دانشکده مهندسی کامپیوتر و فناوری اطلاعات

سامانه شمس

(شبیه ساز مدیریت سلامت)

نام استاد

دکتر عبدالله زاده

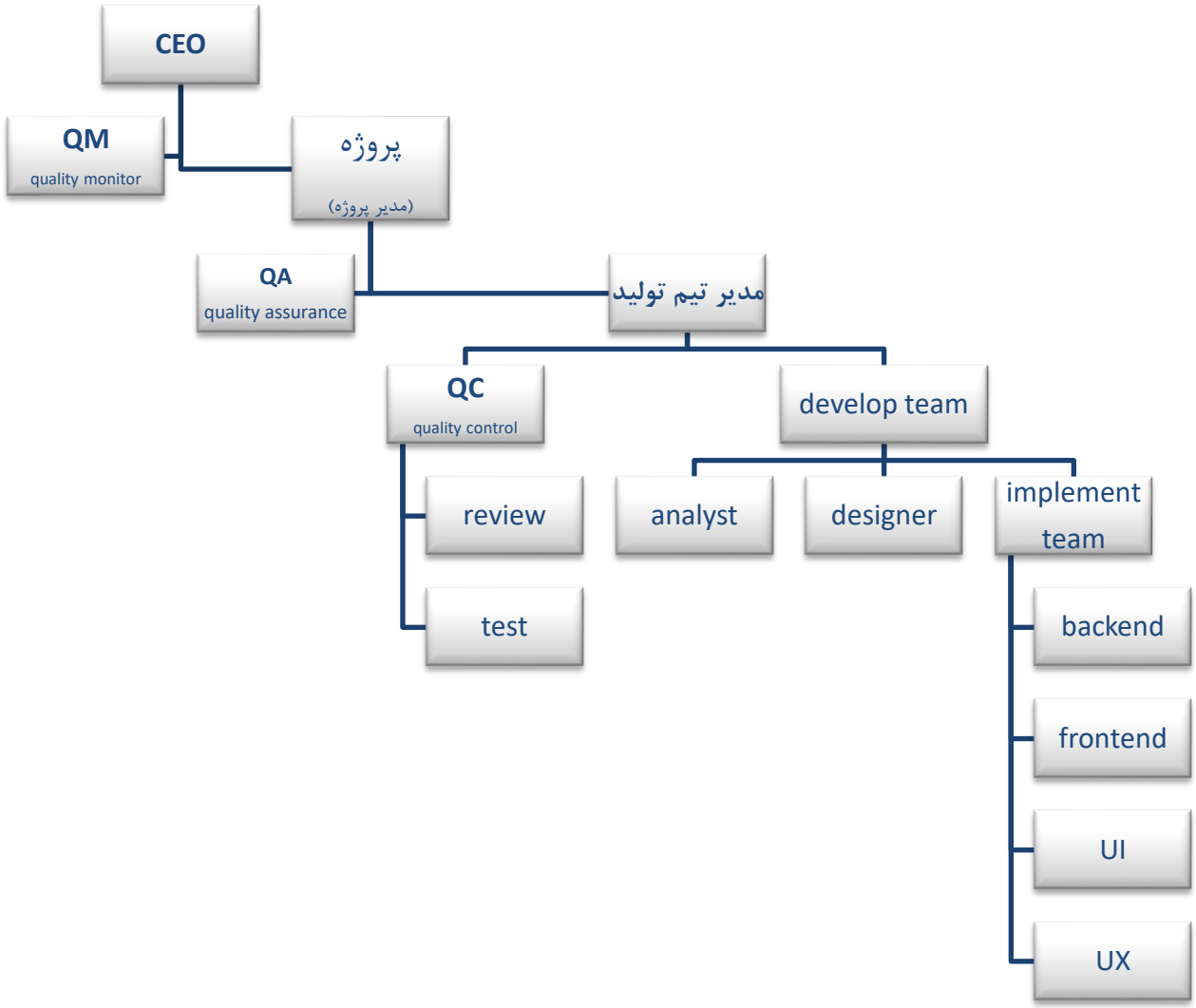
نام دانشجو

زهرا دهقانیان

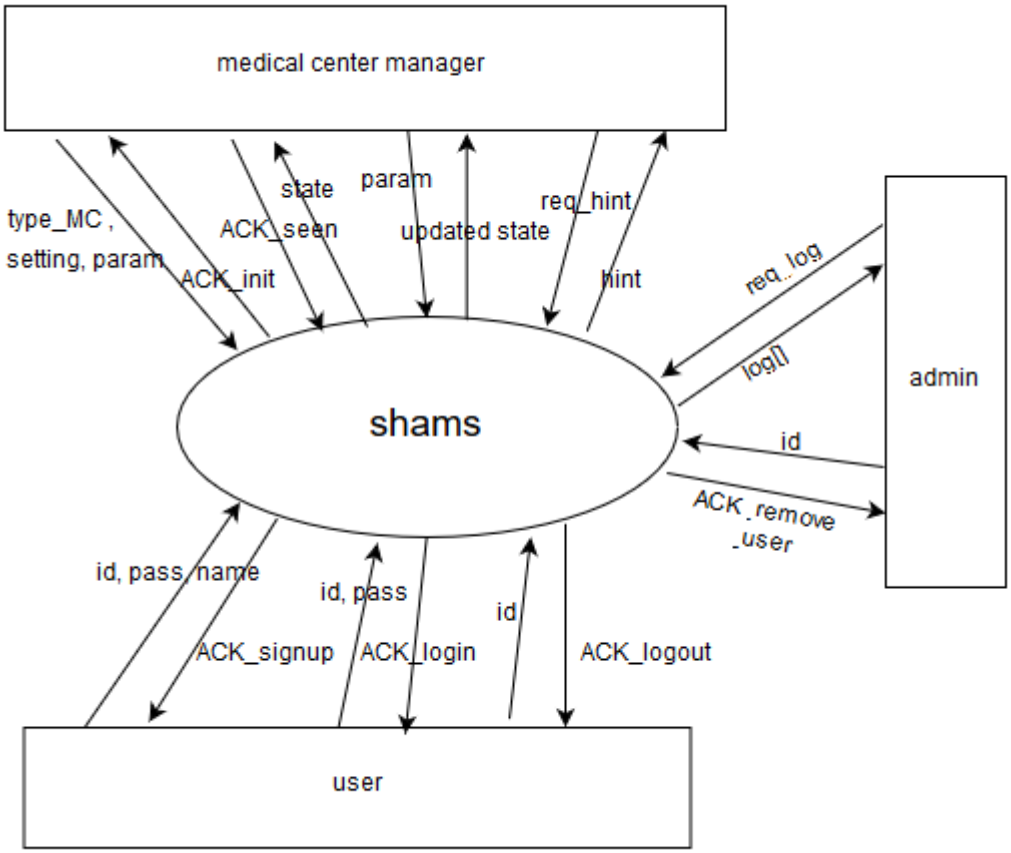
پاییز ۱۳۹۷

فهرست

ساختار سازمانی



مدل مفهومی پروژه



لایه های مهندسی نرم افزار

مهندسی نرم افزار به طور کلی شامل ۵ لایه زیر میباشد، در ادامه به W5H2 برای هر کدام پاسخ میدهیم :

نام لایه	W5H2	توضیحات
کیفیت	**What**	رسیدن به مدل کیفی تعریف شده برای نیازمندی های مشخص شده در مرحله inception (نه بیشتر و نه کمتر)
	When	به طور کلی میتوان گفت در تمام طول عمر پروژه (فعالیت چتری) باید انجام شود. اما هر کدام از بخش ها، مسیول تامین وظایف متفاوتی در lifr cycleپروژه جهت رسیدن به مدل کیفی هستند .
	Where	در تمامی نرم افزار های تولید شده توسط متدولوژی RUP باید کیفیت در نظر گرفته شود
	Who	تمامی افراد در تیم از جمله مهندسین نرم افزار و تیم تولید، مدیران و ذینفعان و به طور خاص بخش های QMدر تیم مشتری و QA و QC در تیم نرم افزاری مسیول بررسی و تضمین کیفیت هستند.
	Why	جهت جلوگیری از دوباره کاری و کاهش هزینه ها و کاهش زمان تولید
	How	برای رسیدن به یک نرم افزار با کیفیت باید ۴ فعالیت انجام دهیم : از فرایند های مهندسی نرم افزار ثابت شده استفاده کنیم. فعالیت های مدیریت پروژه را با قوت

		انجام دهیم.کنترل کیفی همه جانبه داشته باشیم و از زیر ساخت قوی تضمین کیفیت (QA) بهره ببریم.
	How much	تا رسیدن به سطح کیفی و اندازه های تعیین شده در quality model
مدل فرایند	**What**	مدل فرایند به طور کلی تمامی مراحل و مسیر و فعالیت های تولید سیستم از اولین قدم ها تا تولید نهایی را برای ما مشخص میکند.یا به طور کلی W5H2 را در هرگام برای ما روشن میکند، در این جا RUP میباشد
	When	همان طور که گفته شد process model تمامی فعالیت ها از اولین قدم یعنی مهندسی نیازمندی ها تا آخرین قدم یعنی maintenance را برای ما مشخص میکند
	Where	مدل فرایند وظایف تمامی اعضای تیم را در تمامی مراحل مشخص میکند و در هر شرکت نرم افزاری که قصد تولید یک سیستم مهندسی باکیفیت اصولی (نه شانسی) را دارد به کار می آید.
	Who	role های مورد نیاز در RUP شامل: آنالیزر سیستم، طراح UI ، طراح دیتابیس، integrator، برنامه نویس، معمار نرم افزار(software architecture) و ...
	Why	همان طور که گفتیم مدل نیازمندی برای ما رویکرد و رویکرد برای ما متدولوژی و به همین ترتیب مدل فرایند تعیین میشود. پس به جهت تامین دقیق و منطبق بر نیازمندی ها از مدل فرایند استفاده میکنیم.
	How	مراحل انجام RUP شامل inception, elaboration , construction , transition, production
	How much	**فعالیت های تعریف شده در مراحل مختلف RUP باید کاملا منطبق بر فرایند ها و به میزان تعیین شده در مدل کیفی انجام شود.**
روش	**What**	این لایه شامل جزییات دقیق هر کدام از مراحل مدل فرایند است. به عنوان مثال Brain storming یا مصاحبه یک از روش های communication است.
	When	در هر گام از پروژه باید از دستورالعمل و روش موجود استفاده کنیم
	Where	برای هر کدام از مراحل در مدل فرایند روش های مختلفی وجود دارد که درهر پروژه با توجه به محدودیت های پروژه و زمان و هزینه یک سری از این روش ها validate و verificate میشود.
	Who	مسیول انجام هر گام (با توجه به روش مشخص شده) معلوم است مثلا در فاز inception برای مشخص سازی نیازمندی ها باید ذی النفعان و technical writer و requirement reviewer وrequirement specifier است.

	Why	علت استفاده از روش های دقیق و مشخص تضمین عملکرد و صحت انجام پروژه و عدم وجود خطا در محصول نهاییست.
	How	همان طور که گفته شد در هر گام از روش های تعیین شده در هر بخش استفاده میکنیم تا آن گام را با کیفیت لازم به انجام برسانیم.
	How much	فعالیت های تعریف شده باید طبق روش های تعیین شده و به میزان مشخص شده در مدل کیفی انجام شود.
ابزار	**What**	ابزارها به طور کلی نرم افزار هایی هستند که فرایند های انجام پروژه را به صورت خودکار و یا نیمه خودکار انجام میدهند و روند پروژه را تسهیل میبخشند.از ابزار های rup میتوان به Rational Requisite®Pro برای track کردن نیازمندی های پروژه و یا ClearQuest™ برای مدیریت تغییرات
	When	زمان استفاده از ابزار ها در بخش های مختلف با توجه به نیازمندی های هر بخش تعیین میشود.
	Where	در هر گام پروژه به منظور تسهیل فعالیت ها میتوان از ابزارها استفاده کرد
	Who	تمامی role های تعیین شده در بخش های مختلف ، از ابزار ها برای انجام مسیولیت محول شده میتوان بهره جست.
	Why	به منظور تسهیل و تسریع و کاهش هزینه و زمان در فرایند های تعریف شده در مدل فرایند
	How	نحوه استفاده از هر کدام از ابزار ها با یکدیگر متفاوت است اما در اغلب این نرم افزار ها ابتدا اطلاعات مربوطه را وارد میکنیم و آن نرم افزار نمودار و یا تخمین و یا ... را برای ما محاسبه میکند.
	How much	میزان استفاده از ابزار ها در هر بخش همان طور که گفته شد تا زمانی است که به سطح کیفی و خروجی موردنظر آن گام با شاخص های کیفی مدنظر برسیم

رویکرد انجام پروژه

در هر سیستم ، با توجه به نیازمندی های سیستم و کیفیت مورد انتظار stack holder ها یک رویکرد مناسب برای انجام پروژه است. نحوه انتخاب این رویکرد به گونه ایست که محصول نهایی در عین داشتن کیفیت ، در زمان معین و با هزینه معین به سرانجام برسد و تیم پروژه با شکست روبرو نشود .

	Structured	Object-oriented	Agile
نحوه طراحی	در طراحی ساختار یافته معمولا با رویکرد بالا به پایین عمل میکنیم و از طراحی کلی ماژول ها به زیر بخش های کوچک میرسیم.	در این رویکرد، ارتباط بین O های مختلف وجود دارد و وظیفه طراح پیاده سازی روابط با object-oriented design است	در این رویکرد انرژی زیادی صرق طراحی نمیشود و روند تولید به صورت : طراحی مجدد – برنامه ریزی و پیاده سازی است
نقش user	کاربر نقش خاصی در روند تولید ندارد	کاربر در بخشی از روند تولید برای تولید همکاری میکند	همکاری کاربر به شدت در تمامی مراحل نیاز است.
بازخورد به تغییرات	نسبت به تغییرات در روند تولید اصلا پاسخگو نیست	تا حد خوبی تغیرات را پوشش میدهد	تغییرات را پوشش میدهد
reusability	در این رویکرد استفاده مجدد، مورد توجه نیست	یک از اصول اساسی در این رویکرد مهندسی است	تا حدی به این اصل توجه میکند.
خوبی و بدی	۱. بسیار محبوب ۲. ساختار ساده ۳. عدم پشتیبانی از تغیرات ۴.ریسک پذیری کم ۵.مناسب برای پروژه با نیازمندی ثابت و مشخص ۶.توجه بسیار به documentaion	۱. مناسب maintain ۲. کاهش هزینه ۳. کاهش زمان توسعه ۴. کاهش resource های موردنیاز ۵. ریسک پذیری بالا	۱. بسیار تغییرپذیر ۲.وابستگی خیلی زیاد به کاربر ۳.مناسب maintain ۴.عدم آشنایی تیم تولید

با توجه به توضیحات از هر رویکرد ، رویکرد منتسب برای این پروژه به نظر رویکرد شی گرایی می باشد.

علت این انتخاب نیز با توجه به بالا به شرح زیر است:

۱)ایجاد تغییرات با هزینه و زمان کمتر

۲)مستند سازی به کمک زبان UML

۳)توجه به رضایت مشتری به میزان کافی

۴)آشنایی تیم تولید با این رویکرد و کاهش هزینه آموزش

۵)قابلیت تطابق با نیازمندی و منابع موجود در پروژه

۶)وجود ابزار های بسیار متنوع خودکار و نیمه خودکار همانند : Rational SoDA® برای مستندسازی خودکار پروژه و یا Rational Purify® برای بررسی خطاهای زمان اجرا

از میان محصولات متفاوت این رویکرد ، به نظر رویکرد RUP که از ابتدا تا انتهای فرایند تولید را شامل شده و تمامی نیازمندی ها را پوشش میدهد و توانایی انطباق با سایز پروژه را دارد ، گزینه مناسبی می باشد.

