بسمه تعالی

دانشگاه صنعتی همدان

جزوه کامل

تحلیل و طراحی سیستم ها

استاد: دکتر حسن بشیری

دستیار استاد: امیرحسین بابائیان

مسئول محتوا: ساجده مقامی

ویراستاران: سارا اسماعیلی، علی اسدی مهران

همکاران در تهیه جزوه: سارا اسماعیلی، ایمان احمدی، کیارش عظیمی پور، مهدیس اکبری جهانی، نرگس میرزایی، محدثه احمدی، مهدی سرمدی، مهدی غلامی اشرف، زینب شیری، رویا منوریان، مهدی افاضل

طراح صفحه و تنظیم کننده: علی اسدی مهران

ویراست اول

زمستان ۱۳۹۹

فهرست

فصل ۱

IT

ترکیب محصولات و خدماتی که به شکل سخت افزار و نرم افزار ارائه می شود که بخش اصلی اول تمرکز روی اطلاعات در غالب مدیریت اطلاعات، دستیابی به اطلاعات ارتباط با آنها و اشتراک گذاری آنها را می گویند.

اوایل سال ۱۹۷۰ نرم اغزارهایی نوشته می شدند که در قالب برنامه های کامپیوتری تحویل کارفرما می شدند.

کارفرماها از تعدادی از مهندسین می خواستند که اشکالات این برنامه ها را برطرف کنند در واقع در قالب باگ یک سری گزارشات خطا می دادند، که کار نرم افزار بدرستی انجام نمی شود و تیم نرم افزاری موظف به نگهداری از برنامه نوشته شده بود.

طی این مراحل برای کارفرما هزینه زیادی را به همراه داشت.

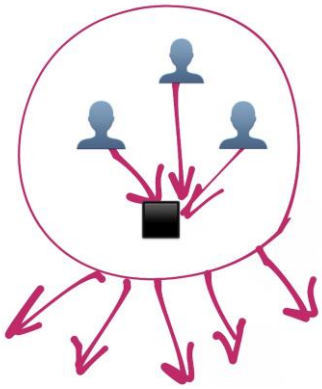
این اتفاقات منجر به بحران نرم افزار شد که تصمیم گرفتند نرم افزار را در قالب یک فرآیند مهندسی شده ایجاد و توسعه دهند تا سیستم اطلاعاتی را باکیفیت مناسب توسعه دهند.

نیاز کارفرما یا کسب و کار برای اینکه منجر به تبدیل محصول شود که آن محصول بتواند خدماتی را ارائه بدهد باید فرآیندی را طی کند که این طی این فرآیند را توسعه نرم افزار می گویند، که analysis and design یکی از گام های مهم است.

برای آغاز پروژه و برنامه ریزی یک سری کارها را انجام می دهیم سپس سعی می کنیم سیستم را تحلیل کنیم و راه‌حلی را طراحی کنیم و عد پیاده سازی کنیم و مراحل بعدی تست، استقرا و نگهداری ست.

روش هایی وجود دارند که میگوییم این فرایند یک فرایند چرخه ای باشد و تکرار شود.

سیستم های اطلاعاتی



فناوری افراد و دیتا اولین کارکردی که ما از کسب و کار انتظار داریم را برای ما برآورده می کند.

افراد در سیستم وجود دارن و تلکنوژی وجود داره که ارتباط بین افراد و اطلاعات(دیتا) را به یکدیگر متصل می کند در نهایت سیستم ما خدماتی را به بیرون(مشتری، بیزینس های دیگر) ارائه می دهد.

تحلیل گر سیستم چه کارهایی باید بکند؟ چطور باید برای پروژه ها برنامه ریزی کند؟

- سیستم را توسعه بدهد
- سیستم را نگهداری کند
- پروژه‌های آیتی را مدیریت کند
- جلسات را رهبری کند.(وظیفه تحلیل گر این است که اطلاعات را جمع آوری کند و در جلسات presentکند و نظرات بقیه را بپرسد.)

سیستم: جموعه ای از اعضا که هدف خاصی را دنبال می کنند.

اجزا » باهم ارتباط دارند-ورودی دریافت می کنند- خروجی تولید می کنند-در یک مرز محدود شدن- اینترتی دارند.

همه ی سیستم ها نیاز به دریافت داده بعنوان ورودی هستند.

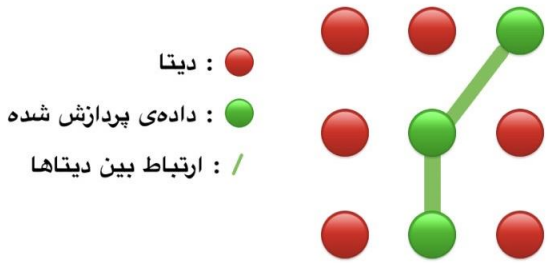
Data: داده ی خام(بود ها – هر چیزی که هست) Data base

Information: داده ی پردازش شده. Data + process

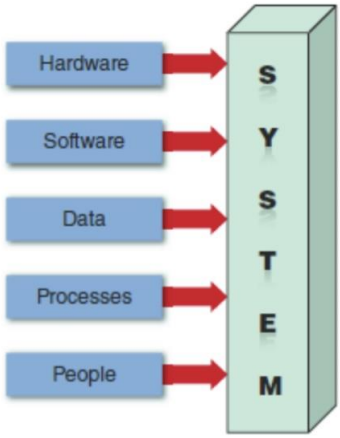
Knowledge: اگر داده های پردازش شده باهم ارتباط پیدا کنند که ما را به آگاهی و دانشی برسانند.

مثال: شماره دانشجویی ۹۷۷۱۰۳۱

- داده: شماره دانشجویی
- پردازش:
- دو رقم اول(۹۷): سال ورود
- دو رقم دوم(۷۱): دانشکده
- سه رقم سوم(۰۳۱): نفر چندم لیست
- دانش: متوجه میشویم دانشجو ورودی چه سال و چه رشته و نفر چندم بوده



ممکن است نتوانیم از یک اطلاعات دانشی بدست بیاریم وقتی این اطلاعات کنارهم قرار بگیرند به دانش برسیم.



Process: فلوچارت » مجموعه ای از کارها تا یک خدمت ارائه شود. در حوزه کسب و کار work flow هم گفته می شود. همه ی اینها بیانگر ایده ای هستند که قرار است داخل یک سازمان انجام شوند. از یک نقطه شروع می شود و یک سری کارها روی آن انجام می شود تا به یک انتهایی می رسد روی این مراحل پردازش انجام می شود و داده ها را بعنوان ورودی دریافت می کند و در نهایت منجر به خروجی می شود.

People: stakeholders (ذینفعان-سهام داران) کسانی که درگیر سیستم هستند و نفعی از سیستم میبرند. مثل: مشتری های سیستم،کارمندان سیستم،مدیرپروژه، تیم اجرایی

راهبردهایی که در کسب و کارهای اینترنتی وجود دارد ترند آن این است که به سرعت سمت جهانی شدن globalization حرکت می کنیم.

