیات ساخت یافته یک محصول نرم افزاری و کاربردها و مزایای آن می پردازیم و جزئیات فازها و گامهای چرخه را شرح می دهیم.

۱. مهندسی نرم افزار



پروفسور کاوه پهلوان (Kaveh Pahlavan)

متخصص و محقق در زمینه‌های مهندسی نرم افزار، تک پردازنده
ها، ارتباطات دیجیتال، شبکه های اطلاعاتی بی سیم
استاد دانشگاه بوستون
نویسنده کتاب گرانقدر Wireless Information Networks

۱.۱. خلاصه فصل

در عصر تکنولوژی اطلاعات و ارتباطات و جامعه دانش محور امروز، بی شک نرم افزار و حوزه کاربرد وسیع آن، از مناسبترین و بهینه ترین ابزارهای تجزیه و تحلیل اطلاعات، توزیع آسان و وسیع سرویسها و حل مسائل در ارائه خدمات مختلف می باشند. امروزه اندیشمندان و صنعت گران، با استفاده از نرم افزارهای کاربردی در حوزه های مختلف صنعت و تکنولوژی به راحتی توانسته اند ابزارهای مکانیزه مجتمعی را جهت ذخیره سازی، پردازش و بازیابی کارا و سریع حجم عظیمی از اطلاعات در قالب نرم افزارهای کاربردی ایجاد نموده و از دستاوردهای عظیم این نرم افزارها در جهت تسهیل فرآیندهای کاری پیچیده، افزایش کارایی و سرعت پاسخگوئی بهره مند گردند. مهندسی نرم افزار مبین اصول و قوانینی است که به بهینه سازی و مجتمع سازی روشها، ابزارها و روالهای انجام کار به منظور ارائه یک محصول نرم افزاری کارا و مفید می پردازد. تعاریف این اصول پایه، شرح جزئیات این روشها و معرفی الگوهای ساخت و تولید نرم افزارهای کارا و مقایسه آنها از مهمترین نکاتی است که در این فصل به آن ها می پردازیم.

۱.۲. مقدمه

امروزه با توجه به افزایش روزافزون کاربردها و خدمات نرم‌افزار، جایگاه نرم‌افزار و مهندسی نرم‌افزار اهمیت ویژه‌ای پیدا نموده است، بطوریکه در چهاردهه اخیر شاهد یک تحول عظیم در صنعت نرم‌افزاری در دنیا بوده ایم. امروزه بدون بهره‌گیری از خدمات و دستاوردهای عظیم صنعت نرم‌افزار ارائه خدمات کاربردی مختلف فرآیندی پیچیده، زمان‌بر و گاهی طاقت‌فرسا به نظر می‌رسد. نرم‌افزار^۱ محصولی است که به واسطه فرآیند مهندسی نرم‌افزار طراحی و ایجاد می‌شود و شامل مجموعه‌ای از دستورالعملها، ساختارهای داده‌ای، برنامه‌ها و اسناد و مدارک مرتبط مبین عملکرد و نحوه ارائه خدمات می‌باشد که کاربرد خاصی را اتوماتیک و مکانیزه می‌سازد و کنترل فعالیتهای سخت‌افزار و هدایت پردازش داده‌ها را بر عهده دارد. همچنین مهندسی نرم‌افزار ایجاد و استفاده از اصول مفید روشهای مهندسی به منظور دستیابی به نرم‌افزارهای قابل اطمینان و کارآمد را تحت یک فرآیند جامع تضمین می‌کند.

کتاب حاضر به عنوان راهنمائی جامع برای متخصصین این حوزه و دانشجویان دوره کارشناسی و کارشناسی ارشد و به عنوان یکی از مراجع تدریس در دانشگاهها با تاکید بر اصلاح و بازنگری برخی منابع موجود و با هدف ارائه روشها و ابزارهای جدید مهندسی نرم‌افزار و بکارگیری راهبرهای عملی توسعه نرم‌افزار در قالب یک فرآیند آزمایشگاهی و عملی تهیه گردیده است. فصل اول، به معرفی مفهوم نرم‌افزار و خصوصیات آن، اصول پایه مهندسی نرم‌افزار^۲، چرخه حیات نرم‌افزار، چرخه حیات ساخت یافته محصول و معرفی فازهای آن با شرح عملکرد هر فاز، پرداخته و دیدگاه مبتنی بر اصول ساخت یافته مهندسی نرم‌افزار را در ذهن خواننده ایجاد می‌نماید. بنابراین درک عمیق و تحلیل مفاهیم کلیدی بیان شده در این فصل، به منظور تفهیم بهتر مطالب فصول آتی، اکیدا توصیه می‌گردد.