متدولوژی پروژه

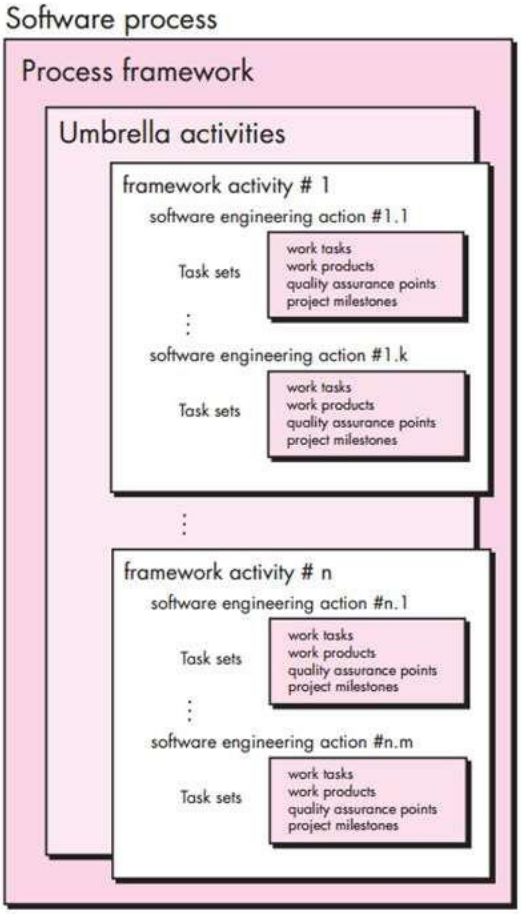
برای پیاده سازی این پروژه از رویکرد شی گرایی و از محصول RUP این رویکرد استفاده میشود . این متدولوژی تا حد خوبی قابل استفاده برای پروژه ها با هر سایزی هست. طبق دسته بندی IBM در مقاله " Using RUP to manage small projects and teams" که در آدرس زیر[1] قابل دسترسی است، این پروژه از دسته پروژه های کوچک محسوب میشود که در همین مقاله نحوه بکارگیری این متدولوژی شرح داده شده است. در این جا ما نیز از متدلوژی RUP به عنوان یک framework کلی و یک guideline برای انجام پروژه استفاده میکنیم.

روند انجام این پروژه به این صورت خواهد بود که مراحل مدل فرایند را منطبق بر فازبندی RUP انجام میدهیم (inception-elaboration-construction -...) اما در حین اجرای هر مرحله تیم را ملزم به تولید تمامی مستندات و artifact های موجود در فازهای RUP نمیکنیم. در واقع از اصول agile برای همگام سازی این متدولوژی با سایز پروژه خود همانند "توجه بیشتر به کد تا مستندات" نیز استفاده میکنیم. البته واضح است که این بدان معنا نیست که هیچ گونه مستنداتی در فاز های پروژه ایجاد نشود ، بلکه یعنی تنها مستنداتی را ایجاد میکنیم که بقیه فازهای پروژه منطبق بر آن هاست به عنوان مثال در فاز ابتدایی inception برای شناسای دقیق نیازمندی ها و actor های پروژه از نمودار usecase استفاده میکنیم.

[1] آدرس : https://www.ibm.com/developerworks/rational/library/jul05/kohrell/kohrell-pdf.pdf

چارچوب فرآیند

به طور کلی چارچوب و یا Framework شامل تعدادی فعالیت است که این فعالیت ها یا همیشگی هستند و در تمامطول حیات پروژه انجام میشوند (umbrella) و یا نستند ، هر فعالیت (activity) از تعدادی action تشکیل شده و در نهایت هر action به تعدادی task شکسته میشود. چارچوب در هر پروژه ای باید متناسب با رویکرد و متدولوژی منتخب برای انجام آن پروژه و نیاز های پروژه و اهداف آن تعریف میشود. در این پروژه به دلیل تجربه کم تیم توسعه از چهارچوب رایج RUP استفاده میشود . این چهارچوب عبارت است از :



- Inception : هدف اصلی این بخش بدست آوردن یک درک کلی از پروژه است یعنی بررسی محدوده پروژه، نیازمندی های سیستم ، و ایجاد یک نقشه و برنامه اولیه برای تولید است. در این فاز توافقات اولیه با مشتری و محدودیت ها و نیازمندی های اجرایی پروژه بررسی میشود .
- Elaboration : هدف اصلی این بخش مدلسازی پروژه دارد . در این بخش معماری سیستم شکل گرفته و کامل میشود . در این مرحله یک بررسی کامل روی نقشه طراحی شده انجام میشود تا از ایجاد هرگنه هزینه های احتمالی اضافی و هر نوع ریسک جلوگیری شود .
- Construction : هدف این بخش کاملا مشابه مرحله ی پیاده سازی در مدلهای عمومی فرآیندهاست که به کمک معماری دریافتی از مرحله قبل ، component های قابل اجرا و قابل تحویل به مشتری می سازیم؛ با این تفاوت که این مرحله یک مرحله افزایش-تکرار است و قطعات سیستم به صورت مرحله به مرحله کامل میشوند. قابل توجه است که در پایان هر تکرار به یک نسخه از نرم افزار که توانایی برآورده کردن نیازمندی خاصی را دارد ، دست می یابیم . در این مرحله تست های مختلف مانند unit test، integration test، acceptance test انجام میشود .

- Transition : هدف این بخش به طور کلی آموزش کاربران و همینطور دریافت بازخورد کاربران است در این مرحله نسخه ای اجرایی(بتای نرم افزار) برای تست در اختیار کاربران سیستم قرار میگیرد . کاربران نیز خطاهای سیستم و تغییرات مورد نیاز آن را به اطلاع تیم توسعه نرم افزار می رسانند . در این مرحله تغییرات مهم کاربران اعمال میشود و نسخه نهایی برای release آماده میشود.
- Production: هدف از این بخش در مجموع فراهم سازی پشتیبانی از نرم افزار است . این فاز از RUP همانند مرحله deployment در مدل عمومی فرآیندهاست که وظیفه پشتیبانی از نرم افزار و اعمال نظرات و تغییرات (در چارچوب قرارداد تعیین شده) را دارد .

ورودی و خروجی های process framework activity ها

به طور کلی خروجی هر مرحله ورودی مرحله بعد خواهد بود به طوری که در اولین مرحله نیازمندی های خام مشتری به ما تحویل داده میشود و در نهایت خروجی مرحله آخر باید نرم افزاری کاملا منطبق بر نیازمندی ها مشتری تحویل وی داده شود.

- Inception :

ورودی	-نیازمندی های ابتدایی مشتری
خروجی	-مدل نیازمندی - usecase – برنامه/ plan پروژه – زمان بندی/ scheduling پروژه –معماری/ architecture ابتدایی پروژه – تخمین هزینه کلی پروژه – مستند تحلیل ریسک

- Elaboration :

ورودی	-مدل نیازمندی -usecase - معماری ابتدایی پروژه
خروجی	-Usecase کامل - معماری دقیق پروژه – مدل baseline - زمانبندی دقیق -مستند تخصیص منابع – مستند تحلیل ریسک دقیق(با interval ریسک کمتر) – مدل کسب و کار

- Construction :

ورودی	-معماری دقیق پروژه – مدل طراحی کلاس
خروجی	-پیاده سازی component/ بخش های سیستم -تست واحد/ unit -تست integration / ادغام –سیستم تولیدشده اولیه (اتصال کامپوننت ها به یکدیگر)

- Transition:

ورودی	-تست integration –تست unit – سیستم تولید شده اولیه
خروجی	-سیستم تست شده و تکمیل شده – اطلاعات تکمیلی محصول شامل توصیف سیستم و اطلاعات موردنیاز کاربران

- Production :

ورودی	-سیستم تست شده - اطلاعات مورد نیاز کاربر
خروجی	-Defect های سیستم - تغییرات پیشنهادی برای نسخه بعدی نرم افزار

ذینفعان سیستم

ذینفعان و یا stackholder ها تمامی افرادی هستند که به نحوی نیازمندی هایشان در سیستم تاثیرگذار است و در طول پروژه سود می برند و در توسعه نرم افزار نقشی را ایفا میکنند .

در این سیستم مدیران بیمارستان های عضو سایت و یا تمامی کاربرانی که در سامانه عضو میشوند از ذینفعان اصلی محسوب میشوند . همچنین صاحب اصلی سیستم که سرمایه گذار برای این مجموعه محسوب میشود نیز از ذینفعان این سیستم محسوب میشود . علاوه بر این موارد ، در تیم تولید ذینفعان متعددی وجود دارد که از جمله آنها میتوان به تیم designer , tester , programmer , maintainer و.... اشاره کرد . در زیر لیست تمامی ذی النفعان آمده است :

- مدیران بیمارستان (کاربران سایت)
- سرمایه گذار
- مدیر پروژه
- مدیر تیم تولید
- اعضای بخش QA
- اعضای بخش QC (tester)
- اعضای بخش QM
- برنامه نویس (UI/UX – backend – frontend)
- گرافیست
- روانشناس

مدل نیازمندی های سیستم

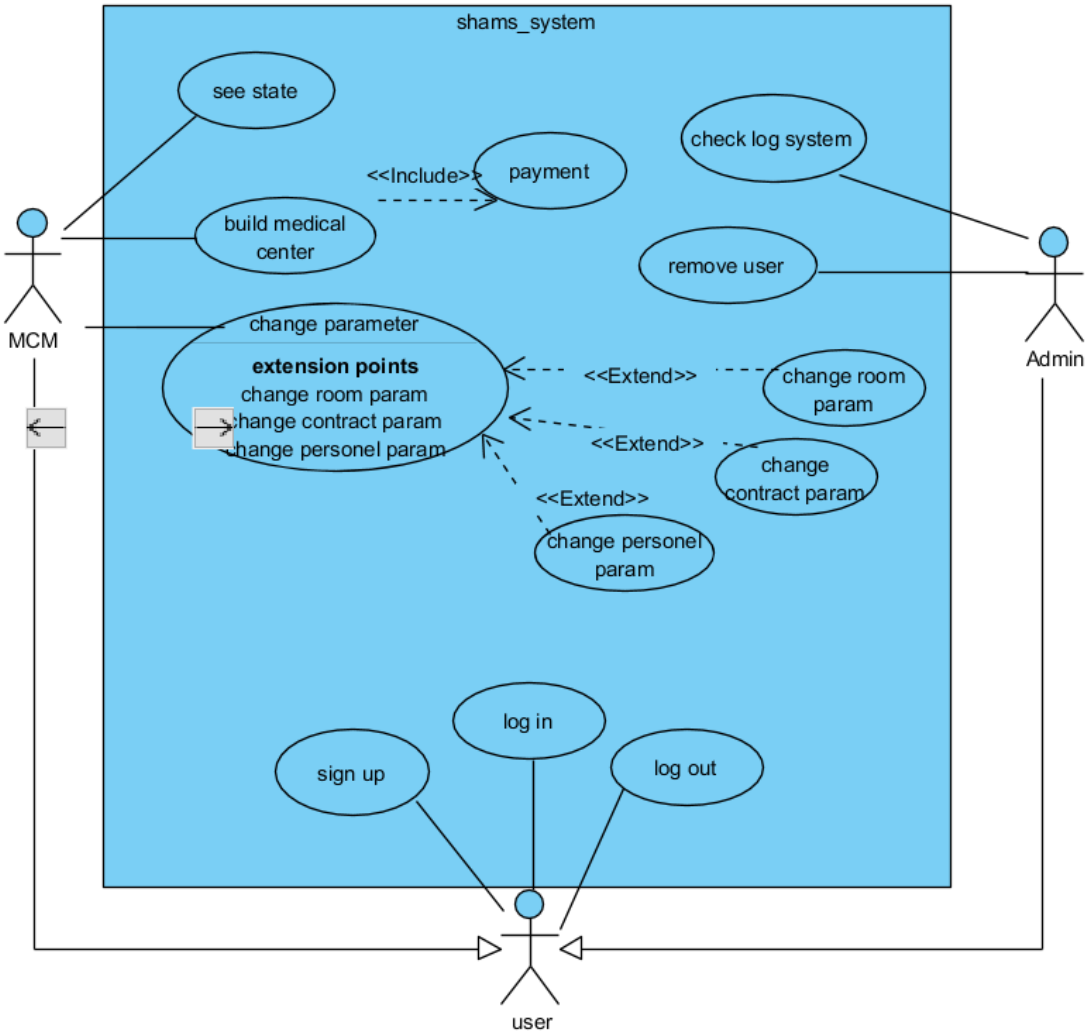
در این سیستم هدف ایجاد یک نرم افزار تحت وب است که شبیه ساز مدیریت بیمارستان با هدف ایجاد بستری مناسب برای آموزش و بدست آوردن تجربه مدیریت پیش از ورود به بازار کار است .

در زیر لیست نیازمندی ها functional دریافتی از مشتری پس validate و verify ارایه میشود.

- مدیران بیمارستان ها و همین طور admin سیستم میتوانند به عنوان کاربر در سیستم عضو میشوند .
- هر کاربری میتواند پس از آن به سیستمsignup و login و logout انجام دهد
- یک MCM و یا مدیر مرکز بهداشتی پس از ورود به سیستم میتواند یک بیمارستان مجازی بسازد
- یک MCM بعد از ساخت مرکز بهداشتی میتواند در بازه های زمانی مشخص تغییراتی که در بیمارستان ایجاد شده را مشاهده کند.
- یک MCM میتواند پارامتر های کنترلی خود (مانند پرسنل هر بخش، قرارداد های برون سپاری بخش ها ، وضعیت اتاق ها و ...) را تغییر دهد.
- Admin سیستم به طور کلی وظیفه نظارت بر روی سیستم را دارد و میتواند در هر مقطع زمانی log تمامی transaction های بازی را ببیند.
- Admin سیستم میتواند با فرستادن اطلاعات کاربری هر user وی را حذف کند

**در فاز بعدی لاگ های این بازی برای train کردن یک شبکه عصبی به جهت شبیه سازی رفتار مدیران بیمارستان ها بکار خواهد رفت.

با توجه به نیازمندی های اولیه میتوان از usecase برای نمایش نیازمندی ها به عنوان مدل نیازمندی ها استفاده کرد : c



دسته بندی نیازها

برای دسته بندی نیازمندی ها میتوان به روش های مختلفی عمل کرد . در این پروژه دسته بندی استفاده شده همان استاندارد FURPS+ است و با توجه به کم تجربه بودن تیم توسعه در این استاندارد تغییری داده نمیشود و تنها نیازمندیهای مختلف را در این قالب جای میدهیم و در بخش های بعدی برای هر کدام از این نیازمندی ها متریک و indicator مناسب تعریف خواهیم کرد :

Type Requirment	Parameter
Functionality	- ثبت نام در سیستم - خروج از حساب کاربری - تعریف مرکز بهداشتی - مشاهده تغییراتstate - تغییر تنظیمات اتاق - تغییر تنظیمات پرسنل - تغییر تنظیمات قرارداد
Usability	- حفظ یکپارچگی سایت - وجود راهنمای مفید برای کار با سیستم برای افراد جدید - ظاهر مناسب و قابل فهم برای کاربران
Reliability	- زمان ریکاوری و یا پاسخ به خطا مناسب در سیستم(کمتر از ۱ ثانیه) - تولید خروجی های مناسب در زمان ایجاد خطا

Performance	- زمان پاسخ دهی مناسب (کمتر از ۰٫۵ ثانیه) - امکان استفاده چندین کاربر به صورت همزمان از سیستم
Support	- قابلیت نگهداری آسان سیستم - قابلیت گسترش سیستم و اعمال تغییرات در عملیات های سیستم

مستند SRS

هدف کلی از ایجاد SRS و یا System requirements specification رفع ابهامات موجود در نیازمندی های اولیه و با هدف ایجاد یک تعریف دقیق ، کامل و یکسان برای تعامل بین تولید کننده سیستم و مشتری است .

- مشخصات مستند :

نام مستند	مستند SRS
نام پروژه	سامانه شبیه ساز مدیریت سلامت
مسیول پروژه	مهندس زهرا دهقانیان
مشتری	دکتر عبدالله زاده
نسخه مستند	۱,۰
تاریخ	۹۷,۹,۲۷

- **معرفی پروژه :**

با توجه به نرخ رشد چند درصدی جمعیت کشور و افزایش نیاز به مراکز بهداشتی و حساسیت مدیریت این بخش ها نیاز به وجود چنین بستری بسیار ضروری بود . این پروژه در شهریور سال ۱۳۹۷ به درخواست واحد پزشکی جهاد دانشگاه تهران کلید خورد.

- **هدف پروژه** :

اولین هدف از تولید این نرم افزار فراهم سازی بستری مناسب برای آموزش مدیران مراکز بهداشتی آینده است .زیرا این سمت، بسیار حساس است و هر نوع اشتباهی ، هر چند کوچک ممکن است سبب ایجاد خساراتی جبران ناپذیر شود. در این پروژه هم چنین ، هدف ایجاد یک آزمایشگاه برای مدیران فعلی بیمارستان هاست . تا هر تغییری که مدنظرشان است ابتدا در این بستر تست کنند و در صورت مناسب بودن بازخورد ها ، این تغییرات را در مراکز بهداشتی خود نیز اعمال کنند.

- **محدوده پروژه :**

محدوده و scope این پروژه ، شبیه سازی بیمارستان های کشور ایران با درجه بندی ۳ منطقه ای (مناطق مرکز استان = منطقه۱ مناطق شهری=منطقه۲ مناطق روستایی = منطقه ۳) خواهد بود .