خود این مسئله راهنمایی ست که ما متوجه شویم وقتی کسب و کاری را می خواهیم طراحی کنیم آن را در سطح local اجرا کنیم و چشم انداز بلند مدت داشته باشیم که به سمت جهانی شدن پیش ببریم.

یک ترند دیگر خدمات ابری ست خیلی از سرویس ها را به صورت cloud داریم مثل افیس روی pc نداریم و از گوگل داکیونت یا درایو دریافت کنیم یا فضای بیشتر برای ذخیره سازی اطلاعات از drop box استفاده کنیم.

برخی از کسب و کارها به سمت سمت تجارت الکترونیک پیش رفتند که کمک می کند از طریق یک ui داده ها را بررسی کنند و سرویس مورد نظر خودشان را انتخاب کنند که از طریق pc و موبایل امکان پذیر است.

B۲C(Business-to-Customer): ارتباط مستقیم بین کسب و کارها و مشتری هاست. که مشتری با مراجعه به سایت محصول خود یا خدمات مورد نظر خودش را بررسی میکند که وجود دارد یا نه در صورت وجود انتخاب میکند به مقایسه قیمت، ویزگی و... میپردازد و در نهایت روش پرداخت را انتخاب می کند.

B۲B(Business-to-Business): ارتباط بین کسب و کارهاست. بیزینس کاری را انجام میدهد مثلا بیزنسی در حوزه هتلداری کار میکند و بیزینس دیگر در آژانس. هتل تعداد اتاق ها و قیمت رو به آژانس میدهد و آژانس اطلاعات رو به مشتریان خودش میدهد.

وقتی یک کسب و کار را تحلیل می کنیم، یکی از اولین گزینه هایی که باید سراغ آن برویم، این است که کسب و کار ماموریتش چیست؟ اهدافش چیست؟ کارکرد های اصلی آن چیست؟ ساختار و سازمانش به چه ترتیب هستند، چه محصولات یا خدماتی رو تولید میکند، مشتریانش چه کسانی هستند؟ تامین کننده های مواد اولیه یا هر ورودی که لازم دارد چه کسانی هستند؟ محدودیت ها و جهت گیری که کسب و کار دارد به کدام سو است؟

Business Profile

موضوع **Business Profile** چه در مورد کسب کارهایی که موجود هستند و ما می خواهیم راجع به آنها تحلیل کنیم، چه کسب و کارهایی که میخواهیم راه اندازی کنیم مثل استارتاپ ها بسیار موضوع کلیدی و مهمی است.

اینکه چگونه یک ایده را تبدیل به یک کسب و کار کنیم و Business Plan یا طرح کسب و کار رو تدوین کند، یکی از بخش های اصلی آن موضوع Business Profile هست که بتونیم اهداف و ماموریت ها را مشخص کنیم.

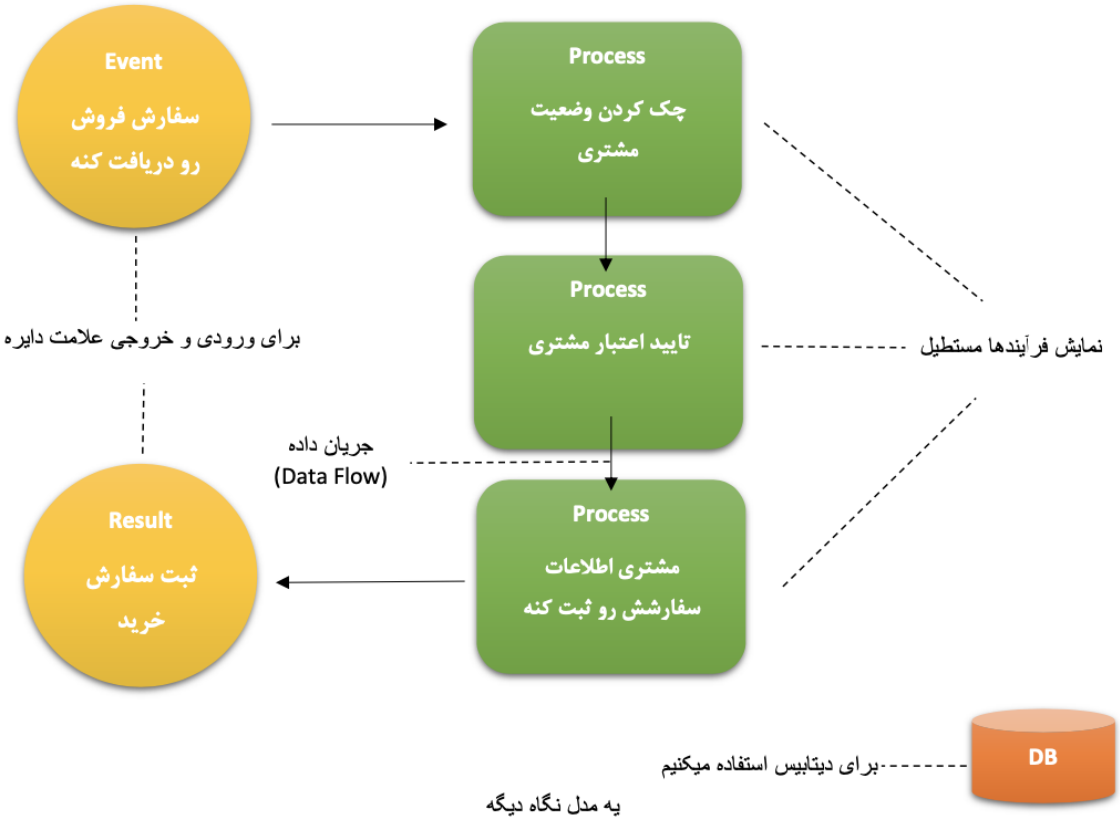
یک تصویری را به ما میدهد، اما ما گفتیم سرویس ها چه هستند و محصولات کدامند، اینکه چگونه محصولات در این کسب و کار تولید میشوند یا این خدمات به مشتریان ارائه می شود، مربوط به بحث Business Process است که یعنی فرایندهای کسب و کار چگونه است.

فرض کنید کسب و کار دانشگاه را به عنوان یک کار خدماتی، آموزشی، پژوهشی میخواهیم تحلیل کنیم وقتی می گوییم که فرآیندها کدامند به عنوان مثال ثبت نام، اینکه روز اول دانشگاه از کجا شروع کردیم، چه اطلاعاتی را از ما گرفتند، چه فرم هایی را تکمیل کردیم، چگونه برای خوابگاه اقدام کردیم، برای امور تغذیه، کارت دانشجویی، انتخاب واحد همان موقع انجام شد. این فرآیند یک مجموعه کار بود که به آن Process می گوییم که برای تعریف آن همین مراحل ثبت نام رو می گوییم که چه کار کردیم.