۱.۳. مفاهیم نرم‌افزار و مهندسی نرم‌افزار

در سالهای اخیر، نرم‌افزار به عنصر کلیدی در تکامل محصولات و سیستم‌های مبتنی بر کامپیوتر تبدیل شده است. طی چهاردهه اخیر، نرم‌افزار از یک ابزار تحلیل اطلاعات و حل مسئله، به صنعتی زیربنائی و مستقل تکامل یافته است و عملاً همواره شاهد پویائی، تغییرات و بروزرسانی‌های سریع در ابزارها و تکنولوژی‌های مرتبط با این صنعت بوده ایم. هر نرم‌افزار یک عنصر منطقی، شامل مجموعه‌ای از برنامه‌های کاربردی، داده‌ها و مستندات می‌باشد. درواقع، نرم‌افزار محصولی است که به واسطه فرآیند مهندسی نرم‌افزار طراحی و ایجاد می‌شود و شامل مجموعه‌ای از دستورالعملها، ساختارهای داده‌ای، برنامه‌ها و اسناد و مدارک مرتبط مبین عملکرد و نحوه ارائه خدمات می‌باشد که کاربرد خاصی را اتوماتیک و مکانیزه می‌سازد و کنترل فعالیتهای سخت‌افزار و هدایت پردازش داده‌ها را بر عهده دارد. در

1 Software

2 Software Engineering

مقام مقایسه نرم افزار و سخت افزار، مجموعه ای از فاکتورها و مشخصات این دو مفهوم را از یکدیگر تفکیک می کنند که مهمترین این مشخصات شامل موارد زیر است:

● نرم افزار یک فرآورده منطقی است که مطابق اصول مهندسی نرم افزار ساخته می شود، درحالی که سخت افزار یک موجودیت فیزیکی است که بصورت کلاسیک با مجتمع سازی مجموعه ای از قطعات، تحت تکنولوژی خاصی ساخته می شود.

● سخت افزارها بصورت محصولات آماده ای که تحت پیکربندی و قطعات خاصی ساخته شده و موجودند، بر اساس قطعات مونتاژ می شوند درحالی که نرم افزارها مرحله به مرحله با پویایی زیادی که در ذات محصول سخت افزاری نمی توان دید با دستورالعمل مشتری ساخته و تکمیل می گردند و بر اساس ترجیحات کاربر سفارشی می شوند.

● سخت افزار در چرخه حیات خود دچار فرسودگی و استهلاک و نهایتاً از کارافتادگی می گردد، در حالی که نرم افزارها فرسوده و از کار افتاده نمی شوند بلکه دچار اختلال می گردند.

● در صورت از کار افتادگی قطعه سخت افزاری با محصول یدکی تعویض می گردد، در حالی که برای نرم افزار هیچ قطعه یدکی وجود ندارد.

امروزه انواع مختلفی از کاربردهای نرم افزار ایجاد گردیده که مهمترین رده های انواع این محصولات نرم افزاری شامل: نرم افزارهای تجاری، نرم افزارهای کنترلی، نرم افزارهای سیستمی، نرم افزارهای کاربردی کامپیوترهای شخصی، نرم افزارهای بلادرنگ، نرم افزارهای علمی مهندسی و نرم افزارهای هوش مصنوعی می باشند که آرایه عظیمی از کاربردهای تک منظوره یا همه منظوره را مجتمع سازی و سازماندهی می نمایند.

در سال های 1960 تا 1970 تولید نرم افزارها به بلوغ امروزی نرسیده بود و طراحی و توسعه محصولات نرم افزاری بدون بکارگیری اصول و مبانی فنی و توجه به سیاستهای ساخت یافته صورت می پذیرفت. بنابراین در این دهه مشکلی به نام بحران نرم افزار¹ ایجاد گردید. مهمترین دلایل بحران نرم افزار را می توان در موارد زیر خلاصه نمود:

1. هزینه بالای تولید نرم افزار
2. نرم افزارهای تولید شده تمام نیازهای مشتریان را برآورده نمی کرد.
3. تحویل نرم افزار در موعد مقرر انجام نمی شد.
4. پیشرفت سخت افزار به قدری سریع بود که نرم افزار نمی توانست با آن رقابت کند.
5. خطاهای موجود در نرم افزارها بسیار زیاد بود و برای رفع این خطاها مشکلاتی وجود داشت.
6. امکانات توسعه نرم افزار، پشتیبانی و نگهداری آن بسیار محدود بود.