- **اصطلاحات:**

در این پروژه منظور از MCM ، medical center manager و یا مدیر مرکز بهداشتی ، هر فردی اعم از مدیران فعلی بیمارستان ها/ درمانگاه ها و یا متقاضیان این سمت ها و یا هر فردیست که با این نرم افزار کار میکند. منظور از مرکز بهداشتی نیز یکی از مکان هایی مانند بیمارستان ، خانه بهداشت ، درمانگاه است که در ابتدای بازی باید configure شود.

- **محدودیت ها :**

۱- در مود آنلاین نمیتوان سرعت و یا pace بازی را تغییر داد و این قابلیت تنها در حالت تک نفره و نه در حالت گروهی قابل استفاده است.

۲- در حالت تک نفره سرعت بازی تا ۴ برابر و نه بیشتر قابل افزایش است

۳- هر MCM تنها میتواند ، ۵ مرکز بهداشتی به طور همزمان مدیریت کند

۴- در صورت عدم فعالیت MCM برای ۲ ماه ، حساب کاربری وی حذف خواهد شد.

۵- این پروژه حتما باید با دیتابیس 2016 پیاده سازی شود.

- **رابط کاربری :**

این نرم افزار شامل ۵ رابط کاربری است :

۱- رابط کاربری ثبت نام : در این رابط هر کاربر سیستم ابدا باید یک profile برای خود بسازد و سپس میتواند به سایر قابلیت های این سیستم دست یابد

۲- رابط کاربری configuration : این رابط وظیفه ایجاد یک فضای مناسب برای ساخت دقیق یک مرکز بهداشتی مطابق با واقعیت را در اختیار کاربر قرار دهد

۳- رابط کاربری admin : این رابط به طور خاص برای admin سیستم ساخته شده و در هر لحظه خلاصه این از وضعیت سیستم ، لاگ فعالیت ها و قابلیت حذف هر یک از کاربران ر ا در اختیار ادمین برنامه میدهد.

۴- رابط کاربری مدیریت پارامتر ها : این رابط کاربری برای MCM ها به منظور کنترل و اعمال تغییرات پس از ساخت مرکز بهداشتی میباشد که شامل سه نوع تغییر می باشد

I. تغییر پارامتر های اتاق : مانند تجهیزات ، تخت ها ، سقف بیماران بستری شده

II. تغییر پارامترهای قرارداد: این ویژگی در حالتی فعال است که بخشی از مرکز بهداشتی برون سپاری شده باشد همانند : آشپزخانه ، عینک سازی ، داروخانه

III. تغییر پارامترهای پرسنل : مانند تعداد پرسنل هر بخش ، نسبت تعداد کادر خانم به آقا

- **واسط سخت افزاری:**

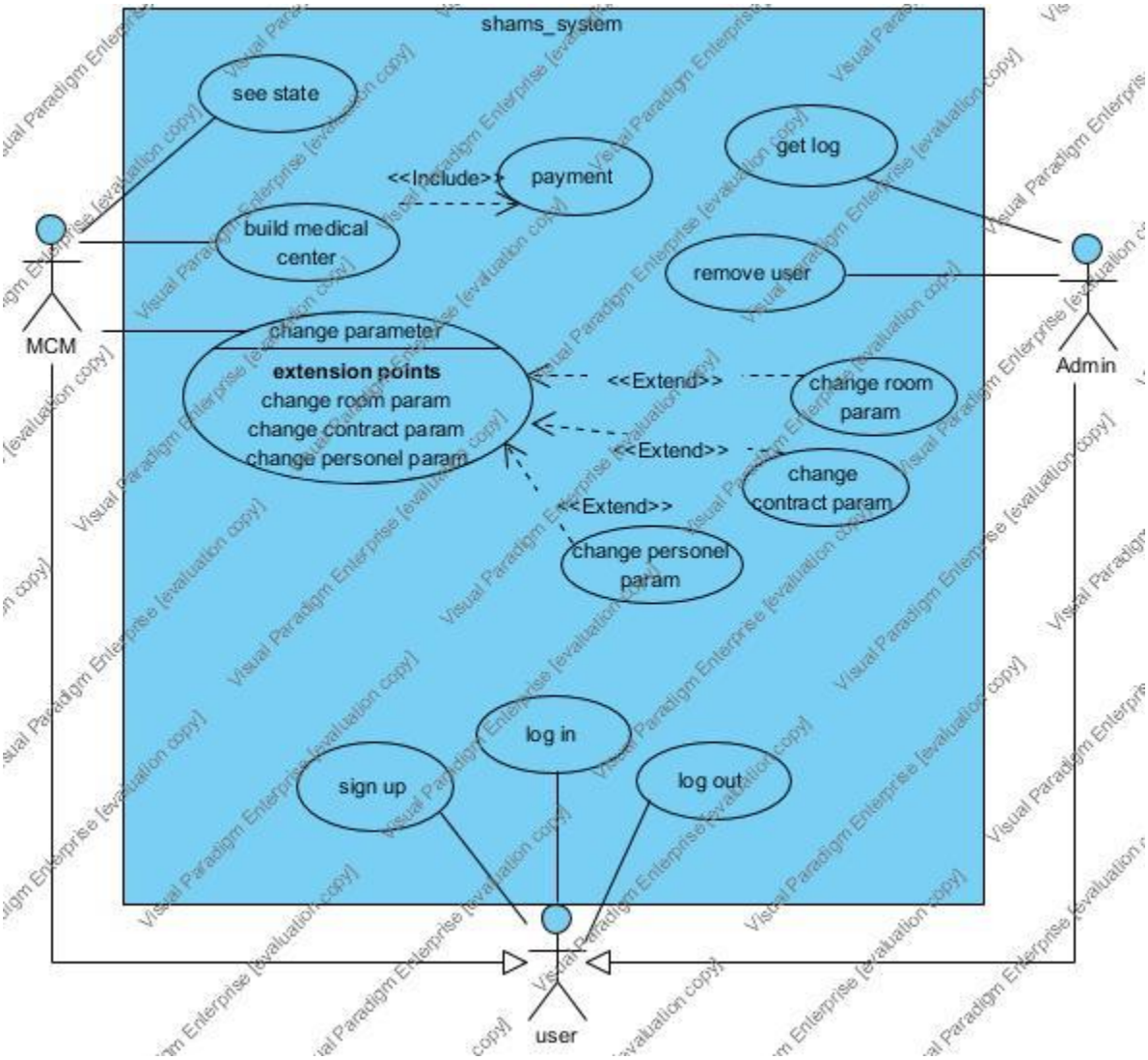
برای پیاده سازی نرم افزاری با مشخصات بالا به ۴ سیسم سخت افزاری با حداقل ۱۶ گیگ رم و با cpu هفت هسته ای نیاز می باشد که این ویژگی های سخت افزاری در اکثر سیستم های امروزی موجود میباشد .

- **ابزارها** :

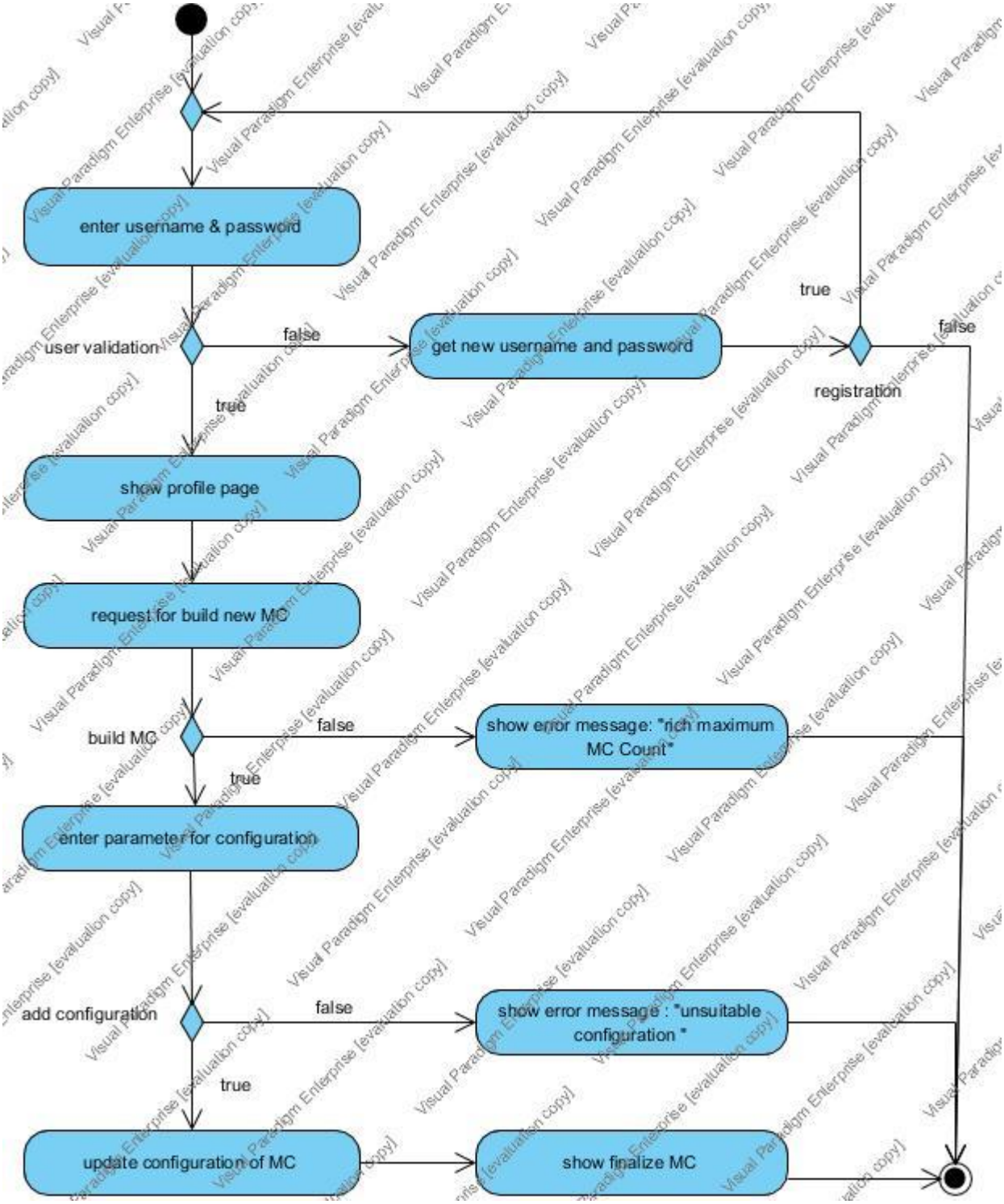
- SQL Server 2016
- Microsoft Visual Studio 2018
- Microsoft Project 2018
- Rational Requisite 2010

مدل های طراحی

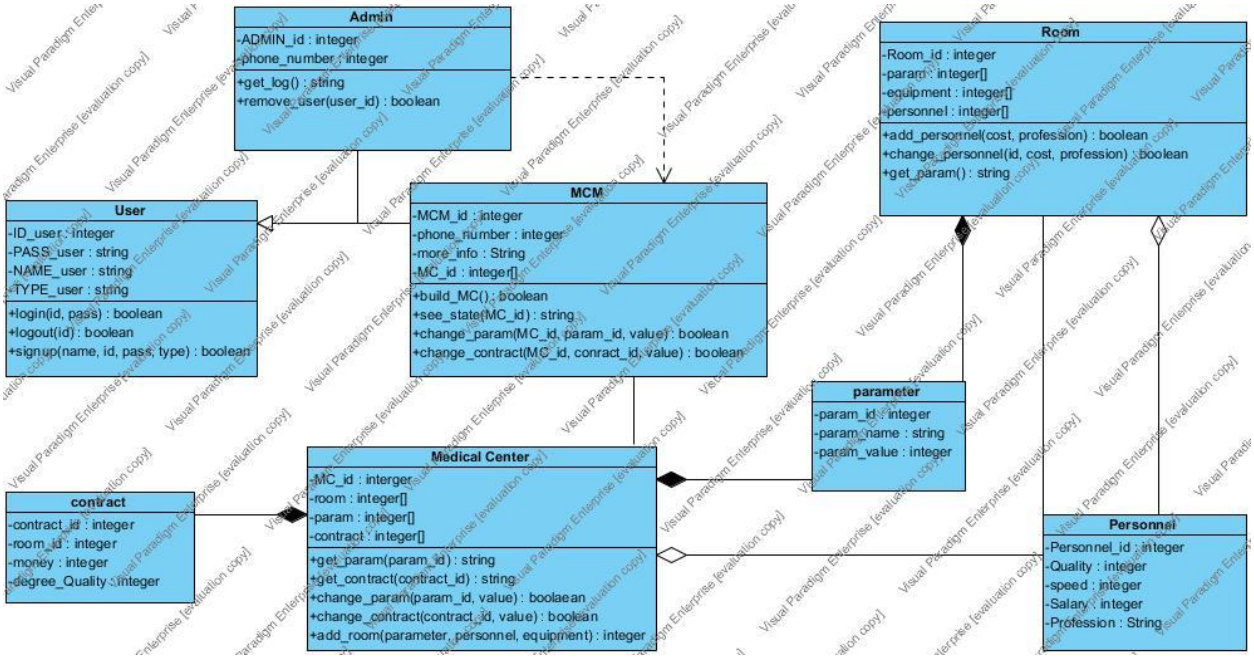
Usecase Diagram:



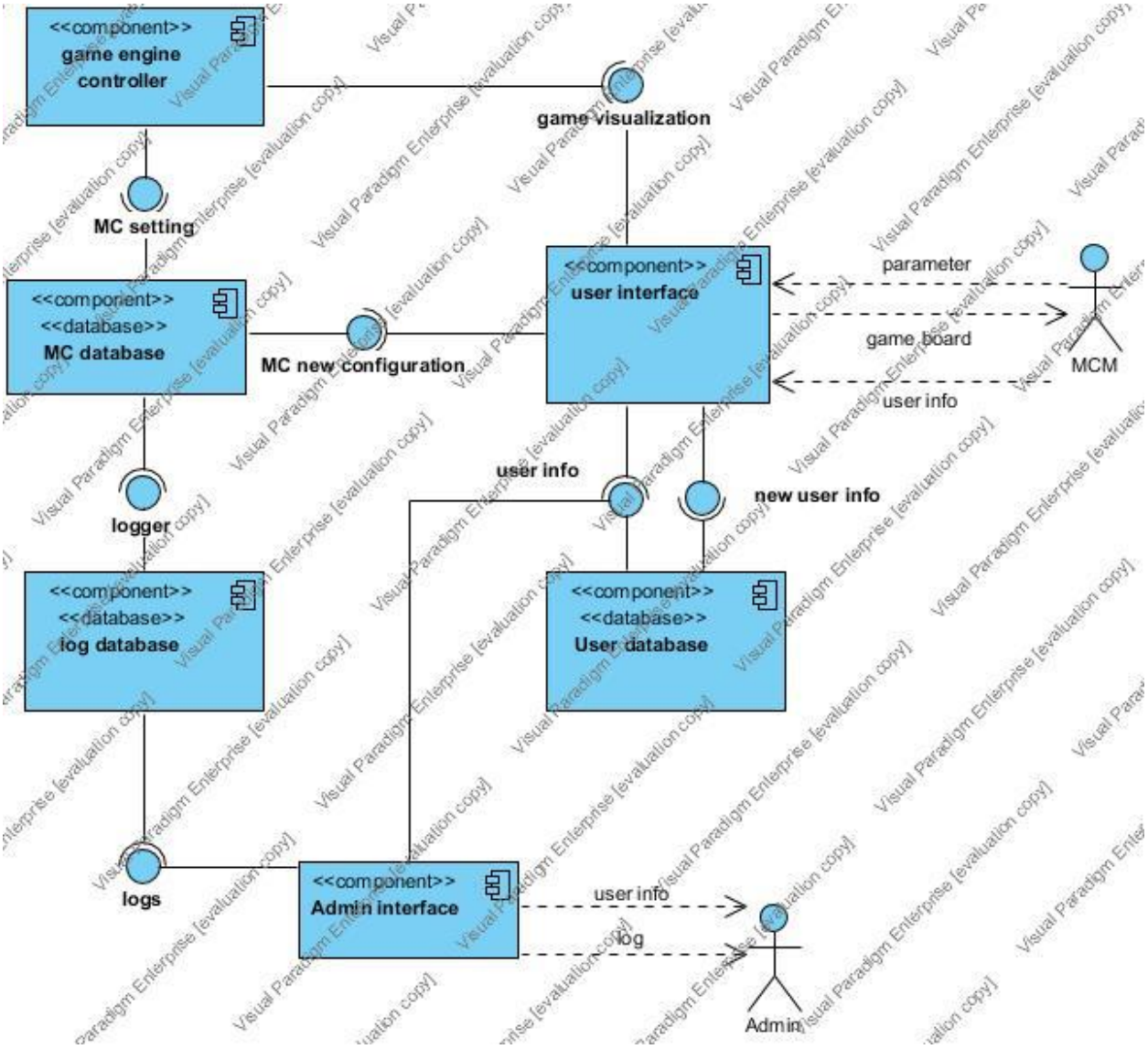
Activity Diagram:



Class Diagram :

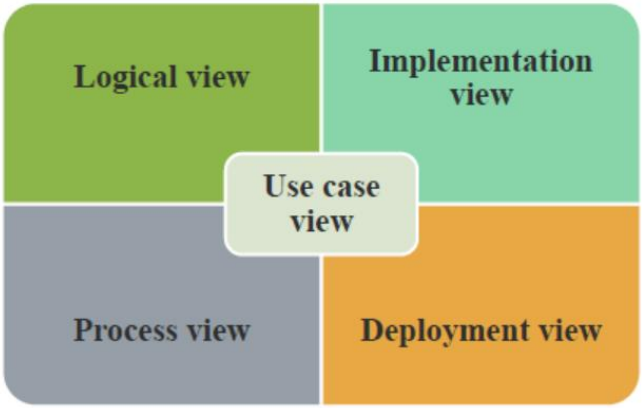


Component Diagram :



علل انتخاب مدل های طراحی

چون رویکرد شی گرایی و محصول RUP انتخاب شد .پس در روند انجام این پروژه از ابزار معرفی شده توسط همین متدولوژی یعنی UML استفاده میکنیم. در نمودارهای مربوط به UML به طور کلی ۵ view مختلف وجود دارد . در این پروژه تلاش شد تا از هر view یک نمودار اصلی کشیده شود.۵ view کلی به صورت موجود در شکل زیر است :



نمودار های انتخابی از هر کدام از view ها به شرح زیر است :

- نمودار usecase از usecase view : این نمودار یکی از اصلی ترین و ابتدایی ترین نمودار ها برای هر پروژه است که در فاز های ابتدایی (آنالیز و طراحی) و در حین ارتباط با مشتری ، مورد نیاز است. در این نمودار به طور کلی رفتار سیستم را از دید موجودیت های خارجی سیستم بررسی میکنیم و یک نتیجه observable برای موجودیت های خارجی سیستم فراهم می آوریم.
- نمودار activity از process view: این نمودار از نمودار های رفتاری سیستم است و بیشتر بر انتقال جریان کنترل و توالی عملیات ها بین object های مختلف تمرکز دارد. این نمودار ، بیشتر جنبه های دینامیک سیستم مورد نظر ما را مدل میکند و جریان کاری رخ داده را به طور دقیق در سطح کاربرد ساخت یک بیمارستان نمایش میدهد.
- نمودار class از logical view: این نمودار بر خلاف activity diagram که بر جبه های دینامیک سیستم تمرکز دارد، بر ساختار و جنبه های استاتیک سیستم توجه دارد . در این نمودار اجزای مورد

نیاز ، صفت ها و عملیات های هر object ونوع روابط بین object ها نمایش داده میشود . این نمودار به عنوان یک از ورودی های اصلی مورد نیاز برای فاز construction مورد نیاز است.

- نمودار component از implementation view : این نمودار شامل کامپوننت های سیستم است و برای رسیدن به معماری دقیق سیتسم و داشتن تمامی جزییات هر یک از کامپوننت ها و تعاملات آن ها در مرحله inceptionو elaboration برای طراحی معماری سیستم بکار می آید. در این نمودار ساختار منطقی موجودیت ها به همراه جزییات دقیق و روابط میان کامپوننت ها و artifact مورد نیاز برای پیاده سازی هر کدام به طور کامل مورد بررسی قرار میگیرد
- نمودار deployment از deployment view : این نمودار بازنمایی فیزیکی از سیستم یعنی دقیقا اجزای فیزیکی مثل فایل های اجرایی و سخت افزار مورد نیاز را بر عهده دارد .

به کمک این نمودار ها میتوان یک بازنمایی کلی از تمام ابعاد و نیازمندی های این سیستم را در تمامی مراحل انجام پروژه داشت

جایگاه و نقش QC و QA

وظایف این دو جایگاه به طور کلی به این صورت است که QA وظیفه تولید استاندارد ها و قالب ها و به طور کلی زیرساخت های مورد نیاز برای داشتن یک نرم افزار با کیفیت را ایجاد میکند و QC وظیفه اندازه گیری معیار ها و پارامتر ها تولید شده توسط QA ، برای رسیدن به محصول با کیفیت یاد شده را دارد. وظایف دقیق این دو گرو ه در جدول زیر درج شده است:

Quality Control	Quality Assurance
• تعیین برنامه،استراتژی و ابزار و استاندارد و متد های مورد نیاز برای test و review • مدیریت امنیت • مدیریت ریسک پروژه • مدیریت تغییرات مورد نیاز در پروژه • جمع آوری و بررسی error ها و defectها در چرخه حیات پروژه • بررسی روند انجام تست و استفاده از ابزارو استاندارد ها و عملکرد توسط تیم کنرل کیفیت • تولید SQA plan • آموزش تیم QC و همین طور stackholder های پروژه	• بررسی صحت و سازگاری و کامل بودن مدل ها و کدهای تولیدی در روند انجام پروژه • انجام دقیق تست و review ها کاملا منطبق بر استانداردها و با متد ها و ابزار و استراتژی های تیم QA • اندازه گیری و Report کامل دقیق تمامی متریک ها و خطا های موجود در پروژه

فریم مهندسی کیفیت ، metrics و mesurment

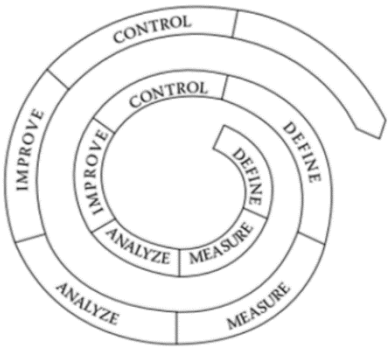
با توجه به سطح کوچک پروژه و همینطور دانش آموخته در روند درس و همین طور مدل نیازمندی های پروژه ، مدل FRUPS با این پروژه تطابق کامل را دارد . در این مدل با پنج نوع نیازمندی روبرو خواهیم بود که برای هر بخش یک indicator و یک metric معرفی خواهیم کرد.

Type Req.	mesurment	metrics	Indicator
Functionality	- میزان تطابق نرم افزار با خواسته مشتری	خروجی acceptance test	> ۹۰٪
	- دقت و درستی و صحت عملکرد نرم افزار	خروجی integrity test	>۹۵٪
	- امنیت سیستم	**نسبت تعداد ورود اشتباه به حساب کاربری به کل ورودها**	<۰/۰۵
Usability	- حفظ یکپارچگی سایت - وجود راهنمای مفید برای کار با سیستم برای افراد جدید - ظاهر مناسب و قابل فهم برای کاربران	تعداد layout راهنما هر صفحه	<5 >۲۰ خط
		نسبت تعداد عکس به کلمه در صفحه	>۰/۵
Reliability	-تعداد خطای اجرایی سیستم -مقدار زمان ریکاوری و یا پاسخ به خطا در سیستم - تولید خروجی های مناسب در زمان ایجاد خطا	**تعداد خطا در هر ۱۰۰ اجرا**	<۱۰
		زمان ریکاوری(s)	<۰/۲
		تعداد کلمه در پیغام خطا	> ۱۰ کلمه

Performance	- زمان پاسخ دهی سیستم - امکان استفاده چندین کاربر به صورت همزمان از سیستم -سرعت پردازش سیستم	Response time تعداد کاربر تعداد پردازش در ثانیه	۰/۱ ثانیه > ۱۰۰ کاربر> ۲۰۰ ثانیه>
Supportability	- قابلیت نگهداری آسان سیستم - قابلیت گسترش سیستم و اعمال تغییرات در عملیات های سیستم	Maintain هر۱۰ خط Debug هر ۱۰ خط	۵ دقیقه> ۱۰ دقیقه >

رویکرد تضمین کیفیت

رویکرد ما برای تضمین کیفیت منطبق شدن با روش ها و استاندارد های موجود در این زمینه است . دو استاندارد CMMI و six sigma گزینه های مناسبی با توجه به نیازمندی های این پروژه هستند. و چون تیم تولید با روش six sigma آشنایی نسبی دارد و این روش در بلند مدت سبب جلوگیری از defect و افزایش کیفیت و سود پروژه میشود، برای تضمین کیفیت انتخاب میشود.در واقع این روش با هدف کم کرردن defect های پروژه، کاهش ناپایداری-variation در پروژه ، توجه بر مشتری ، انجام فعالیت ها بر اساس وضعیت حقیقی پروژه و افزایش کارگروهی طراحی شده که همگی منطبق بر اهداف ما در این پروژه نیز می باشد.



روش six sigma یک روش برای اندازه گیری کیفیت است و یک معیار کلی به نام سیگما دارد. و با توجه به سطح سیگما پروژه میزان کیفیت محصول را بر حسب تعداد defect های تولید شده در یک میلیون بار تضمین میکند.به عنوان نمونه اگر سطح سیگما محصولی برابر با ۴ باشد ، به این معنیست که احتمال وجود defect در این محصول ۰,۶۲۱٪است.

در این روش به صورت iterative پنج مرحله زیر را انجام میدهیم

۱. Define : این مرحله به طور کلی هدف روشن شدن صورت پروژه است در این راستا فعالیت هایی با هدف مشخص شدن اهداف پروژه، نیازمندی ها ،مشتری ها و وضعیت فعلی سیستم و همین طور سامان دهی و اولویت بندی نیازمندی ها در پروژه انجام میشود .مراحل در قسمت بعد مفصلا شرح داده میشود
۲. Measure :در این مرحله تمرکز بر وضعیت فعلی سیستم است و کیفیت عملکرد فرآیندها و خروجی های فعلی مورد اجرا و محصولات آن را بر اساس متریک های مشخص جمع آوری میکنیم .
۳. Analyze : این مرحله هدف شناسایی مشکل و پیشنهاد راه حل است. یعنی در این مرحله اطلاعات اندازه گیری شده در مرحله قبل را آنالیز میکنیم تا مشکل را بیابیم و سپس با انجام brain storming راه حل های موجود را می یابیم و سپس راه حل مناسب انتخاب میشود.
۴. Improve : در این مرحله برای انجام راه حل یافته شده در مرحله قبل را برنامه ریزی کرده و سپس اقدامات لازم را انجام میدهیم.
۵. Control : در این گام تلاش میکنیم که متریک های موثر موجود را بیابیم و برای باقی نگه داشتن کیفیت در همین وضعیت برنامه ریزی و اقدامات لازم را انجام دهیم .

SQA plan

این برنامه یک roadmap و یک نقشه کلی از تمامی فعالیت های quality assurance را مشخص میکند. برای تولید SQA plan نیز به سراغ استاندارد های موجود میرویم . این نقشه در واقع در یک framework واحد برای تمامی تیم quality assurance ایجاد میکند و میتوان با استفاده از آن به فعالیت های انجام شده در طول پروژه اطمینان کرد . این نقشه شامل بخش های مختلفی است. در ادامه SQA Plan بر طبق استاندارد IEEE برای این پروژه آمده است

۱. Introduction

این سند یک پایه کلی برای تمامی فعالیت های مربوط به بخش تضمین کیفیت پروژه شبیه ساز مدیریت سلامت است .هدف از تولید این سند، مشخص سازی دقیق تمامی فرایند ها ، رویه ها و و فعالیت ها استانداردها و ابزار های مورد نیاز در پروژه برای رسیدن به کیفیت مطلوب محصولات ذکر شده در طرح پروژه است. محدوده این سند شامل موارد زیر خواهد بود:

- معرفی کامل تیم QA و وظایف هر یک از اعضادر روند پروژه
- تعریف کامل فعالیت ها و خروجی حاصل از هر مرحله کار تیم QA
- بررسی خروجی نهایی حاصل از فعالیت این تیم

۲. Refrenced document

این سند بر پایه استاندارد IEEE نسخه IEEE Std. 730-2002 طراحی شده است.

۳. Management organization

نام نقش ها ممکن است با اسامی متفاوتی به کار رفته باشند ، اما role های موجود و وظایف هر یک به طور کلی به شکل جدولی خواهد بود که در صفحه بعد آمده است.

نقش	وظیفه
مدیر واحد QA	• مدیریت عملیات های مربوط به QA را در تمامی مراخل دارد
صاحب سیستم	• نمایش نیازمندی ها و exception های موجود در پروژه • همراهی در acceptance test است
مشاور QA (Reviewer)	• بررسی کیفیت deliveable ها • Review و بررسی plan ها برای تطابق با استاندارد ها و همین طور بررسی اجرایی بودن برنامه ها • Review و بررسی artifact های موجود و شناسایی defect ها • ارایه feedbackبه مدیر QA
مدیر تیم تولید	• تضمین پیاده سازی فعالیت های تضمین کیفیت در هر بخش از پروژه • هماهنگی و هم گامی تیم تولید با QA • مشخص کردن سختی های پیاده سازی در بخش های مختلف
عضو تیم QA	• ساپورت و حمایت تیم با انجام task های مختلف محول شده از سمت reviewer در تیم

۴. Documentation

در پایان هر کدام از فاز های پروژه مستندات زیر طبق قالب های تعریف شده توسط یکی از استاندارد های IEEE و یا MCM تحویل داده می شود.

فاز اجرایی	مستندات مورد نیاز در پایان فاز
فاز **inception**	• Project Overview • نقشه پروژه • نمودار Gantt Chartز • مستند تحلیل هزینه/Cost Analysis • مستند SRS • نیازمندی ها (به طور ناقص)
فاز **elaboration**	• برنامه تست QA (SQAP) • مدل object • Formal Specifications • تست پلن • چک لیست formal review
فاز **construction**	• نقشه implementation • نقشه معماری اجزای مختلف سیستم • نحوه ارتباط اجزا مختلف
فاز **transition**	• راهنمای استفاده کاربر/ User Manual • نتیجه تست های مختلف انجام شده • ارزیابی کلی از عملکرد-هزینه-بودجه پروژه
فاز **production**	• Defect های یافته شده • راه حل و برنامه و تخمین بودجه حل defect ها • گزارش های اجرا در محیط واقعی(نه ازمایشی)

۵. Standard & Metrics

- استاندارد ها برای بخش ها مختلف :
  - استاندارد document نویسی : ۱-IEEE71 documentation ۲-ISO 9000
  - استاندارد کد نویسی : ASP.Net 2.02
  - استاندارد document کدها : ASP.Net documentation
  - استاندارد تست ها : استاندارد IEEE برای test documentation
- Metrics : برای اندازه گیری نرم افزار از معیار LOC استفاده میکنیم.

۶. Review & Audit

در پایان هر فاز انجام پروژه ، تیم تولید باید تمامی مستندات آن فاز را در قالب یک ارایه ی Formal در جلسه ای با بخش QA ارایه دهند . و سپس این ارایه ها و مستندات توسط تیم QA با استاندارد ها و نیازمندی های های ذکر شده تضمین کیفیت هر بخش تطابق یابد. این مستندات در صورت بازبینی و عدم وجود کیفیت لازم، نیاز به بررسی مجدد توسط تیم تولید و رفع نقایص موجود و ارایه دوباره به تیم QA دارد.

۷. Test & Tools

یک STP (software testing plan) برای رفع تمامی نیازمندی های پروژه تولید میشود. در این نقشه یک overview از تمامی فعالیت ها، ابزار و زمانبندی و منابع مورد نیاز برای انجام unit test و integrity test شرح داده شده است .هم چنین در پایان هر فاز review نیز انجام میشود تا از عدم وجود defect و تطابق با roadmap پروژه و استاندارد ها مطمین شویم.

از ابزار virtual box به عنوان سرور مجازی در روند این پروژه استفاده میشود. برای اندازه گیری سایز پروژه و آنالیز های لازم نیز ابزار COCOMO II استفاده میشود. هم چنین رویکرد ما در این پروژه ، رویکرد شی گرایی است و از ابزار Visual Paradigm برای رسم نمودار های UML استفاده میشود. همچنین برای رمزنگاری ارتباط بین سرور و کلاینت میتوان از ابزارCryptix استفاده کرد

۸. Training

دانش مورد نیاز برای تیم تولید شامل :

- آشنایی کامل با ابزار Visual Paradigm
- آشنایی با ابزار Crptix
- آشنایی با Virtual Box

علاوه بر این، تیم تضمین و کنترل کیفیت باید دانش زیر را نیز داشته باشند :

- آشنایی کامل با ابزار COCOMO II
- آشنایی با ابزار audit
- آشنایی کامل با فریم ورک Sixsigma

تکنیک کنترل کیفیت

همان طور که در بخش قبلی توضیح دادیم ، در این پروژه از روش Six sigma برای تضمین کیفیت بهره میجوییم. این روش، یک روش آماری است و تاکید بسیاری بر اندازه گیری متریک ها و به طور کلی کمی بودن تمامی پارامتر ها دارد.