روش های توصیفی روش های پر خطایی هستند معمولا در مدل سازی Business Process Model فرآیندهای مان را مدل می کنیم.

مثل برداشتی که از زبان های برنامه سازی داریم می دانیم که زبان های برنامه سازی مختلفی داریم و زبان های مدل سازی متفاوتی داریم.

مثال زبان مدلسازی



کسب و کار پس دو قسمت اصلی دارد:

۱-Business Profile

2-Business Process

البته این یک نوع زاویه نگاه است:

مثلا یک سازمان از این زاویه

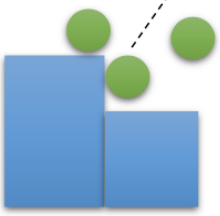
Process

Data

People

یه مدل نگاه دیگه

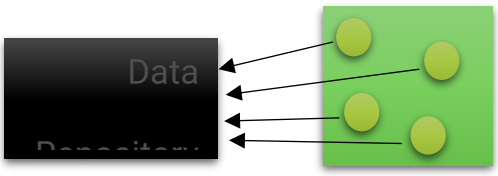
Object



سعی میکنیم اشیاء رو شناسایی کنیم و ارتباط بین اشیاء رو تشخیص بدیم ، اینکه چه پیام هایی رو بین هم رد و بدل میکنن

ولی در نهایت فرآیند یکی ست، زاویه نگاه فرق میکند ولی کار مدل سازی، پیاده سازی، تست مراحلی ست که در همه سیستم ها دنبال می کنیم.

روش هایی که الان موجود است: فرض کنید سیستم خیلی بزرگ است، هر مجموعه ای سامانه مربوط به خودش را استفاده میکند و دیتابیس خودش را دارد، مثلا گلستان کار خودش را انجام میدهد، هر بخش مجزا کار می کند مثل امور مالی، امور تغذیه؛ که Current method می شود (از همان سیستم هایی که هر کس در هر بخش که استفاده میکند تا بهره وری مورد نیاز اتفاق بیافتد)حالت ایده آل این است که ما بتوانیم همه ی این سیستمها را به صورت یک پارچه داشته باشیم یعنی یک دیتابیس باشد (Data Repository یا مخزن داده) بخش های مختلف سیستم نه تنها با این مخزن در ارتباط هستند بلکه در صورت لزوم با هم دیگر ارتباط دارند. اصطلاحا این ها را یکپارچه و به عنوان ERP از این نوع سیستم ها یاد می کنیم.



ER

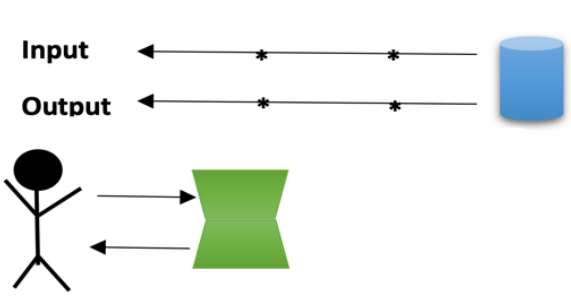
عملیات های گسترده سازمان رو مدیریت می کند از این نوع خدمات پشتیبانی می کند. طراحی سیستم های ERP مفید است ولی تحلیلشان بسیار پیچیده و در عین حال بسیار هزینه بر و زمان بر است. ایران خودرو و سایپا نمونه هایی از این دست هستند.

Transaction Processing System

یک دسته از سیستم های اطلاعاتی، سیستمهایی هستند که تمرکزشان بر پردازش تراکنش است (منظور داده هایی ست که روز به روز تولید می شوند) مثل: سیستم های بانکی.

تراکنش: به هر داد و ستدی که ما با دیتابیس داریم از یک نقطه شروع می شود، یک سری مراحل را طی میکند وارد دیتابیس می شود یک تغییراتی ایجاد میکند یا اطلاعات را از آنجا میخواند و برمیگردد.

مثل اینکه ما پشت دستگاه ATM هستیم و تقاضایی را به دستگاه میدیم (مثل برداشت پول) به شعبه مرکزی بانک می رود، بعد به بانک مرکزی وصل می شود، بعد از بانک مرکزی دیتابیس اطلاعات به روز می شود و به این دستگاه ATM دستور داده می شود که به میزانی که درخواست شده پول به مشتری بدهد.



(بعد از اینکه چک کردن های اولیه تمام شد) می گوییم تراکنش "commit" شده یعنی تراکنش خاتمه پیدا کرد.

این سوال پیش می آید که چه لزومی بود از بانک ما به بانک مرکزی وصل شود !؟

اطلاعات**Day-To-Day** یعنی ما به اطلاعات**real time** نیاز داریم و پردازش اطلاعاتمان کاملا باید اتفاق بیافتد و این به روزرسانی ها در دیتابیس های مختلف باید شکل بگیرند. به خاطر این به بانک مرکزی وصل می شویم که اگر به ATM دیگری مراجعه کردیم و خواستیم برداشتی از حسابمان انجام دهیم و آنATM مربوط به بانک ما نبود، بتواند خدمات را به ما ارائه بدهد و اطلاعات به روز را در اختیار داشته باشد.

Business Support System: سیستمهایی که پشتیبان کسب و کارهای مختلف هستند مثل MIS و تکنولوژی هایی مثل RFID

Knowledge Management: سیستم هایی که مدیریت دانش رو انجام می دهند(یکی از حالت های ایده آل سیستم هاست) در واقع ما به جای Data Base،**Knowledge Base** داشته باشیم.

وقتی Knowledge Base داریم یعنی یک دیتا به صورت خام نداریم، بلکه این دیتا را پردازش کردیم به یک سری قوانین رسیدیم، استنتاج ها و نتایج و گزارش های بسیار بهتر وبهینه تری را میتوانیم از سیستم بگیریم.

از Data Base گزارش می گیریم ولی گزارش ها عمدتا دیتا ها را به ما نشان میدهند نیاز به تفسیر انسانی روی دیتا وجود دارد. ولی در Knowledge Base گزارش هایی هست که میتوانند ما را در تصمیم گیری کمک کنند به نحوی دیتایی که در اینجا جمع شده مثل expert عمل میکند مثل افراد متخصص، این دسته از سیستم ها عمدتا محصول تکنولوژی های هوش مصنوعی و ماشین لرنینگ هستند، یعنی انسان خیلی علاقه مندست به جای اینکه دیتا را به صورت خام یا در سطح محدود، در سطح information در دیتابیس نگهداری کنیم، دنبال این هستیم که به صورت Knowledge نگهداری کنیم.

اتفاق هایی که توی سیستم میتونه بیافتد یعنی user productivity اند. تکنولوژی وجود دارد و تکنولوژی بهره وری را improve می کند ولی تمرکزش روی data sharing است.

یه سری ابزارها هستند که ما را در ترکیب اطلاعات دیجیتال کمک میکنند تا بهره وری سازمان را بالا ببریم.