حل مشکلات مرتبط با بحران نرم افزار منجر به ایجاد شدن شاخه جدیدی از علم به نام علم مهندسی نرم افزار گردید. مهندسی نرم افزار شاخه ای از علم است که به آشنایی با روشهای توسعه، تکنیکها و اصول پایه فرآیند ساخت و تولید قانونمند محصول نرم افزاری کارا، قابل اطمینان و با کیفیت می‌پردازد. این اصول و روشها، شامل مجموعه وسیعی از وظایف و فازهای کاری از جمله: تجزیه و تحلیل خواسته ها، طراحی، توسعه و ساخت برنامه ها، آزمایش و پشتیبانی و نگهداری محصول نرم افزاری تولید شده می‌باشند. اصطلاح مهندسی نرم‌افزار بعد از سال ۱۹۶۸ شناخته شد. این اصطلاح طی کنفرانس مهندسی نرم‌افزار ناتو در سال ۱۹۶۸ که در گارمیش آلمان برگزار شد توسط ریاست کنفرانس F.L. Bauer معرفی شد و از آن پس بطور گسترده مورد استفاده قرار گرفت. هدف مهندسی نرم افزار فراهم آوردن چارچوبی جامع برای ساختن نرم افزارهایی بهینه با کیفیت بالاتر است. ایجاد این چارچوب جامع توسط مهندس نرم افزار، منجر به ساخت محصول نرم افزاری با کیفیت و قابل اطمینان تحت اصول و قواعد پیش فرض می‌گردد.

محصول نرم افزاری نهائی شامل برنامه هائی است که در کامپیوتری به هر اندازه و با هر معماری و بستر سخت افزاری قابل اجرا هستند، همچنین این محصول مستنداتی دارد که شامل فرمهای کپی شده و مجازی می‌باشد و داده هایی دارد که ترکیبی از ارقام و حروف است و البته می‌تواند شامل اشکال نمایشی از قبیل اطلاعات تصویری، ویدئویی و صوتی باشد.

مهندس نرم افزار، براساس نیازمندی ها مجموعه ای از کارکردها و وظائف^۱ را تولید می کند که در کل، کاربرد خاصی را تشکیل می دهد، و الزاما باید در شرایط کار مشتری یا کارفرما تعریف شده و قابل قبول باشد. در این راستا توانائی درک صحیح مسئله و تجزیه تحلیل درست آن بمنظور تفکیک و واگذاری بخشها و وظائف کاری، به سایر اعضا تیم در طراحی و ساخت نرم افزار، ضروری می‌باشد. دو عامل مهم در مهندسی نرم افزار و پیشبرد صحیح اهداف پروژه، برآوردهای هزینه و زمان می باشد و یک مهندس نرم افزار خوب باید برآورد دقیق از هزینه و زمان بدست آورد. فرآیند تولید نرم افزار مطابق اصول مهندسی نرم افزار، هدف تولید محصول نرم افزاری با کیفیتی که در زمان مورد نظر و بر اساس بودجه ای که برآورد شده، در حد مطلوبی پاسخگوی نیازهای کاربر باشد را مدنظر دارد. یک تعریف جامع از انجمن IEEE مهندسی نرم افزار را در قالب یک تکنولوژی لایه ای شامل چهار لایه: ابزارها^۲، روشها^۳، فرآیندها^۴ و لایه تاکید بر کیفیت^۵ توصیف می‌کند که لایه ابزار پشتیبانی خودکار یا نیمه خودکار ماشینی را برای فرآیند و روش ها را فراهم می‌کند. لایه روش ها شیوه های فنی ساخت نرم افزار را مجتمع سازی می‌نماید. لایه

1 functionality

2 Tools

3 Methods

4 Processes

5 Quality Assurance

فرآیندها گامهای توسعه و ساخت نرم افزار را در قالب الگوهای قابل اطمینان سفارشی می‌نماید و در نهایت لایه تاکید بر کیفیت کوششهای سازمانی را روی نکات کیفی نرم افزارهای ایده آل و کارا مطابق ترجیحات کاربر متمرکز می‌نماید.