نحوه انجام این تکنیک به این صورت است که در فاز اول ابتدا نیازمندی خواسته شده مشتری و سیستم فعلی و فرایند هایش را به طور کامل شناسایی میکنیم و چارتر پروژه و زمانبندی پروژه را رسم میکنیم .

در فاز دوم متریک های اندازه گیری کیفیت را مشخص میکنیم و برای فرایندهای فعلی سیستم اندازه گیری میکنیم و همینطور سیگما سیستم فعلی را نیز محاسبه میکنیم. و ظرفیت سیستم/capability را نیز میابیم تا variationموجود را در مقایسه با محدودیت های اعمال شده از سمت کاربر مقایسه کنیم.

در فاز بعدی علت وجود این variation را با انالیز root-cause به کمک نمودار های تیغ ماهی انجام میدهیم و علت بروز اشکالات را میابیم. سپس یک طوفان فکر ایجاد میکنیم و تمام راه حل های ممکن برای برطرف کردن این علت را بدست میآوریم. حال برای هر کدام از این راه حل ها بر اساس اصل ROI و سه معیار کلی اثرگذاری- زمان-هزینه و بررسی ریسک این راه حل ارزش تعیین میکنیم .

در فاز چهارم راه حل انتخاب شده را قطعی میکنیم و برای پیاده سازی این راه حل plan لازم را ایجاد میکنیم و سپس فعالیت ها و تغییرات را طبق پلن اعمال میکنیم

در فاز آخر متریک های تاثیرگذار را میابیم و یک برنامه کنترلی برای کیفیت بدست آمده تعریف مکنیم و برنامه کنترلی را بر فعالیت ها اعمال میکنیم.همین طور میزان پیشرفت سیستم را اندازه میگیریم و گزارش میدهیم.

متریک های technical review

در این بخش از متریک های معرفی شده در کتاب پرسمن استفاده میکنیم:

متریک	تعریف متریک	واحد	مقدار
Ep	میزان فعالیت لازم برای آمادگی پیش از جلسه review	نفر-ساعت	۵
Ea	میزان فعالیت لازم برای انجام جلسه review	نفر-ساعت	۲۰
Er	میزان فعالیت لازم برای تصحیح خطاهای پوشش داده نشده در جلسه review	نفر-ساعت	۳۵
WPS	حجم محصولی(که به طور میانگین) قرار است برایش review صورت گیرد (طبق استاندارد در SQAP)	LOC	۱۰۰۰
Err_{minor}	تعداد خطاهای یافته شده در هر review که برای تصحیح به طور میانگین ۵ نفر-ساعت انرژی میبرند(بین ۳ تا ۷ نفر-ساعت)	تعداد	۱۵
Err_{major}	تعداد خطاهای یافته شده در هر review که برای تصحیح به طور میانگین ۱۰ نفر-ساعت انرژی میبرند. (بین ۸ تا ۱۲ نفر-ساعت)	تعداد	۵

حال با استفاده از متریک های محاسبه شده در بالا، به محاسبه متریک های اساسی دراین زمینه میپردازیم:

$$E_{total} = E_p + E_a + E_r = 5 + 20 + 35 = 60$$

$$Error_{total} = E_{major} + E_{minor} = 15 + 5 = 20$$

$$Error\ density = \frac{Error_{total}}{WPS} = \frac{20}{1000} = 0.02$$

برای بررسی صرفه اقتصادی انجام بازبینی در پروژه یک بار میزان انجام کار با review و یک بار میزان انجام کار بدون بازبینی را اندازه میگریم و اگر حجم کار و هزینه در حالت با بازبینی کمتر بود، بدین معنیست که بازبینی صرفه دارد.برای انجام این محاسبات فرض میکنیم در review تیم ما ۷۵٪ خطا ها را مییابد. محاسبات انجام شده برای یک مرحله بازبینی در حالت میانگین به صورت زیر خواهد بود :

کار مورد نیاز برای بازبینی + کار مورد نیاز برای تصحیح خطاها = کل کار مورد نیاز

در حالت داشتن review :

$$\text{کل کار} = E_{total} + \text{هزینه خطاهای یافت نشده} = E_{total} + \left(E_{major} * 15 + E_{minor} * 5\right) =$$

$$60 + (2 * 10 + 5 * 5) = 105 \text{ نفر} - \text{ساعت}$$

در حالت بدون review:

$$\text{کل کار} = E_{total} + \text{هزینه خطاهای یافت نشده} = 0 + \left(E_{major} * 15 + E_{minor} * 5\right) =$$

$$0 + (7 * 10 + 20 * 5) = 170 \text{ نفر} - \text{ساعت}$$

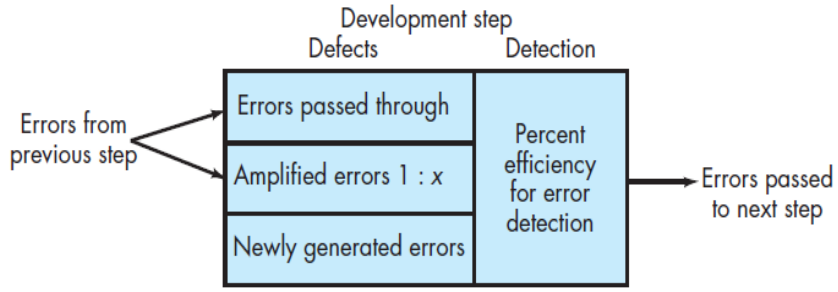
* اعداد ۷ و ۲۰ با فرض یافتن ۷۵٪ از خطاها در توسط بازبینی محاسبه شده اند.

همان طور که دیده میشود با انجام بازبینی حجم کار مورد نیاز در هر گام تا حد بسیار خوبی کم شده است یعنی برحسب محاسبات بالا با انجام بازبینی در هر گام به طور متوسط ۶۵ نفر-ساعت کار کمتری برای نصحیح خطاها نیاز است تا انجام بدهیم . پس بازبینی کاملا برای ما به صرفه است.

مدل defect amplification

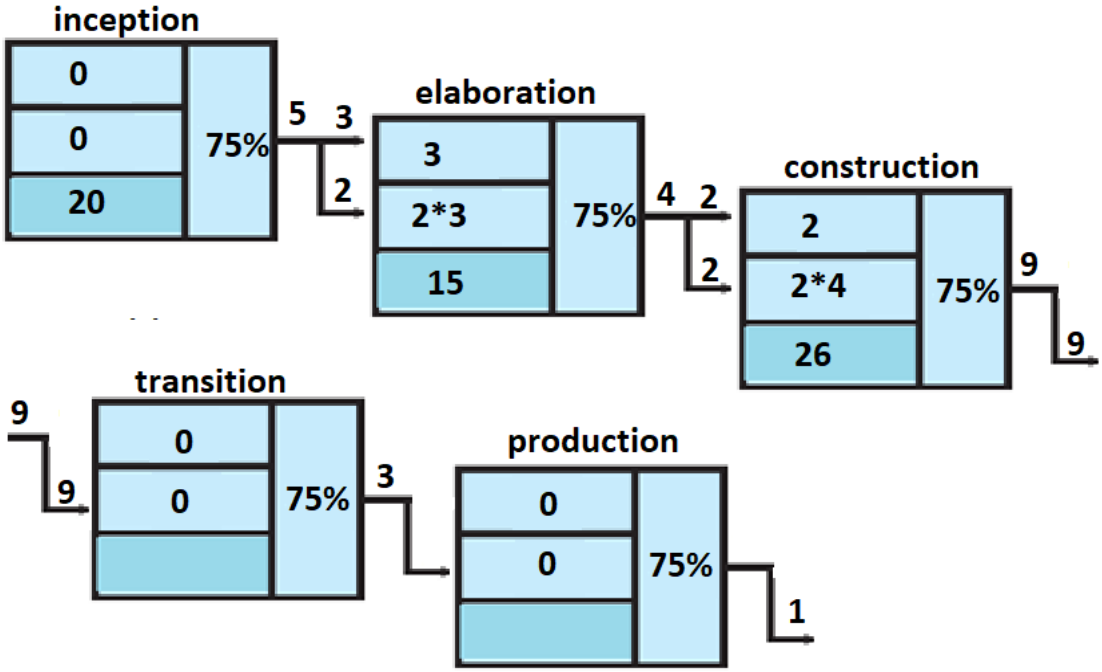
به طور کلی در هر پروژه هر چه در مراحل زودتری خطاها را بیابیم، هزینه کمتری برای تصحیحشان باید بکنیم، این جمله یعنی این که هر چه خطاها به فاز پایانی پروژه نزدیک میشوند، هزینه لازم برای برطرف کردنشان بیشتر خواهد شد. همین طور هر چه این خطاها مربوط به فاز های اولیه پروژه باشند هزینه rework برای تصحیحشان بیشتر خواهد شد. به عنوان مثال اگر یک خطا در فاز inception ایجاد شده باشد و یک خطا در فاز construction و ما هر دو این خطا ها در مرحله transition متوجه شویم. قطعا هزینه برطرف سازی خطای اول چندین برابر خطای دوم خواهد بود ؛ زیرا این نیازمندی که اشتباه تشخیص داده شده منجر به طراحی چندین بخش اشتباه و چندین پیاده سازی اشتباه در پروژه شده است.

بر اساس توضیحات داده شده، چندین مدل برای بزرگنمایی خطاها طراحی شده است. هدف همگی این مدل ها انتقال همین مفهوم انتشار است که در بالا توضیح داده شد .یعنی به کمک این مدل ها میزان تاثیر خطا و شدت انتشار و بزرگنمایی را در هر مرحله از پروژه به نمایش میگذاریم.در این بخش از مدل معرفی شده در کتاب پرسمن استفاده خواهیم کرد :



در این مدل برای هر فاز از مسیله از یک نمودار همچون بالا استفاده میشود . در این نمودار برای هر فاز ، تعدادی خطا از مرحله قبل وارد شده که بعضی از آن ها که اساسی تر هستند (اما خطاهای major در بخش قبلی) با یک نرخی بزرگنمایی میشوند. و تعدادی خطا نیز بدون بزرگنمایی منتقل میشوند(همان خطاهای minor) . همچنین در هر مرحله ممکن است تعدادی خطا نیز ایجاد میشوند. اما درصدی این خطاها در همین با انجام بازبینی ها و فرایند های لازم شناسایی میشوند و به مرحله بعد انتقال نخواهند یافت. و به همین ترتیب برای کل پروژه نمودار های زنجیره ای خواهیم داشت

برای این پروژه نمودار بزرگنمایی خطاها شکلی به صورت صفحه بعد خواهد بود :



در این نمودار به طور کلی (طبق فرضی که در قسمت قبلی سوال داشتیم) فرض میکنیم که با انجام فعالیت های مربوط به بازبینی نرخ کشف خطاها به طور متوسط در تمامی مراحل ۷۵٪ باشد. با اعمال این فرض، توضیح نمودار به این صورت خواهد بود که : درابتدا چون مشتری ما یک سری پزشک هستند و با زبان مهندسی آشنایی کافی ندارند ، مقدار زیادی خطا در تشخیص نیازمندی و مدل کردنشان رخ خواهد داد. اما با انجا بازبینی تعداد بسیار خوبی از این خطاها کشف خواهد شد و به مرحله بعد راه نخواهند یافت. در مرحله بعد نیز تعدادی خطا از این فاز تولید میشود و تعدادی خطا از مرحله قبل منتقل میشود. در فاز construction چون در حال

در کد نویسی هستیم تعداد خطا بیشتر میشود. در فاز بعدی چون تنها به رفع خطاها و انجام تست ها میپردازیم خطایی ایجاد نمیشود و تنها ۳ خطا بدست مشتری میرسد. در این فاز با بررسی مشتری و انجام عملیات های مربوطه ۲ خطا را کشف و تصحیح میکنیم و تنها ۱ خطا به دور بعدی و به عنوان defect تشخیص داده میشود. (گرچه خطاهایی که در فاز production وارد میشود نیز بسیار دیر تشخیص داده شده و به دست مشتری رسیده است و میتوان آن ها را نیز defect تشخیص داد)

چرخه حیات،استراتژی و واحد تست

همان طور که میدانیم متدولوژی کیفیت، فرایند ها، روش ها، تکنیک ها و ابزار های مورد نیاز انجام یک پروژه را در تمامی فازها پوشش میدهد. در این پروژه همان طور که در فاز های قبلی توضیح داده شد رویکرد ما شی گرایی است و از محصول RUP این رویکرد بهره میبریم. با توجه به این متدولوژی باید از ۴ تست معرفی شده در این متدولوژی بهره جست .

در جدول زیر، انواع تست ها در این سیستم، واحد تست هر کدام و استراتژی ما در انجام هر یک و ورودی و خروجی هر تست به تفکیک مشخص شده است :

نام تست	هدف تست	واحد تست	استراتژی تست	ورودی تست	خروجی تست
واحد	صحت عملکرد تک تک ماژولها	کلاس	رندوم	تست پلن + کلاس های برنامه (تعریف و یا کد)	گزارش تست + خطاهای منطقی تعریف کلاس ها یا اشکالات پیاده سازی
یکپارچگی	صحت عملکرد کامپوننت های سیستم در ارتباط با هم	Build	سلسله مراتبی	تست پلن + کلاس های برنامه (تعریف و یا کد)	گزارش تست + خطاهای منطقی در ارتباط بین کلاس ها یا خطا در عدم هم خوانی interface و یا ارورهای عملیاتی
صحت	تطابق سیستم پیاده سازی شده با نیازمندی ها مشتری	Use case	جعبه سیاه	تست پلن + مدل نیازمندی + Use case	گزارش تست+ نیازمندی های پوشش داده نشده + عملیات های بدون نیازمندی
سیستم	ارتباط مناسب سیستم با شرایط و عوامل بیرونی	سیستم	Top down	تست پلن + شرایط خاص ورودی + برنامه های بیرونی در ارتباط	گزارش تست + گزارش نحوه عملکرد سیستم در شرایط مختلف

چرخه حیات تست شامل ۶ مرحله کلی هست که به شکلی به صورت زیر نمایش داده میشود :

setup محیط تست → تولید test case → planning/ برنامه ریزی تست→ آنالیز نیازمندی ها → اجرای تست → حلقه تست closure

فرایند، خروجی و ورودی هر کدام از این مراحل در جدول زیر مشخص است :

مرحله	*فعالیت ها*	*ورودی*	*خروجی*
آنالیز نیازمندی ها	انتخاب نوع تست آماده سازی RTM شناسایی محیط تست امکان سنجی	مدل نیازمندی ها	*RTM* *گزارش امکان سنجی*
برنامه ریزی تست	آماده سازی تست پلن انتخاب ابزار تست تخمین effort تست برنامه ریزی تخصیص منابع	RTM گزارش امکان سنجی	*تست پلن* *گزارش تخصیص منابع و نیروی انسانی*
تولیدtest case	ساخت تست کیس ساخت داده های مورد نیاز برای تست	تست پلن گزارش تخصیص منابع و نیروی انسانی	*تست کیس* *داده های تست*
setup محیط تست	آماده سازی نرم افزار و سخت افزار مورد نیاز محیط تست جایگذاری داده های تست و برنامه تست	تست کیس داده های تست	*محیط آماده تست*
اجرای تست	انجام تست مستندسازی نتیجه تست	محیط آماده تست	*گزارش تست (defect)* *RTM کامل شده* *تست کیس کامل شده*
closureحلقه تست	ارزیابی تست انجام شده (تست پلن، تست کیس ،تست دیتا و نتایج حاصل	گزارش تست (defect) RTM کامل شده	*متریک های تست* *گزارش این مرحله*

W5H2 test plan

طبق خواسته سوال برای هر کدام از تست ها به هفت سوال اساسی در این رابطه پاسخ میدهیم.

Unit test

سوالات	پاسخ ها
What	یکی از انواع تست ها که بر صحت کلاس ها تمرکز دارد
Why	هدف از این تست بررسی صحیح و بدون خطا و منطبق بر نیازمندی بودن هر کلاس است
Who	تیم تست که شامل (test leader-test manager- test designer-tester)
Where	این تست بر روی تک تک کلاس ها در طول پروژه انجام میشود
When	در این پروژه در پایان هر فاز صحت تمامی کلاس ها تست میکنیم
How	با توجه به این که استراتژی ما در اینجا رندوم است. در هر گام به طور تصادفی تعدادی از کلاس ها انتخاب میکنیم و صحیح عمل کردن آن را با بررسی دقیق تعریف کلاس،ورودی ها و خروجی ها و attribute ها و توابع کلاس را چک میکنیم. با توجه به پیاده سازی این پروژه در معماری MVC و با Asp.net از ابزار XUnit.net استفاده میکنیم
How much	طبق اصل ROI تا زمانی به انجام هر یک از تست ها ادامه میدهیم که برایمان در هزینه ها و زمان صرف داشته باشد .