یک شکل هم integration است که میتوانیم بخش های مختلف را روی دیتابیس های افراد و فرآیند ها رو با همدیگر integrate کنیم یا یک پارچه کنیم که بتوانیم بهره وری سیستم را بالا ببریم.

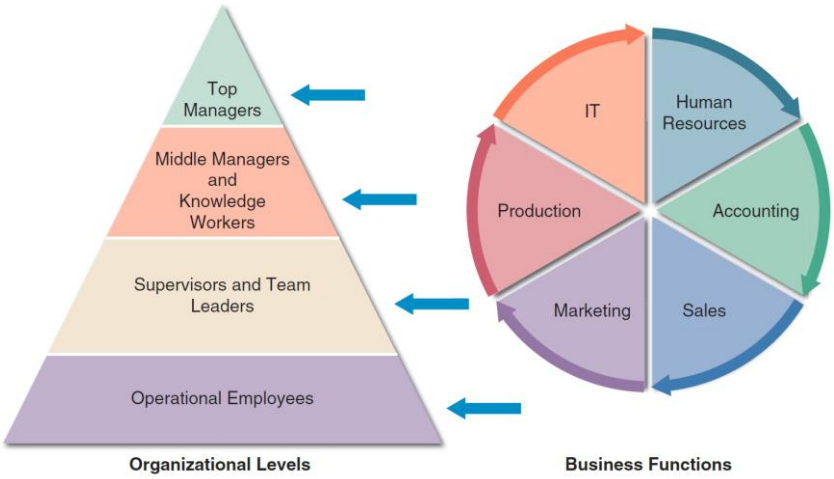
مدلهایی که برای سازمان هایی که عمدتا با اطلاعات کار میکنن وجود دارد، یعنی وارد هر سازمانی که می شویم سطوح سازمانی برای آن تعریف شده و یه تعداد مدیر بالا دستی در راس سازمان قرار دارند که کارشان این است

که برنامه های راهبردی و سیاست گذاری کلان سازمان را انجام بدهند، یک تعداد مدیران میانی داریم که واسط بین مدیران بالایی و نیروهای عملیتای ما هستند، سیاست های بالایی ها را پیاده می کنند و مشکلات پایینی ها رو منتقل می کنند.

۱- هیئت مدیره ی سازمان یا مدیران ارشد کار سیاست گذاری و تصمیم های کلان

۲- مدیران میانی

۴- نیروهای عملیاتی در صورتی که تعدادشان زیاد باشد به گروه هایی تقسیم می شوند که هر گروه توسط یه تعداد (۳- سرکارگر و مدیر پروژه) حمایت می شود و معمولا بخش هایی که برای این سازمانها متصور هستیم بخش مدیریت نیروی انسانی و مدیریت منابع است، بخش حسابداری و امور مالی، بخش فروش، بخش بازاریابی، بخش تولید » هسته اصلی شرکت و بخش مدیریت فناوری اطلاعات هستند.



تصور بسیاری این است که یک استارتاپ یک نمونه کوچک شده از یک کمپانی ست که غلط است، همه ی سازمان ها، بخش هایی که در صفحه قبل گفتیم را دارند حالا ممکن است، یکی؛ یک بخش کمتر و یکی بیشتر داشته باشد.

فکر می کنیم بخش تولید کلیدی ترین بخش یک سازمان است این موضوع درست است، یعنی همه بخش‌ها جمع شدند که این بخش product راه بیافتد اما هر کدام از این بخش ها نقش خودشون را باید خیلی خوب بازی کنند.

اگر ما یک محصول باکیفیت تولید کنیم ولی نتوانیم آن را به خوبی بفروشیم ue مجموعه نمیتواند خوب بهره ببرد، بنابراین باید به همه بخش ها توجه کنیم.

سیستم دانشگاه را در نظر بگیرید: یک تعداد افراد هستند که مسئول انجام یک سری خدمات هستند و بین این سرویس ها دیتا جابجا می شود.

عینکی که به چشم زده بودیم و این مدل را می دیدیم روشstructured analysis بود (روش ساختار یافته)در این نگاه دنبال این هستیم که در هر سیستم دیتا، process، people، را استنتاج کنیم و ارتباط بین آنها را درآوریم و معمولا از برنامه ای که برای انجام تحلیل و طراحی سیستم ها استفاده می کنیم پلن SDLC هست.

software development life cycle SDLC چرخه حیات توسعه نرم افزار

یعنی وقتی می خواهیم یک پروژه را شروع کنیم، این مراحل را طی می کند:

برنامه ریزی برای پروژه← تحلیل پروژه← طراحی راه حل← پیاده سازی طراحی← قسمت پیاده سازی← پشتیبانی محصول

Object-Oriented: در این نگاه سیستم را به صورت مجموعه ای از اشیاء میبینیم بنابراین مثلا اطلاعات درس، یک آبجکت است؛ دانشجو یک آبجکت برای سیستم دانشگاهی که مثال زدیم، استاد-دانشکده-گروه-مدیر گروه و..... آبجکت هستند و سعی می کنیم ارتباط بین این ها را استخراج کنیم و سیستم را به این روش تحلیل می کنیم.

تعریفی که از نگاه Object-Oriented برای سیستم وجود دارد این است که سیستم دانشگاه مجموعه‌ای از اشیائی مانند دانشجو، استاد و..... (دانشجو البته آبجکت نیست، کلاس است؛ کلاس مجموعه ای از اشیائی ست که دارای ویژگی ها و رفتارهای مشترک هستند)ولی آبجکت چندتا در سیستم هست که هویت منحصر به فرد دارند. رفتار دارند و یک سری دیتا یا atrebute دارند. مثلا در سیستم دانشگاه امین محمدی یه آبجکت است، رضا مرتضوی یه آبجکت است. اینها را باهم دسته بندی می کنیم در یک کلاس دانشگاه قرار میدهیم یا کلاسی به نام استاد قرار میدهیم.

روش Agile method: در روش قبل ← analyze plan ← design ← implement ←.....

این پیاده سازی تقریبا آخر افتاده بود. در این روش گفته می شود که ما باید توسعه نرم افزار را چابک کنیم و هرچه سریع تر نمونه اولیه را بیرون بدهیم. مثل روش های spiral

یک نسخه از محصول را ارائه می کنیم ولی تمرکز این است که ما در روش های agile ما به سرعت prototype بسازیم و به کاربر نشون بدهیم و با نیازهای کاربر آن ها را تطبیق بدهیم.

Prototyping: یعنی نمونه اولیه محصول در این روش وقتی پروژه ای به تیم محول میکنند که این پروژه نرم افزاری را انجام بدهیم ما درواقع تلاشمان این است که اطلاعات اولیه که اسکلت موضوع را فهمیدیم یا بخش کوچکی از موضوع را فهمیدیم یه طراحی سریع ارائه می کنیم و از طراحی یک دمو میسازیم و این دمو را به کاربران نشان میدهیم و فیدبک ها را دریافت میکنیم و اصلاحات رو انجام میدهیم. این چرخه را تا جایی تکرار می کنیم که روی دمو اتفاق نظر شکل بگیرد و بعد آنرا پیاده سازی می کنیم.