طی فرایند تولید محصول نرم افزاری، فعالیتهای مرتبط با مهندسی نرم افزار را بطور کلی در فاز به شرح زیر می‌توان طبقه بندی کرد:

فاز تعریف: بر چپستی مسئله تاکید دارد. به این معنی که طی فاز تعریف، مهندس نرم افزار می‌کوشد تا مسئله موردنظر و حوزه کاری، نهادها و موجودیتهای مرتبط با سیستم، محدودیتهای فنی و محیطی، نیازمندیهای اطلاعاتی و عملیاتی را تعریف و تعیین کند که چه اطلاعاتی باید پردازش شود، کدام عمل و کارایی مطلوب است، چه رفتارهای سیستمی قابل انتظار است، بین نهادهای شناسائی شده چه رابطه هایی را می‌توان برقرار نمود، چه نقاط تقابل و محدودیتهایی در طراحی وجود دارد و چه ملاکهای معتبر سازی برای تعریف یک سیستم موفق مورد نیاز است. همچنین در این فاز خواسته های کلیدی سیستم و نرم افزار تشخیص داده می‌شود. گرچه روش های اعمال شده طی فاز تعریف، به الگو (یا ترکیبی از الگوهای) مهندسی نرم افزار بستگی دارند، در این راستا سه وظیفه عمده صورت می‌پذیرد :

7. مهندسی اطلاعات یا سیستم

8. طرح ریزی پروژه نرم افزار

9. تحلیل خواسته ها

فاز توسعه: بر چگونگی فرآیند انجام کار تاکید دارد، یعنی در خلال توسعه محصول نرم افزاری، مهندس نرم افزار می‌کوشد تا تعیین کند داده ها چه ساختاری داشته باشند (ساختار داده ای)¹، عملیات چگونه در یک معماری نرم افزاری صحیح پیاده سازی شوند، واسطهها² چه ویژگیهایی داشته باشند، طراحی چگونه به یک زبان برنامه نویسی ترجمه شود و آزمایش ها به چه نحوی انجام شوند. دیدگاهها و روش های قابل اعمال طی فاز توسعه متغیرند، ولی سه وظیفه فنی همواره باید به انجام برسد:

10. طراحی نرم افزار³

11. تولید دستورها

12. آزمایش نرم افزار

فاز پشتیبانی و نگهداری محصول: پس از واریسی فنی و رسمی^۱ بر فعالیتهای فاز توسعه توسط تیم ساخت و توسعه محصول نرم افزاری در صورت برقراری نقاط تایید می‌توان بر تغییراتی که با تصحیحات مورد نیاز در جهت تکامل محیط نرم افزار در ارتباط هستند و نیز تغییراتی که ناشی از تغییر خواسته های مشتریان هستند، تاکید دارد. طی فاز پشتیبانی چهار نوع تغییر قابل مشاهده است که چهار استراتژی نگهداری محصول نرم افزاری را تعریف می‌نماید :

نگهداری تصحیحی: حتی در صورت انجام بهترین کارها برای حصول اطمینان از کیفیت، باز هم این احتمال وجود دارد که مشتری معایبی را در نرم افزار بیابد. نگهداری تصحیحی، نرم افزار را در جهت تصحیح معایب تغییر می‌دهد.

نگهداری تطابقی: به مرور زمان، محیط اولیه (مثل CPU، سیستم عامل، قواعد کار، ویژگیهای محصول خارجی) که نرم افزار برای آنها بسط یافته است، احتمالاً تغییر می‌کند. نگهداری تطابقی منجر به انجام اصلاحاتی در نرم افزار می‌شود به نحوی که پاسخگوی تغییرات محیط خارجی خود شود.

نگهداری بهبودی: به موازاتی که نرم افزار مورد استفاده قرار می‌گیرد، مشتری/کاربر متوجه عملکردهایی می‌شوند که افزودن آن ها موجب بهتر شدن نرم افزار می‌شود. نگهداری بهبودی نرم افزار را به فراسوی خواسته های اولیه آن ها یا ترجیحات کاربر سوق می‌دهد.

نگهداری جلوگیری از افت نرم افزار: نرم افزار کامپیوتری در اثر تغییرات زیاد، کارایی خود را از دست می‌دهد و از این رو، نگهداری پیشگیرانه که غالباً مهندسی مجدد(مهندسی معکوس) نرم افزار نامیده می‌شود، باید اجرا شود تا نرم افزار نیازهای کاربران نهایی خود را بر آورده سازد. در اصل نگهداری پیشگیرانه، تغییراتی را در نرم افزارها و برنامه های کامپیوتری اعمال می‌کند که بتوان آنها را راحت تر تصحیح کرد، تطابق داد و بهبود بخشید.