Integrity test

سوالات	پاسخ ها
What	یکی از انواع تست که بر روابط بین کلاس ها تمرکز دارد
Why	در مواردی پیش می آید که تک تک کلاس های موجود در برنامه به درستی نوشته شده اند. اما این کلاس ها به درستی با همکاری نکرده و نتیجه دلخواه را ایجاد نمیکنند. به عنوان مثال ممکن است که interface دو کلاس به یکدیگر نخورد و یا واحد یا scale خروجی های دو کلاس مثل هم نباشد. پس لازم است که کلاس ها در ارتباط با هم در قالب build ها تست شوند
Who	تیم تست که شامل (test leader-test manager- test designer-tester)
Where	این تست بر روی build های مختلف در پروژه انجام میشود
When	این تست در پایان هر فاز برای درستی سنجی ارتباطات انجام میشود

How	استراتژی تست ما در اینجا به صورت سلسله مراتبی است . یعنی به صورت پله پله صحت ارتباطات را بررسی میکنیم و به سراغ والد میرویم. ابزار مورد نیاز برای انجام این تست در این پروژه پلاگین postman است.
How much	طبق اصل ROI تا زمانی به انجام تست ادامه میدهیم که برایمان در هزینه ها و زمان صرف داشته باشد .

Validation test

سوالات	پاسخ ها
What	یکی از انواع تست ها با تمرکز بر نیازمندی های مشتری
Why	هدف اساسی از انجام این تست این است که سیستم نهایی تمامی نیازمندی های مشتری را پوشش دهد. یعنی ممکن است کلاس ها و ارتباطاتشان کاملا صحیح عمل کند اما این پروژه ، نیازمندی مشتری را به درستی پوشش ندهد و یا در پروژه بدون وجود نیازمندی مشتری قابلیت های اضافه قرار داده باشیم. تشخیص همه ی این شرایط بر عهده این تست است
Who	تیم تست که شامل (test leader-test manager- test designer-tester
Where	براساس مدل نیازمندی روی تک تک usecase ها در هر فاز تست میکنیم
When	در این پروژه در پایان هر فاز تست صحت را انجام میدهیم
How	استراتژی ما در این تست به صورت جعبه سیاه هست. و به این معناست که به ساختار داخلی کلاس ها کاری نداریم و تنها با توجه به usecase ها ، نیازمنددی های مختلف را trace میکنیم. ابزار کاربردی در اینجا ValGenesis است
How much	طبق اصل ROI تا زمانی به انجام تست ادامه میدهیم که برایمان در هزینه ها و زمان صرف داشته باشد .

System testing

سوالات	پاسخ ها
What	این تست یا هدف بررسی رفتار سیستم در شرایط مختلف انجام میشود
Why	هدف از این تست شناسایی رفتار سیستم و سطح عملیاتی سیستم است . این تست شامل چندین بخش است که در هر بخش رفتار سیستم را از یک جهت بررسی میکنیم .به عنوان مثال بیشتر تعداد کاربر که سیستم میتواند به صورت همزمان پاسخگو باشد، یا رفتار سیستم در مواجه با حمله های امنیتی (مثلا SQL injection) و یا رفتار سیستم در صورت ورودی نامعتبر توسط کاربر را بررسی میکنیم.
Who	تیم تست که شامل (test leader-test manager- test designer-tester
Where	این تست بر روی کل سیستم اجرا میشود
When	این تست در پایان هر فاز بر روی کل سیستم اجرا مشود
How	رویکرد ما در انجام این تست top down میباشد. یعنی از بالاترین سطح موجود یعنی کل سیستم به صورت یک جا تست را شروع میکنیم و سپس به سراغ زیر سییتم ها میرویم. ابزار ما در این جا برای انجام این تست TestWorks است که یک پکیج کامل از انواع و اقسام تست ها است .
How much	طبق اصل ROI تا زمانی به انجام تست ادامه میدهیم که برایمان در هزینه ها و زمان صرف داشته باشد .

scenario base testing

scenario based testing یکی از الگوهای تست نرم افزار در بخش validation تست ها است . یعنی این تست پس از انجام تست های unit و integrity انجام میشود و چک میکند که آیا نرم افزار کاربر را راضی میکند یا خیر. نکته کلیدی این تست این است که بر مبنای این عمل میکند که کاربر چه نیازمندی دارد و نه این که محصول چه ویژگی هایی دارد .در واقع این تست یک تکنیک است برای بررسی سیستم از نقطه نظر کاربر و شکست این تست به معنی این است که نیازمندی های مشتری به درستی شناخته و در پروژه پیاده سازی نشده است.

نحوه عملکرد ما در این تست به این صورت است که سناریو های مختلف سیستم را می گیریم و سناریو را trace می کنیم به این معنی که هر گام از این سناریو ها را در حالت های مختلفش بررسی میکنیم که آیا پاسخ مورد انتظار ذکر شده در سناریو را را دریافت کردیم یا نه. در ادامه یک سناریو از سیستم که میتواند به عنوان ورودی این تست قرار بگیرد بررسی میکنیم.

- سناریو اول : تعریف مرکز بهداشتی
  - کاربر وارد حساب کاربری خود میشود
    - اگر حساب کاربری نداشت :

-وارد صفحه ثبت نام شود

-نام کاربری و رمز عبور خود را انتخاب کند

-به صفحه حساب کاربری هدایت شود

- اگر حساب کاربری داشت و رمز عبور را به درستی وارد کرد :

-به صفحه کاربری هدایت شود

- اگر حساب کاربری داشت و رمز عبور را اشتباه وارد کرد

-به صفحه فراموشی رمز عبور هدایت شود

-رمز عبور جدید به ایمیل ثبت شده از کاربر در سامانه ارسال شود

-در صورت وارد کردن رمز اراسالی به طور صحیح ، وارد صفحه حساب کاربری شود

-در صورت اشتباه وارد کردن رمز با پیغام خطا "رمز عبور صحیح به ایمیل ثبت شده شما در سامانه ارسال شده است" برای ۲ ثانیه روبرو شود و به صفحه ورود به سایت هدایت شود

- بر روی دکمه "درخواست ساخت MC" از منو سمت راست کلیک بنماید
  - اگر تعداد MC های کاربر کمتر از ۵ است، با پیغام "با درخواست شما برای ساخت یک مرکز بهداشتی موافقت شد، لطفا تنظیمات این مرکز بهداشتی را مشخص نمایید "به مدت ۵ ثانیه روبرو شود و سپس به صفحه MC configuration هدایت شود
  - اگر تعداد MC ها بزرگتر و یا مساوی ۵ است، با پیغام خطا "تعداد مراکز بهداشتی شما به سقف مجاز رسیده، و این عملیات برای شما ممکن نیست" برای ۲ ثانیه روبرو شود و به صفحه حساب کاربری هدایت شود
- Configuration مرکز بهداشتی (شامل نوع مرکز بهداشتی - تعداد اتاق ها- تعداد پرسنل هر بخش- امکانات هر اتاق – تعداد تخت هر اتاق)را تنظیم نماید
- با زدن دکمه تایید پیغام "آیا از تنظیمات خود مطمین هستید؟" نمایش داده شود
  - در صورت تایید کاربر
    - سازگاری تنظیمات انجام شده باهم ، بررسی شود.
      - در صورت سازگاری اطلاعات در دیتابیس ذخیره شود. پیغام "تبریک! شما از هم اکنون مدیر یک مرکز بهداشتی هستید " برای ۵ ثانیه نمایش داده شود و به صفحه MC ساخته شده هدایت شود
      - در صورت عدم سازگاری پیغام خطای "تنظیمات در خواستی با هم سازگار نیستند ، لطفا مجددا سعی نمایید." برای ۴ ثانیه نشان داده شود و کاربر به صفحه MC configuration هدایت شود
  - در صورت لغو تایید ، به صفحه MC configuration هدایت شود.

مدل CRC

مدل CRC یکی از مدل های کاربردی در رویکرد شی گرایی است و در هنگام بررسی consistency مدل ها استفاده میشود. این مدل یک نمایش گرافیکی از class diagram است و تمرکز اصلی اش بر نمایش روابط بین کلاس هاست .نمونه ۵ CRC در این سیستم به شرح زیر است :

Class : MCM	
Responsibities	**Collaborators**
Build_MC	Medical Center
See_state	Medical Center
Change_param	Medical Center
	Parameter
	Room
	Personel
Change_contract	Medical Center
	Contract

Class : Room	
Responsibities	**Collaborators**
Add_personel	Medical Center
	Personel
	Parameter
Change_personel	Personel
Get_param	Parameter

Class : Admin	
Responsibities	**Collaborators**
Get_log	MCM
Remove_user	Medical Center
	MCM

Class :Medical Centar	
Responsibities	**Collaborators**
Get_param	Parameter
Get_contract	Contract
Change_param	Parameter
	MCM
Change_contract	Contract
	MCM
Add_room	Room
	MCM
	param

Class : User	
Responsibities	**Collaborators**
login	MCM
Signup	MCM
Signup	MCM

تست multiple class

ابتدا به بررسی تست multiple class testing میپردازیم در این تست طبق توضیحات موجود در کتاب عمل میکنیم . گام اول این تست شامل یافتن پیام های ارتباطی میان هر کلاس کلاینت و سرور است که به این منظور میتوانیم از عملیات های این کلاس ها استفاده کنیم. در گام دوم کلاس های مشارکت کننده در پیام های یافته شده در مرحله قبل را میبابیم . در گام سوم پیام های که برای هر عملیاتی که در پاسخ به پیغام ارسالی کاربر انجام شده، ایجاد شده است را می یابیم و درگام آخر عملیاتی که به دلیل پیام های یاد شده در بخش قبل انجام میشود را پیدا میکنیم. به این ترتیب به زنجیر عملیات ها در میان سرور و کلاینت دست می یابیم. حال طبق توضیحات ارایه شده در بالا ابتدا عملیات های مربوط به کلاینت را می یابیم. به این منظور همان usecase دنبال شده در بخش های قبل یعنی ساخت MC را دنبال میکنیم. عملیات های مورد نیاز برای این usecase از کلاس MCM به شرح زیر خواهد بود :

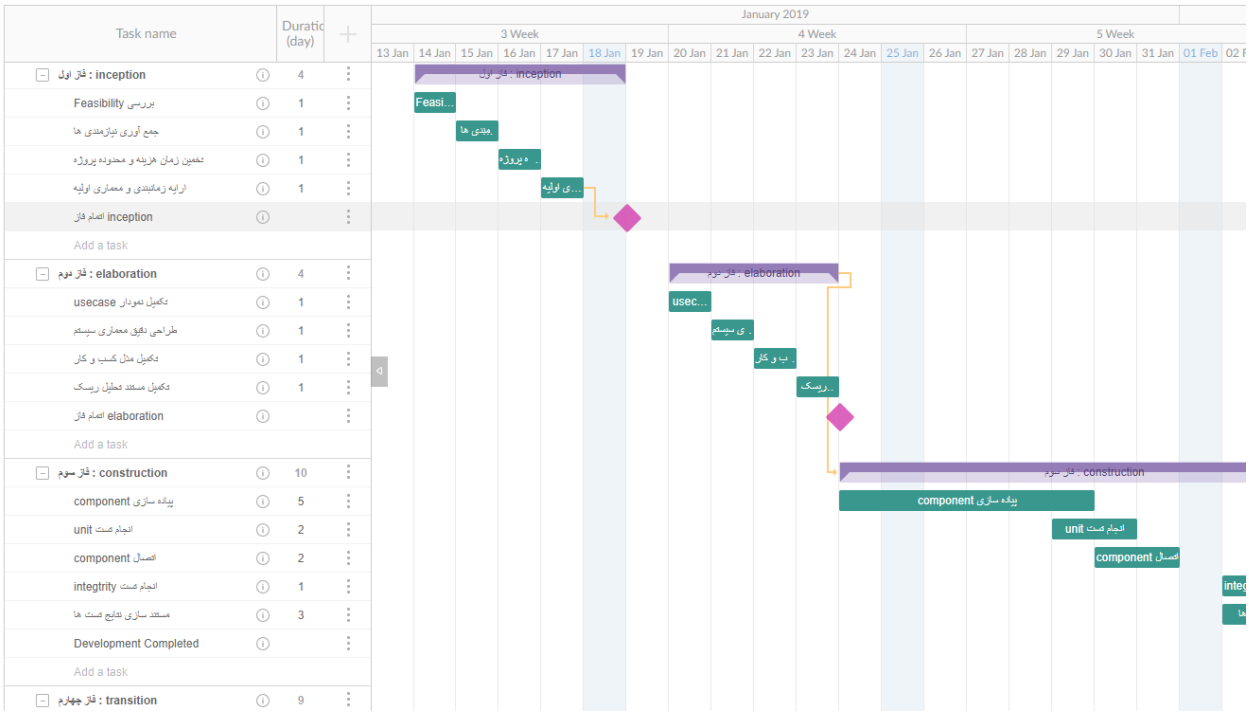
Login – build_MC – change_param– see_state

گام بعدی این تست شامل شناسایی کلاس های collaborator است که میتوان از مدل CRC تولید شده استفاده کرد. در این جا کلاس های user- Medical center-parameter در این فعالیت های مشارکت دارند .در این گام پیام های تولید شده و اثر هر کدام را در usecase انتخابی بررسی میکنیم. یک نوع اجرا برنامه میتواند به این صورت باشد که به ازای درخواست login از سرور (طبق کلاس دیاگرام) یک پیغام با دو مقدار صحیح برای کاربر ارسال شود که نتیجه این پیغام redirect کاربر به صفحه شخصی است . در صفحه شخصی با کلیلک بر روی دکه ساخت MC پیغام build_MC برای سرور ایجاد میشود و سرور پس از چک امکان ساخت یک پاسخ شامل یک عبارت بولی با مقدار صحیح برای کاربر میفرستند که کاربر با دریافت این پیغام به صفحه configure MC هدایت میشود و در آن جا با انجام تنظیمات مختلف MC خود را میسازد . پس از تایید ساخت MC توسط کاربر ، تنظیمات وی در قالب چندین پیغام change_param برای سرور ارسال میشود و سرور در پاسخ به هر کدام یک ack میفرستد.با اتمام اراسال ویژگی ها کاربر یک درخواست see_State ارسال میکند که به صفحه نمایش وضعیت redirect MC میشود. با توجه به این توضیحات یک تست کیس میتواند به شرح زیر باشد:

TESTCASE : Login(id , pass) . build_MC() . change_param(MC_id,param_id,value) . see_state(MC_id)

زمان بندی پروژه

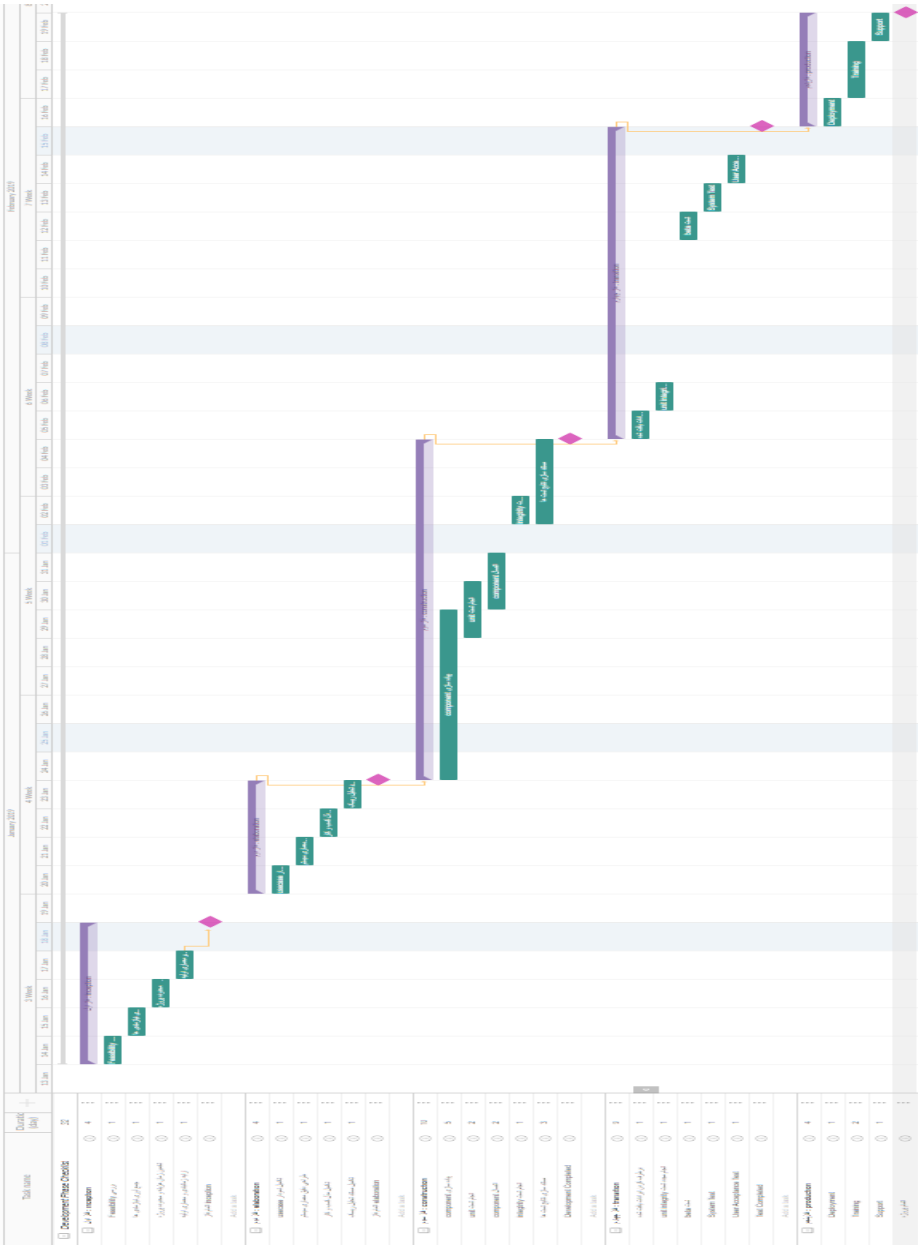
به کمک نمودار گانت چارت میتوانیم یک دید کلی از وضعیت پروژه و مسیر های بحرانی داشته باشیم. به همین منظور تمامی فاز های پروژه را در این قالب ر سم کردیم. در مدل ر سم شده روزهای کاری شرکت از شنبه تا چهارشـــنبه از ســـاعات ۸-۱۲ و ۱-۶ اســـت. وبا توجه به این که ۵ فاز پروژه بازه کوتاهی دارند، میتوان milestone ها را همان اتمام فاز های پروژه درنظر گرفت. این نمودار به چهت خوانا بودن یک بار به طور کلی و بار دیگر به صورت زوم شده قرار داده شده است



تصویر کلی نمودار در صفحه بعد قرار داده شده است

علت فاصله زیاد در فاز transition این است که در صورت ایجاد مشکلی در روند تولید تا ۴ روز بتوانیم milestone ها را extend کنیم.