در جریان فرآیند توسعه یه تعداد ابزار ما رو کمک می کنند تقریبا بدن ابزار هیچ کدوم از رشته های مهندسی کارآمد و کارآیی نخواهند داشت. در حوزه مهندسی مکانیک مثلا ابزارهایی که استفاده می شود مثلا دستگاه تراش و وایرکات آچار و... باشد.

اما ابزارهایی که ما در حوزه نرم افزار استفاده می کنیم case tools هستند **computer aided software engineering** یعنی ابزارهای کامپیوتری کمک مهندس نرم افزار. مثلا ابزارهای دیگری برای مدل سازی می گوییم یا IDE یا ابزارهای code generators مثلا فلوچارت می کشیم ابزار کدش را می نویسد.

مدیر IT در راس هستند

Application development: توسعه اپلیکیشن یعنی فرض کنید مثلا وزارت علوم بخشی دارد تحت عنوان فناوری اطلاعات و تیمی هست که کارش توسعه نرم افزارست. البته این روش خیلی مرسوم نیست. خیلی از مواقع این کارها را Out Source میکنند یعنی تیم IT نیاز اطلاعاتی، اپلیکیشنی را به بیرون میدهد به شرکت های خصوصی مینویسند و تحویل میگیرند ولی حالا در بعضی از مراکز به دلایل تخلف مثلا حفاظت از داده ها و یا اینکه داده ها خیلی مهم هستند و افراد باید استخدام باشن مثل کاری که در وزارت دفاع انجام می شود و نیرو های برنامه نویس در خود شرکت کار می‌کنند.

System support and security: بخش امنیت خیلی از مکان ها این را دارند.

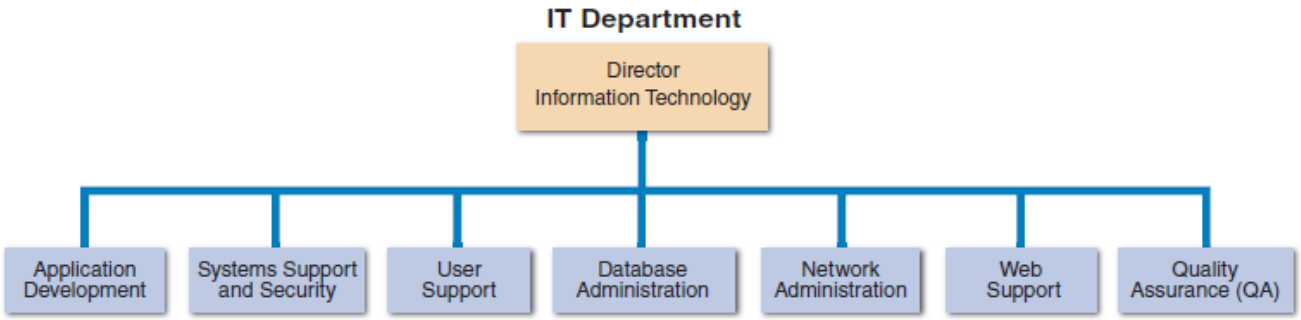
User support: پشتیبانی از کاربران

Data base administration: مدیریت بانک های اطلاعاتی که در سازمان هست مثلا وزارت علوم ممکن است کلی دیتا و دیتابیس و... داشته باشد و یک مدیر دیتابیس دارد.

Network administration: مدیر شبکه

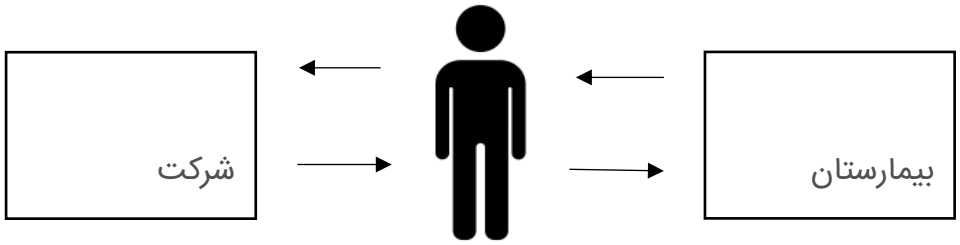
Web support: مبحث اول کسی که وبسایت شرکت یا وبسایت سازمان رو پشتیبانی میکند، به روز میکند، مجموعه کارمندهایی که در حوضه وب کار می کنند را مدیریت میکنند، اینجا مبحث توسعه نیست مبحث نگهداری هست، ما یک وبسایت داریم میخواهیم محتوا برای آن تولید کنیم و روی مبحث های سئو کار کنیم و این ها آن بخشی هستند که وبسایت را تولید میکند بخش Application development است.

Quality assurace (QA): تضمین کیفیت، یعنی این محصولات که ما تولید می کنیم و خدماتی که ارائه میدهیم و مدیریت شبکه و دیتابیس و... این کیفیت لازم را دارند.



The System Analyst: وظیفه تحلیلگر سیستم این است که سیستم های اطلاعاتی که در شرکتها وسازمانها مستقر هستند را بررسی کند، طراحی را انجام دهد، کار توسعه، نصب، ارزیابی و maintains، به نحوی همه کارهایی که مدیر پروژه میتواند متصور شود البته تحلیلگر سیستم به صورت خاص معمولا در قسمت بررسی و آنالیز وطراحی تمرکز داده و بیشترین فعالیت رو انجام میدهد ولی بخش‌های توسعه عمدتا کار برنامه نویس هاست ولی خب چون خیلی از اوقات این تحلیلگرها به عنوان مدیر پروژه هم هستند بنابراین باید با کل فرآیند آشنا باشند.

نقشهایی که یک تحلیلگر سیستم میگیرد در فرآیند این است که به عنوان یک مترجم بین مدیران و برنامه نویسان مثلا بیمارستان و شرکت ما این تحلیلگر کسی است که از شرکت بیرون می آید و با مجموعه بیمارستان سروکار دارد و سعی میکند بشناسد و مدل کند و مدلهایی را به برنامه نویس ها تحویل می‌دهد به نحوی، به عنوان یک مترجم کار می کند.



تفکر سیستمی

تحلیل و طراحی سیستم ها که تفکر سیستمی به نحوی زیر مجموعه ای از تحلیل و طراحی سیستم محسوب می شود چیست؟ و چرا کاربرد دارد؟ کاربرد آن کجاست؟ یک تحلیل اولیه از تحلیل و هم طراحی ارائه می کنیم.

تحلیل

در زبان فارسی خیلی اوقات همراه با نزدیکترین یا پر استفاده ترین تکنیک تجزیه و تحلیل می گوییم.