۱.۴. چرخه حیات نرم افزار

همانگونه که در دنیای مهندسی برای تولید هر محصولی مراحل وجود دارد، مهندسی نرم افزار نیز از این قاعده مستثنا نیست. به عبارت دیگر جهت تولید نرم افزار، چرخه ای وجود دارد که روند تولید نرم افزار را به صورت قانونمند و اصولی به مراحل تقسیم کرده است. این چرخه را به نام چرخه حیات نرم افزار^۲ می‌شناسیم.

در چرخه حیات نرم افزار هر مرحله کار دارای تعریف و ضوابط مشخص می باشد. همچنین قوانین و فعالیت های کلیدی که باید در هر مرحله انجام شود تعریف می شوند. ورودی های هر مرحله کاملاً

1 Formal And Technical Review

2 Software Life Cycle

مشخص می شود و همچنین خروجی هایی را که باید از هر مرحله دریافت نماییم، شفاف است. برای هر مرحله نقاط کنترلی^۱ تعریف می شود که بواسطه آنها میزان پیشرفت کار کیفیت کار انجام شده و ... مشخص خواهد شد.

چرخه حیات نرم افزار عبارت است از تمامی مراحل و گامهایی که مطابق اصول مهندسی نرم افزار طی می شود تا یک محصول نرم افزاری مفید و با کیفیت تولید شود. این مراحل عبارتند از:

13. برنامه ریزی استراتژیک^۲: در این مرحله با توجه به امکانات و محدودیت های حاکم بر شرایط به تدوین استراتژی های توسعه می پردازند و در مورد اینکه آیا یک نرم افزار تولید شود یا نشود، ترجیحات ساخت، خرید، استفاده از مولفه های آماده و ... تصمیم گیری می شود.

14. امکان سنجی^۳: از آنجا که برای اجرای چارچوب های حاصل از برنامه ریزی استراتژیک، شناسایی محیط، امکانات و محدودیتهای موجود ضروری می باشد از این رو قبل از هر اقدامی باید شرایط دستیابی به نرم افزار مورد نظر از جهت نیازمندیهای هزینه و زمان، امکانات فنی، نیروی انسانی مورد نیاز و ... مورد سنجش واقع شود و قبل از هر صرف هر نوع انرژی و زمان امکان پذیر بودن آن احراز گردد. هرچند که بعضی اوقات برای حفظ مزیت های رقابتی یا تطبیق با مقررات نیاز به تولید نرم افزار می باشد.

15. تحلیل سیستم ها^۴: پس از اینکه امکان تولید نرم افزار احراز گردید مرحله تشخیص و تعیین نیازهای کاربران و نیازمندی های سیستم آغاز می گردد. مساله اصلی در این مرحله این است که سیستم جدید باید به چه نیازهایی پاسخ دهد. در این مرحله تمرکز بر این است که سیستم چه چیزی را باید به عنوان خروجی تحویل دهد. باید حد و مرز و ماهیت نیازها با شفافیت و به دور از هر گونه ابهام مشخص شود و کارکرد اصلی محصول نرم افزاری و سرویسها و خدمات پایه آن دانشاسائی و تحلیل گردد.

16. طراحی نرم افزار (سیستم)^۵: طراحی، پس از مرحله تحلیل و بر اساس شناسایی دقیق نیازهای کاربران، شکل عملیاتی سیستم و نحوه پاسخگویی به نیازهای شناخته شده طرح ریزی می شود و به ترتیبی که نیازهای کاربران را در قالب مشخصات سیستم و براساس موجودیتهای، کلاسها، زیرسیستم ها و درنهایت سیستم ارائه خدمات بیان کند، بصورت طراحی اشیاء، بسته ها، روالها، واسطها و عملیات مستقل، صورت می پذیرد.

17. پیاده سازی نرم افزار (سیستم)^۶: طراحی سیستم، برنامه را برای ساخت و آزمایش و ارائه سیستم پایه ریزی می نماید. سپس در این مرحله ضمن نهایی شدن برنامه ساخت سیستم و تولید کد منبع و

1 Check points

2 Strategic Planning

3 Feasibility Study

4 Requirements Analysis

5 System Design

6 Implementation