فایل اصلی نمودار در پروژه قرار داده شده است .



برنامه System Configuration Management

تغییرات در تولید سیستم های نرم افزاری جز جدایی ناپذیری از فرایند تولید هستند . به همین علت ما نیاز داریم تا این تغییرات را به نحو مناسب مدیریت کنیم. در ادامه برنامه مدیریت این پروژه برای اعمال این تغییرات را ارایه میدهیم. در این برنامه ابتدا عوامل نیاز به تغییر را بررسی میکنیم و سپس افراد تیم و وظایف هر کدام از اعضای تیم SCM را می شناسیم .سپس به بررسی اجزا مختلف در این سیستم ها میپردازیم و در نهایت نیز به معرفی فرایند موردنیاز برای اعمال تغییر می پردازیم :

- **عوامل تغییر پیکربندی**

۱. پیکربندی مجدد سیستم
۲. نیازمندی جدید stackholder ها
۳. تغییر در بازار تجاری
۴. محدودیت هزینه و زمان

- **اعضای تیم**

این افراد عضو تیم ECCO هستند :

نقش	وظیفه
مدیر پیکربندی	-نظارت بر روی اجرای درست روال ها و خط مشی های موجود برای هر بخش از فرایند -تعریف روال قابل اعمال بودن تغییر
مدیر پروژه	-نظارت بر زمانبندی توسعه محصول -نظارت بر عملکرد کلی تیم
مهندس نرم افزار	-تولید و نگه داری و اعمال تغییرات در کد -ذخیره دقیق اجزایی از پروژه که شامل تغییرات شده اند
مشتری	-کار کردن با برنامه -درخواست اعمال تغییرات

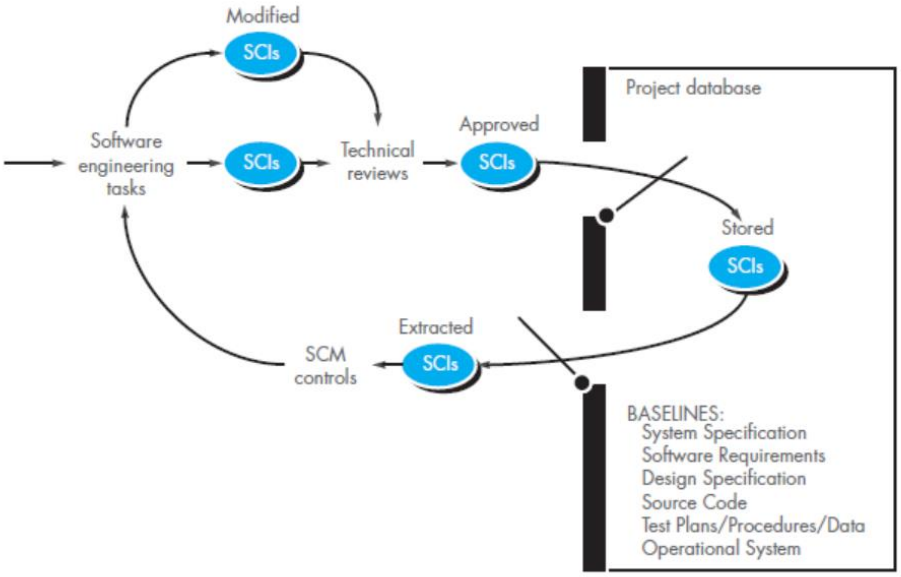
- **آیتم های مدیریت پیکربندی**

در این بخش ابزار های مختلفی که برای مدیریت پیکربندی در سیستم نیاز است و نوع مورد استفاده آن ها در این پروژه را معرفی میکنیم

۱. ابزار مدیریت کامپوننت : در این جا از MySQL استفاده می کنیم
۲. ابزار مدیریت سازه ها : این ابزار برای تسهیل تولید سیستم استفاده میشود و در این جا از Ms. Visual Studio 2018 استفاده میکنیم.
۳. ابزار مدیریت نیروی انسانی : در این پروژه از ابزار Ms. Team Foundation استفاده میکنیم.

- **فرایند مدیریت پیکربندی**

در بخش بعدی به طور کامل به این مبحث پرداخته ایم .



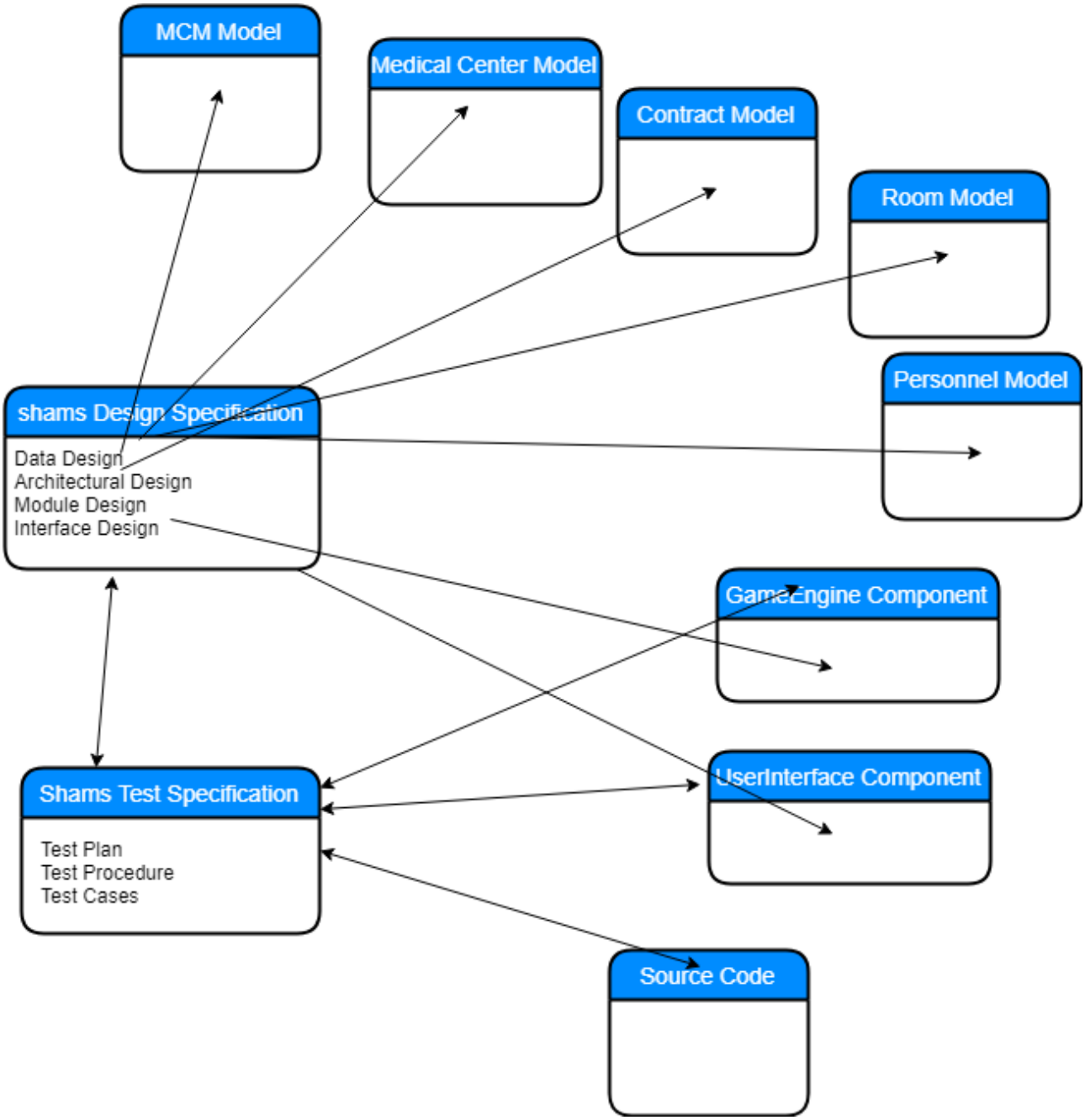
فرایند کنترل تغییرات

به طور کلی این فرایند به این صورت رخ میدهد که ابتدا در بخشی از سیستم نیاز به تغییر حس شود و developer این درخواست کاربر را بررسی و در قالب change report دربیاورد. این درخواست تغییر در CCA ارزیابی میشود. درصورتی که رد بشود، کاربر را آگاه میکنیم و در صورت پذیرفته شدن تغییر ، تیم ECCO با هدف تعریف دقیق محدودیت های موجود و تغییرات مورد نیاز ایجاد میشود. سپس مسیولیت انجام این کارها بر عهده اعضای مختلف تیم قرار میگیرد و تغییرات لازمه صورت می گیرد و بازبینی برای این تغییرات ایجاد شده انجام میشود. حال لازم است تا این تغییرات را در نسخه جدید برنامه قرار دهیم. برای انجام این کار ابتدا item هایی که شامل تغییرات شده اند را می یابیم . سپس یک baseline برای انجام تست ها می سازیم. و فعالیت های تست و QA لازم را بر رویش انجا می دهیم. در نهایت ورژن مناسب را با این تغییرات دوباره می سازیم و بازبینی را انجام میدهیم و در صورت عدم وجود خطایی ورژن جدید برنامه را بر روی پایگاه داده ذخیره میکینم.

این فرایند به تفکیک شامل ۵ مرحله است مرحله اول identification نام دارد. در این مرحلهattribute ها و صفات و قابلیت های object های مختلف موجود در پروژه و همین طور بخش های قابل تغییر obj ها شناسایی میشود و با توجه به این ها میتوان تغییرات و تکاملات جدید را برای هر object نگه داشت..مرحله دوم change control نام دارد. در این مرحله تغییرات درخواستی را بررسی و در صورت موافقت با انجام تغییر ، اعمال مورد نیاز برای تغییر را انجام میدهیم مرحله سوم کنترل ورژن نام دارد. در این جا برای هر کدام از object های موجود ، پیکر بندی ،تغییرات رخ داده ، عوامل تغییر و هر نوع اطلاعات موردنیاز را در قالب ورژن های مختلف نگه داری میکنیم مرحله چهارم configuration audit نام دارد که در این بخش بازبینی برای اطمینان از صحت تغییرات و عدم وجود defect در این ورژن را بررسی میکنیم. مرحله آخر نیز گزارش دهی است . که در این مرحله تغییرات اعمال شده را مستند سازی و مکتوب میکنیم

مدل محتوا SCM Repository

این نمودار را با توجه به کلاس دیاگرام و کامپوننت دیاگرام پروژه به صورت زیر می باشد:



متریک های چرخه حیات

همان طور که میدانیم ، متدولوژی برای ما تمامی روش ها ، فرایند ها ، ابزار ها و متریک های موردنیاز را مشخص میکند . در این مورد نیز ، برای یافتن متریک های مناسب به سراغ RUP رفتیم. برای این کار، به سراغ کتاب ها و منابع این متدولوژی رفتیم . متریک های موجود درکتاب Software Project Management: A Unified Framewor اثر Addison Wesley Longman بررسی شد. در این کتاب چندین و چند نوع متریک برای گام های مختلف پروژه ارایه شـــده بود. بیشـــتر این مدل ها پیچیدگی بالایی داشـــتند و برای پروژه هایی با ابعاد بزرگتر کارآمد بودند. در این کتاب برای اندازه گیری فرایند های پروژه دو مجموعه متریک معرفی کرده بود که مطالعه شد و بر مبنای آنها متریک های زیر برای تمامی گام های این پروژه انتخاب شد

Metrics	**Mesure**	**توضیحات**
Duration	ساعت	کل زمان صرف شده برای این گام
Effort	نفر-ساعت	کل effort مورد استفاده
Output	تعداد artifact * سایز	برای اندازه گیری سایز artifact ها از استاندارد های یاد شده در SQAP استفاده میکنیم
environment usage	قلم-ساعت	میزان استفاده از تمامی منابع مانند CPU – حافظه – تجهیزات -... :
correction rate	نفر-ساعت	مقدار زمان در این گام که صرف از بین بردن خطاها شده است
Change request	تعداد* LOC	به نوعی حجم تغییرات موردنیاز را برای هر گام می رساند
Review rate	تعداد جلسات بازبینی * طول جلسه * تعداد defect	این معیار کیفیت جلسات بازبینی را مورد بررسی قرار میدهد
Process deviation	$\frac{\sqrt{(t-T)^2}}{T}$	این معیار میزان انحراف از معیار این فرایند را نسبت به زمانبندی مقرر مشخص میکند T=زمان مورد انتظار t = زمان واقعی

برای هر گام نیز علاوه بر متریک های کلی که در جدول بالا آمده است ، متریک های زیر نیز باید اضافه شود که چون برای هر گام به نوعی تفسیر میشد در این بخش جداگانه آورده ایم

Phase	Metrics	Mesure
Inception	Size کیفیت	تعداد usecase ها تعداد actor ها تعداد defect ها
Elaboration	Size کیفیت	تعداد کلاس ها تعداد متدها(internal, external) Class fan-out (معیار coupling در class diagram)
Consrtruction	Size کیفیت	نفر-ساعت کار انجام شده برای پیاده سازی معیار LCOM (معیاری برای نبود cohesion در کلاس ها)
Transition	Size Quality	تعداد تست کیس ها ساعت های آموزش تعداد defect های یافته شده
Production	Size Quality	LOC تعداد defect های گزارش شده توسط مشتری

Function Point

طبق روش گفته شده در کتاب پرسمن عمل میکنیم. ابتدا برای ۵ معیار های گفته شده ، مقادی را تعیین میکنیم و سپس به معیارهای VAF پاسخ میدهیم و در نهایت در رابطه اصلی جایگذاری میکنیم.