دلیل بیان تجزیه و تحلیل به خاطر این است که پرکاربرد ترین و آشنا ترین تکنیکی ست که ما در تحلیل سیستم ها از آن استفاده می کنیم.

به طور کلی تجزیه به معنی فرایند شناخت سیستم است که این سیستم میتواند یک سیستم اجتماعی باشد، بایولیژیکی باشد، زیستی،کامپیوتری یا هر چیز دیگری باشد.

پس ما دنبال این هستیم که موضوع یک سیستم، منطق و ساختار آنرا بشناسیم، یعنی نگاه به هر سیستمی رو حداقل در نگاه اول میتوانیم به دو بخش تقسیم کنیم.

logic یا اون منطقی که برای اجرای فرایندهای آن سیستم حاکم است و ساختاری که در سیستم است.

مثلا: دانشگاه به عنوان یک سیستم تشکیل شده از تعدادی ساختمان، دانشکده، محوطه پارکینگ، معاونت های مختلف که هرکدام مدیریت های متفاوتی دارند، مثلا معاونت آموزشی گروه های مختلف دارد،هر کدام از آن گروه ها تعدادی عضو هیئت علمی و تعدادی مدرس حق التدریس و نهایتاً مجموعه ای از دانشجویان که در هر کدام از این گروه ها جمع می شوند.به این اصطلاحاً struct سیستم می گوییم، یعنی ساختاری که میتوانیم سیستم را با آن تصور کنیم.

اما ساختار یک سیستم بیانگر همه ی سیستم نیست logic سیستم خیلی از مواقع بسیار پراهمیت است. منطقی که بر سیستم حاکم است؛ مثلا ما فرایند انتخاب واحد رو چگونه انجام می‌دهیم؟ چه الزاماتی دارد؟ چه مراحلی را باید طی کنیم؟ ورودی ها چه چیزهایی باید باشند؟ بر اساس چه قوانینی این ورودی ها پردازش می شوند؟ چه کسانی مسئول پردازش این اطلاعات هستند؟ و چه خروجی در انتها به ما تحویل داده می شود؟ این ها logic سیستم هستند.

بنابراین وقتی تحلیل سیستم می گوییم و قرار است به سیستم نگاه تحلیلی داشته باشیم (هم منطق و هم ساختار) دسته بندی بود که گفتیم از زوایای مختلف ما باید سیستم را بررسی کنیم.

خیلی اوقات دبستانها مورد خطاب قرار میگیرند به عنوان تحلیلگر مثلا میگویند، فلانی تحلیلگر رویداد های جهان غرب هست یا تحلیلگر اتفاقات آمریکاست و بعد در مورد انتخابات آمریکا از او میپرسیم در واقع انتظار اما از آن تحلیلگر این است که هم ساختار سیستمی که دارد راجع به آن صحبت می کند را به خوبی شناخته باشد و هم منطقی که حاکم بر آن است. بنابراین فرد مدت زیادی رو تخصصی صرف کرده تا قوانین آمریکا، سیستم انتخاباتی و اینکه اصلا هر کدام از این ایالت های مختلف چه نقشی در این ساختار دارند،سیستم های اقتصادی که تاثیر گذارند و... را تا حد زیادی مسلط است و بر اساس تکنیک هایی که میتواند برای تحلیل سیستم استفاده کنه داده هایی رو به ما منتقل می کند.

در کل شناخت یک سیستم از زوایای مختلف

ما برای اینکه بتوانیم از سیستم ها به یک درک مشترک برسیم یعنی کسی که تحلیلگر است فقط از روی یک سری داده ی متنی یا تصویری اطلاعات را جمع آوری نکرده باشد و صرفاً در حافظه ی خودش باشد و بتواند آنها را به اشتراک بگذارد، نیاز به یک زبان مشترک داریم.

مدل سازی

در بسیاری از دیسپلین های علمی از جمله مهندسی، یکی از زبان های مشترک هست. درواقع ما اطلاعاتی رو که از یک سیستم بدست می آوریم، برای اینکه تسلط بیشتری داشته باشیم و هم جزئیات را در سطوح مختلف بتوانیم مورد توجه قرار بدهیم و هم درک خودمان را با دیگران به اشتراک بگذاریم و در خصوص نگاه های متفاوتی که به سیستم می شود کرد این موضوع را به نقد بگذاریم؛ کار مدل سازی را انجام دادیم.

چارچوب های مختلفی در تحلیل سیستم وجود دارد:

چارچوب ذهنی فوق العاده برای تحلیل هر نوع سیستمی که ما آدم ها با آن سر و کار داریم مهم است.

مثلا ۱۳۰ سال پیش یک ریاضیدان فرانسوی به نام کوان کاره موضوعی را مطرح کرد که این موضوع پایه ی علمی شد که ما اخیرا به عنوان علم آشوب از آن یاد می کنیم.

او تلاش کرد یک مسابقه ای را که پادشاه اتریش برای حل مسئله ی ۳ جسم گذاشت تا مسائل دو جسم رو که ریاضیدانها و فیزیکدان ها می توانند حل کنند ولی سالها بود که مسئله ی ۳ جسم را موفق به حل نشده بودن که کوان کاره راه حلی را ارائه داد که البته فهمید اشتباه است و این پایه ی علمی را گذاشت که آنرا با نام کیاس تئوری میشناسیم.

حال رد پای شروع این کیاس رو دانشمندای مختلف دیده بودند ولی چون چارچوب ذهنی آنها اصلا باور نمی کرد که این نویزهایی که در سیستم وجود دارند و ما امروزه به عنوان الگو های کیاس یاد می کنیم. اینها بتوانند پایه ای با علم جدید باشند، تا اینکه نهایتا ادوارد لورنس که یک هواشناس بود و پایه ی ریاضی داشت و بعد هواشناسی خوانده بود و در دانشگاه MIT روی سیستم های هواشناسی کار می کرد به دلیل ذهن بازی که داشت، این الگوها و اتفاقات رو که دیو متوجه شد که احتمالا مسئله ی مهمی وجود دارد و شروع به پرداختن این مسئله ی مهم که چه هست و از کجا آمده و علائم در سیستم چیست؟

مثلا: مثلا ما یک عینک روی چشممان داریم که عینک چارچوب ذهنی ماست، سیستم ها را به صورت مجموعه ای از ساختار ها ببیند. وارد سیستم دانشگاه میشویم از ما بپرسند دانشگاه چگونه جایی ست می‌گوییم؟

دانشگاه از یک مجموعه از روش ها و ارائه ی کارنامه، دریافت گواهی اشتغال به تحصیل و از این دسته نگاه می کنیم و متوجه می شویم که این فرد چارچوب ذهنیش این گونه است که راجع به فرآیندهایی که در سیستم وجود دارد صحبت می کند.

از دیگری میپرسیم میگوید ۴ _۳ تا معاونت دارد و... این هم یک نوع دانشگاه است.