دامنه اطلاعاتی	تعداد	وزن	FP
تعداد ورودی های خارجی-Els	۴	۳	۱۲
تعداد خروجی های خارجی-Eos	۳	۵	۱۵
تعداد درخواست های خارجی-EQs	۷	۶	۴۲
تعداد فایل های منطقی داخلی-ILFs	۳	۱۰	۳۰
تعداد فایل های واسط خارجی-EIFs	۱	۵	۵
جمع FP			۱۰۴

حال به پاسخ به ۱۴ سوال مربوط به adjust این معیار میپردازیم :

سوال	ارزش	سوال	ارزش
۱.اهمیت backup – recovery ؟	۵	۸.ILFs انلاین بروز رسانی؟	۷
۲.ارتباطات خاص داده؟	۱	۹. درخواست پیچیده ورودی/خروجی؟	۱
۳.پردازش توزیع شده؟	۷	۱۰.پردازش داخلی پیچیده؟	۳
۴.performance حیاتی ؟	۷	۱۱.reusable ؟	۵
۵.محیط اجرایی سنگین ؟	۳	۱۲.تغییرات و نصب در طراحی دیده شده؟	۳
۶.انتقال داده آنلاین ؟	۹	۱۳.دفعات نصب متعدد؟	۱
۷.نیاز به تراکنش؟	۳	۱۴.همگام با راحتی کاربر؟	۳

حاصل جمع این مقادی نیز برابر با ۵۸ خواهد بود. حال در فرمول محاسبه FP جایگذاری میکنیم :

$$fp = count\ total * [0.65 + 0.01 * \sum (F_i)] = 104 * [0.65 + 0.01 * 58] = 128$$

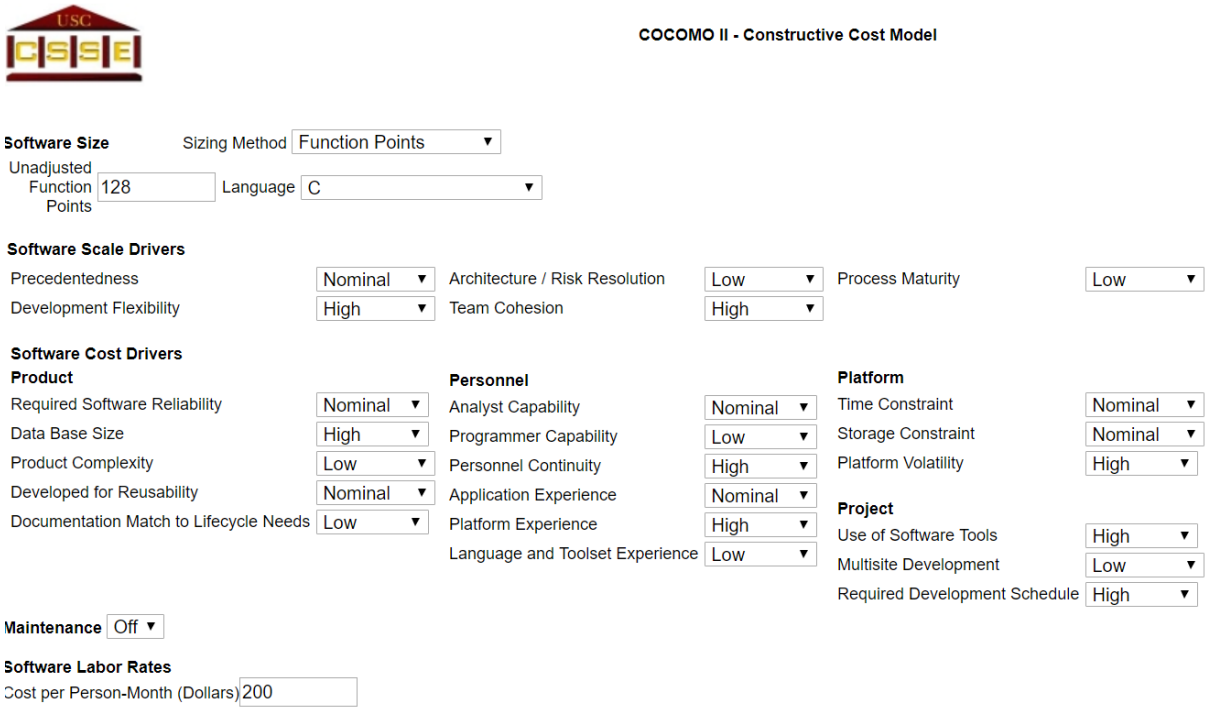
پس function point این پروژه برابر با ۱۲۸ است.

تخمین پروژه

طبق آموزش های این درس ،برای تخمین پروژه چندین تکنیک وجود دارد ، دسته اول مبتنی بر مدل های تجربی هستند که ورودی آن ها function point است و خروجی آن ها effort مورد نیاز برای پروژه

دسته دوم استفاده از historical data است ، یعنی از اطلاعات پروژه های مشابه قبلی استفاده کنیم. دسته سوم استفاده از ابزارهای اتوماتیک است همچون COCOMOII و دسته آخر استفاده از روش decomposition است که بر دو اساس قابل انجام است: بر اساس problem ها و بر اساس فرایند ها.

در این پروژه ما از تکنیک ابزار اتوماتیک برای تخمین پروژه استفاده میکنیم.این یک ابزار آنلاین است که در آدرس[۲] قرار دارد.



[۲] http://csse.usc.edu/tools/COCOMOII.php

با توجه به محاسبات FP در مرحله قبل و همین طور وضعیت فعلی تیم تولید، مثل عدم آشنایی زیاد تیم به نوع پروژه ، آشنایی کامل تیم با زبان C# و همین طور تازه کار بودن تیم این تنظیمات انجام میشود. حقوق پایه برای هر فرد ۲ میلیون تومان (۲۰۰ دلار) در نظر گرفته شده است. حاصل تخمین با این تنظیمات در به شرح زیر خواهد بود :

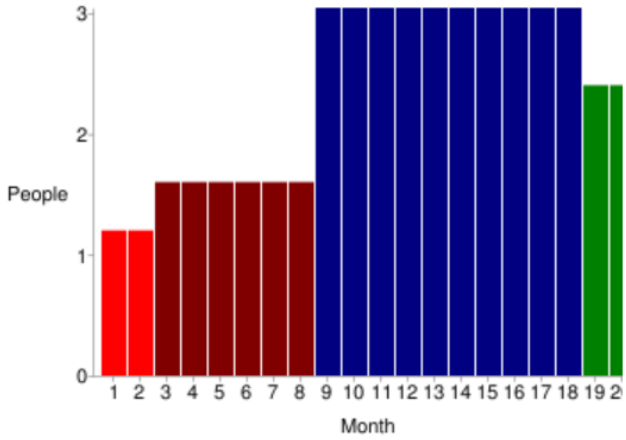
Results

Software Development (Elaboration and Construction)

Effort = 40.5 Person-months
Schedule = 16.2 Months
Cost = $8097

Total Equivalent Size = 10240 SLOC

Staffing Profile



Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	2.4	2.0	1.2	$486
Elaboration	9.7	6.1	1.6	$1943
Construction	30.8	10.1	3.0	$6154
Transition	4.9	2.0	2.4	$972

Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.3	1.2	3.1	0.7
Environment/CM	0.2	0.8	1.5	0.2
Requirements	0.9	1.7	2.5	0.2
Design	0.5	3.5	4.9	0.2
Implementation	0.2	1.3	10.5	0.9
Assessment	0.2	1.0	7.4	1.2
Deployment	0.1	0.3	0.9	1.5

همجنین تخمین را با استفاده از مدل تجربی small project regression model انجام میدهیم که به صورت روبرو خواهد بود:

Effort = -12.88 + 0.405 * FP = 0.405*128 -12.88= 51.84-12.88 = 38.96

که این نتیجه کاملا با نتیجه تست cocomo مطابقت دارد هر دو در حدود ۴۰ نفر تخمین زده اند و این نشانه صحت تخمین ما در این پروژه می باشد .

شناسایی ریسک ها

برای شنا سایی ریسک ها در این پروژه از روش چک لیست ا ستفاده میکنیم. در این بخش نیز چون تیم تجربه کافی ندارد ، پس از چک لیست های آماده و استاندارد موجود استفاده میکنیم. چک لیستی که در ادامه استفاده شده است مربوط به این[۳] کتاب است.

در این کتاب چک لیست ریسک هایی که بیشترین شکست را برای پروژه های نرم افزاری ایجاد کرده اند به شکلی که در ادامه میبینیم لیست شده است. در این جا برای هر بخش چند سوال برای بررسی این که این ریسک تا چه حد برای این پروژه وجود دارد، طبق چک لیست [۴] طراحی شده است:

نام ریسک	سوال	وضعیت
Personnel shortfalls *کمبود پرسنل*	آیا effort لازم پروژه در حال حاضر وجود دارد؟	بله
	آیا افراد آموزش های لازم را دیده اند؟	خیر
	آیا افرادی در پروژه هستند که به صورت پاره وقت باشند؟	بله
Unrealistic schedules and budgets *برنامه ریزی و بودجه بندی غیر واقع گرایانه*	آیا تمام وابستگی بین فعالیت ها پروژه در زمانبندی لحاظ شده است؟	بله
	آیا تمام کامپوننت ها در تخمین حضور داشته اند	خیر
	آیا شرکت به اندازه حجم این پروژه نیرو در ماه دارد (حجم پروژه بر مبنای نفر-ساعت در تخمین)	خیر
	آیا این برنامه دریافت بودجه، منابع را به موقع به پروژه میرساند؟	بله
Developing the wrong functions and properties	آیا نیازمندی ها داکیومنت شده است؟	بله

[۳] Software Risk Management: Principles and Practices

[4] https://www.slideshare.net/Kshitijyelkar/risk-identification-checklist?from_action=save

شناخت اشتباه کاربرد مورد نیاز مشتری	آیا نیازمندی ها کاملا واضح و قابل فهم اســـت؟ (همــه افراد درک یکسانی دارند)	بله
	آیا نیازمندی ها درگذشــته ســابقه تغییر دارند؟	بله
Developing the wrong user interface *ساخت اشتباه UI*	آیــا user interface بــا روش prototype تست شده است؟	خیر
	آیا مدل ذهنی مشـــتری بررســـی شده است ؟	خیر
	آیا سناریوهای سیستم در رابطه با UI بررسی شده اند ؟	بله
Gold plating *قابلیت اضافی*	آیا تیم سابقه اضافه کردن قابلیتی بدون وجود نیازمندی را دارد؟	بله
Continuing stream of requirements changes *تغییرات در نیازمندی به صــورت پی در پی*	آیا تجربه کار با این مشتری نشان میدهد که دایما نیاز هایش را تغییر می دهد ؟	خیر
	آیا تیم درک یکســانی از نیازمندی های پروژه دارند؟	بله
	آیا درخواســت شــناخته شــده ای برای تغییر هســـت که به تعویق افتاده باشد ؟	خیر
Real-time performance shortfalls *کــمبــود performance بــرای اجراهای real time*	آیا نیازمندی مربوط به حجم CPU و حافظه مورد نیاز بررسی شده؟	بله
	آیا محدودیت سرعت پردازنده چک شده است ؟	بله
	آیا نمونه مشابه پروژه وجود دارد؟	بله

بررسی ریسک های پروژه

در بخش ابتدا ریسک های موجود و شدت تاثیر گذاری آن ها را می یابیم (اثرگذاری ریسک ها در بازه ۱ الی ۴ بررسی میشوند ، کمترین اثرگذاری ۴ بیشترین ۱) . سپس احتمال وقوع هر کدام از ریسک ها را در پروژه(با توجه به چک لیست بخش قبلی) می یابیم . در نهایت بر اساس بودجه و زمان در دسترس تصمیم میگیریم که به چه تعداد از این ریسک ها به صورت proactive و به کدام یک به صورت reactive پاسخ دهی کنیم.

ریسک	اثرگذاری	هزینه جبران	احتمال وقوع	دسته
کمبود پرسنل	۱	۱۲ میلیون	۶۰٪	SS
برنامه ریزی و بودجه بندی غیرواقع گرایانه	۲	۱۰ میلیون تومان	۸۰٪	BI
شناخت اشتباه کاربرد مورد نیاز مشتری	۱	۲۰ میلیون تومان	۴۰٪	SC
تولید قابلیت اضافی	۴	۲ میلیون تومان	۵۰٪	SS
تغییرات پی در پی نیازمندی مشتری	۳	۵ میلیون تومان	۶۰٪	PS
وابستگی به عوامل خارج از کنترل پروژه	۲	۱۰ میلیون تومان	۲۰٪	PD
جدید بودن فناوری مورد استفاده	۴	۳ میلیون تومان	۷۰٪	DE
ساخت اشتباه UI	۳	۸ میلیون تومان	۵۰٪	SC

جدول ریسک

دراین بخش ریسک ها را با توجه به حاصل ضرب هزینه در احتمال وقوع مرتب کردیم و سپس یک Treshold با اندازه ۳ برای cut off range تعیین میکنیم.

ریسک	اثرگذاری	هزینه جبران	احتمال وقوع	دسته
برنامه ریزی و بودجه بندی غیرواقع گرایانه	۲	۱۰ میلیون تومان	۸۰٪	BI
شناخت اشتباه کاربرد مورد نیاز مشتری	۱	۲۰ میلیون تومان	۴۰٪	SC
کمبود پرسنل	۱	۱۲ میلیون	۶۰٪	SS
ساخت اشتباه UI	۳	۸ میلیون تومان	۵۰٪	SC
تغییرات پی در پی نیازمندی مشتری	۳	۵ میلیون تومان	۶۰٪	PS
جدید بودن فناوری مورد استفاده	۴	۳ میلیون تومان	۷۰٪	DE
وابستگی به عوامل خارج از کنترل پروژه	۲	۱۰ میلیون تومان	۲۰٪	PD
تولید قابلیت اضافی	۴	۲ میلیون تومان	۵۰٪	SS

برنامه مدیریت ریسک

مدیریت ریسک یک فعالیت دایمی در پروژه است . برای مدیریت ریسک یک چرخه وجود دارد که شامل ۵ مرحله شناسایی، آنالیز ، برنامه ریزی، track ، کنترل است . در بالا مراحل شناسایی و آنالیز ریسک ها را انجام دادیم. برای مرحله plan به ساخت risk information sheet برای ریسک های انتخاب شده میپردازیم.
پس از انجام برنامه ریزی ، ریسک را در مراحل مختلف پروژه دنبال میکنیم و در فاز کنترل بررسی میکنیم که آیا فعالیت انجام شده موثر بوده است یا نه و وضعیت موجود بهینه میشود

همان طور که در بالا گفته شد ، برای ریسک های انتخابی risk information sheet ایجاد میکنیم:

Risk information sheet			
Risk ID : ۱	Data :۹۷,۱۰,۲۳	Prob : 80%	**Impact** : ۴
Description : **برنامه ریزی و بودجه بندی غیرواقع گرایانه**			
Refinement : **Subcondition1 :** **تخمین حجم پروژه به درستی انجام نشده است** **Subcondition2 :** **نیازمندی های مشتری به درستی شناسایی و مدل نشده است**			
Mitigation : ۱. **استفاده از متدولوژی های incremental** ۲. **برنامه ریزی و بودجه بندی با جزییات کامل بر اساس milestone cost**			
Management/ contingency plan/ trigger : **هرگاه هزینه و یا زمان فرایندی deviation بیش از ۵٪ داشت. برنامه ریزی و بودجه بندی دوباره انجام شود**			
Current status : mitigation شروع			
Originator : زهرا دهقانیان		**Assigned :-**	

Risk information sheet			
Risk ID : ۱	Data :۹۷,۱۰,۲۳	Prob : 80%	**Impact** : ۴
Description : **کمبود پرسنل**			
Refinement : **Subcondition1 :** **تعدادی از افراد به طور ناگهانی از تیم خارج شوند** **Subcondition2 :** **تغییر در نیازمندی های مشتری ، سبب افزایش حجم پروژه و کمبود پرسنل شود**			
Mitigation : **۱. در تخمین نیازمندی ها تا ۲۰٪ تغییر را درنظر بگیریم و بر طبق آنبرنامه ریزی را انجام دهیم** **۲. هنگام قرارداد با پرسنل ، برای خروج از تیم ، جریمه ای معادل ۵۰٪ ضرر وارده را به عنوان خسارت دریافت کنیم**			
Management/ contingency plan/ trigger : **در صورت بروز این مشکل، برنامه ریزی و تخصیص کارها به افراد را دوباره انجام دهیم. در صورت نیاز نیروی جدید جذب پروژه بکنیم**			
Current status : mitigation شروع			
Originator : زهرا دهقانیان		**Assigned :-**	

Risk information sheet			
Risk ID : ۱	Data :۹۷,۱۰,۲۳	Prob : 80%	**Impact** : ۴
Description : **شناخت اشتباه کاربرد مورد نیاز مشتری**			
Refinement : **Subcondition1 :** **تخمین حجم پروژه به درستی انجام نشده است** **Subcondition2 :** **نیازمندی های مشتری به درستی شناسایی و مدل نشده است**			
Mitigation : **۱. در جلسات شناسایی نیازمندی ها، از prototyping استفاده کنیم**			
Management/ contingency plan/ trigger : **کاربرد موردنیاز مشتری مدل شود و به عنوان ورودی increment بعدی اعمال شود.**			
Current status : mitigation شروع			
Originator : زهرا دهقانیان		**Assigned :**-	