افراد خلاق کسانی هستند که تعصبی روی چارچوب ندارند و اجازه می دهند که ذهنشان با چارچوب های مختلف آشنا شود.

طراحی

به نحوی یک ارائه ی راه حل است یعنی ما متوجه شدیم که سیستم در فاز تحلیل است و درکی نسبت به آن پیدا کردیم حال میخواهیم، راه حلی برای تبدیل سیستم موجود به یک سیستم جدید یا یک سیستم مبتنی بر فناوری ارائه کنیم.

در مثال دانشگاه به جای اینکه دانشگاه را به صورت روالها و پراسیجر های حضوری بخواهیم طی کنیم خیلی از پراسیجرها را تبدیل به پراسیجر های الکترونیکی کنیم. مثلاً در حوضه ی IT خدمات ارائه بدهیم یا ممکن است بگوییم، سیستمی که داریم یک کارخانه است را تحلیل می‌کنیم مشکلات آن کارخانه را ما استخراج کردیم و حال میخواهیم راه حل بدهیم و طبیعتاً هم logic سیستم موجود رو بهبود داده باشد و هم ساختار سیستم رو بهبود داده باشد.

بنابراین در طراحی ما انتظار داریم راه حلی که ما میخواهیم ارائه بدهیم، سیستمی را با کارایی و بهره وری بالاتری پیش روی ما قرار دهند.

در این مرحله ما مدل سازی هم داریم. معمولا مدل سازی این مرحله راه حل است. مدل سازی که برای تحلیل انجام میدادیم مدل سازی سیستمی بود.(business modelry هم گفته می شود.) یعنی مدل سازی کسب و کار یعنی ساختار و خدمات که در مجموعه ایجاد می شود را مدل می‌کنیم ولی در اینجا راه حل را مدل می‌کنیم.

در طراحی هم چارچوب های ذهنی و ساختارهای مختلفی وجود دارد که ما می‌توانیم دوباره نگاه‌مان را ساختاری... فرایندی_شی گرایی ببینیم.

مثلا پارک های علم و فناوری را نمیدانیم ساختارشان چگونه است، معاونت هایشان چیست؟ از چه بخش هایی درست شده؟ اما میدانیم که الان این ایده ی ما در واقع پذیرش شود؟ یک نگاه به طراحی وبسایت برای خارج سیستم پارک علم و فناوری قرار است به آن متصل شوند یک نگاه ساختاری است. این نگاه مستلزم این است که من به عنوان مخاطب یک ساختاری را از سیستم بشناسم بنابراین بتوانیم در وبسایت بچرخیم اما در نگاه فرآیندی ما میگوییم فرآیند هایی که مورد نیاز مخاطب است در صفحه ی اول سایت قرار میدهیم، بنابراین افرادی که با ساختار سیستم آشنا میشوند این فرآیند درک راحت تری برایشان دارند. بنابراین راه حلی که ما ارائه کردیم دو چارچوب مختلف داشت و هر کدام راه حل های ما طراحی های ما هم میتوانند چارچوب های مختلف داشته باشند. شی گرا،داده گراه،عامل گرا و از این اصطلاحتت زیاد هستند.

مهارت های لازم در تحلیل و طراحی سیستم

۱- مهارت های تحلیلی analytical skill: یکی از کلیدی ترین مهارت های تحلیلی است.

۱-۱- تفکر سیستمی system thinking

۱-۲- دانش سازمانی organizational knowledge

۱-۳- شناسایی و تعریف مسئله problem iolenfication

۱-۴- تحلیل و حل مسئله problem analysing and soluing

یک فرد اگر میخواهد مهارت تحلیلی خود را افزایش بدهد باید این مهارت را کسب کند.

۱-۱- تفکر سیستمی: یعنی ما پدیده ها را به شکل سیستم ببینیم. ما اگر قرار است راجع به یک موضوع فکر کنیم، آن موضوع را در یک context سیستم بررسی کنیم.

سیستم به مجموعه ای از اعضا که در تعامل با هم باشند میگوییم که مجموعه ای از اهداف برای سیستم تعریف شدند دنبال میکنند و این سیستم از اجزای متعددی تشکیل شده و یک جزء یک کاری را انجام بدهد را واژه ی سیستم به آن اطلاق نکنیم.

اجزا در یک سیستم با هم در تعامل هستند. ما هیچ جزئی را در سیستم نمیتوانیم پیدا کنیم که ایزوله یک گوشه ای افتاده باشد. این ها همه خطوط راهنما هستند برای اینکه وقتی ما سیستم را تحلیل می کنیم و مدلش کردیم، متوجه شدیم که یک قسمتی داریم که یک گوشه افتاده با هیچ قسمت از اجزای سیستم تعامل ندارد. یعنی اشکالی در تحلیل ما بوده یک ارتباطی بوده که ما گم کردیم. اصلاً چنین چیزی در هیچ سیستمی وجود ندارد چون اگه قرار است وجود داشته باشد پس چرا شکل گرفته؟

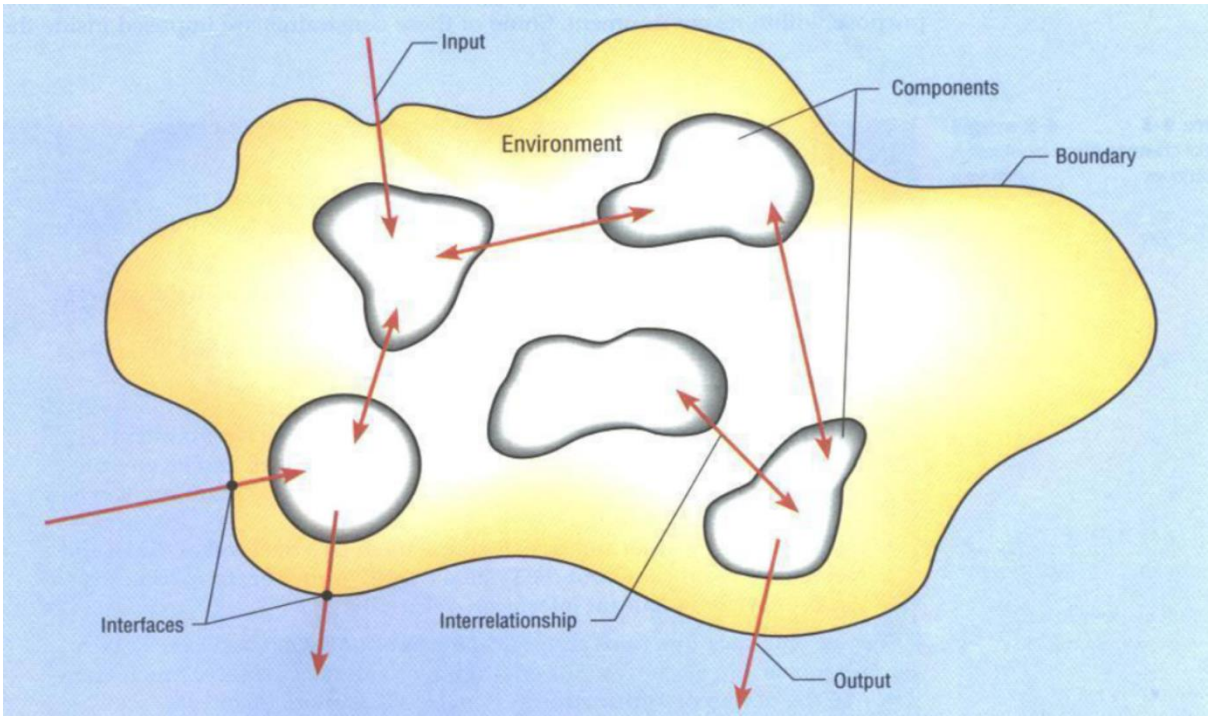
برای سیستم ها میتوان رفتار تعیین کرد و با حذف هر یک از اجزای سیستم رفتار کلی سیستم تغییر میکند موضوعی که ما در سیستم های پیچیده تغییرش میدهیم و می گوییم با حذف هر کدام از اجزای سیستم پاریداری سیستم را بهم نمیریزد، این یکی از ویژگی های یک سیستم پیچیده است.

معمولا برای سیستم ها یک مرز تعریف می شود و هدف در نظر میگیریم اما چیزی که خیلی مهم است این است که این تعامل بین اجزا بسیار کلیدی است. اگر این تعامل و تاثیر گذاری و تاثیر پذیری رو از اجزای سیستم بگیریم؛ فقط مجموعه ای از اعضا داریم سیستم نداریم، یعنی کلید واژه ی سیستم تعامل بین اجزا وجود دارد.

وقتی ما میخوای تفکر سیستمی رو بفهمیم اول باید خود سیستم رو متوجه شویم که سیستم از چهار جزء تشکیل شده است:

۱. اجزا component این اجزا ممکن است افراد باشند _یک سری ساختارها باشند مثل دپارتمان یک دانشگاه که میتواند بخش فیزیکی یک پراسیجر باشد که مجموعه ای از پراسیجرها یک کار را انجام میدهند (بسته به عینک ما دارد)
۲. ارتباط بین اجزا interrelated component
۳. مرز و محدوده سیستم boundry
۴. هدف purpose که برای هر سیستم باید هدف یا اهدافی تعریف شده، مشخص باشد.
۵. محیط enviroment محیطی که اجزا در آن با هم تعامل دارند.
۶. واسط ها interfaces محل هایی که این سیستم با سیستم های دیگر از آنها ورودی میگیرد و خروجی میدهد.
۷. ورودی input
۸. خروجی output
۹. محدودیت ها و شرایط constraints محدودیت ها و شرایطی که بر اون سیستم حاکم است، هرسیستمی یک سری قوانین و آیین نامه ها و شرایط و ضوابطی دارد.

اگر ما میخواهیم تفکر سیستمی داشته باشیم عینک تفکر سیستمی را که بزنیم باید هر سیستمی رو بلافاصله با این ۹ قسمتش ببینیم.



ارتباط ها اگر نباشند سیستم معنی ندارد.

از بیرون سیستم یک سری ورودی وارد سیستم ما می شود این interface جایی هستند که ما تعریف کردیم که در سیسستم میتوانند ورودی ها وارد سیستم شوند. هر جایی از سیستم نمیتوانیم ورودی بپذیریم.

قسمتی دارد که سیستم خروجی به بیرون میدهد، سیستم به هر جایی خروجی نمی دهد از محل هایی که تعریف شده فقط خروجی میدهد، به همین خاطر این interface ها مهم هستند (بحث های security یا امنیت و بحث های coding و decoding و بحث های safely)

هر سیستمی را که میخواهیم تحلیل کنیم بعضی افراد را دیدید که وقتی از آنها سوالی پرسیده می شود مثلا راجع به انتخابات آمریکا صحبت کنند در مورد مشکلاتی که آمریکا در لیبی داشته صحبت می کنند و بعد سیستم لیبی را تحلیل کنند. این یعنی پایبندی به مرز اتفاق نیافتاده البته که ممکن است یک سیستم با سیستم های دیگر در تعامل باشد و تشخیص اینکه ما متوجه شویم این تعامل هست نه اینکه مسائل لیبی داخل مسائل آمریکا باشد.

یا مثلا فرض کنید تو سیستم دانشگاه ما درک کنیم که این دانشگاه خوابگاه های خصوصی دارد بنابراین مدیریت خوابگاه های خصوصی جزء تحلیل سیستم ما نیست بلکه خودش یک سیستم دیگر است که با سیستم ما که سیستم دانشگاه باشه از طریق همین interface در تعامل است. تشخیص مرز سیستم این کمک را به ما میکند که پروژه را بتوانیم جمع کنیم اگر نه که هیچ پروژه ای قابل جمع شدن نیست و احتمالاً ما همینطوری پروژه رو بزرگ و بزرگتر خواهیم کرد.

اولین و مهمترین که همیشه همراه تحلیل سیستم هست ←تجزیه decomposition شکستن سیستم به اجزا مهم ترین تکنیکی است که ذهن آدم ها برای اینکه بتواند سیستم ها را بشناسد خیلی خوب درک میکند. ما برای اینکه یه سیستم بزرگ را بفهمیم به اجزائش تقسیم می کنیم یا breaking down یعنی شکستن یک سیستم یا یک کل به مجموعه ای از اجزا.

اینکه composition فرایند شکستن یک سیستم به اجزای تشکیل دهنده است. نکته ای که توی این شکستن است این است که اجزا باید قابل مدیریت باشند و ما وقتی میشکنیم در هر زمان روی یک بخش تمرکز میکنیم، مسائل آن بخش را حل میکنیم بعد سراغ بخش های بعدی میرویم. اصطلاحاً وقتی همه ی اجزا را حل کردیم انگار کل سیستم را حل کردیم (همیشه این نوع فکر کردن درست نیست) اما روشی است که کار میکند.

مثلا ما یک کیس کامپیوتر رو باز می کنیم. میگوییم هارد یک حافظه ی جانبی دارد، حافظه‌ی اصلی یا رم دارد، cpu چه کار میکند حتی خود cpu را به ۳ بخش Alu و کنترلر میشکنیم و آن را تحلیل میکنیم. می گوییم مادربرد هست.... ما با شکستن به هر جزء تسلط پیدا میکنیم و جلو میرویم اما در مورد همه ی سیستم ها این روش درست نیست.

تجزیه و تحلیل خیلی ذهن ما آدمها باهاش آشناست سابقه ی زیادی هم دارد که ربطی به مسائل علمی ندارد خیلی معروف است که در جنگ هایی که از زمان سانترا در چین شروع شد تا جنگ های اسلام در کل جنگهای مختلف در تمدن های مختلف دیدیم یکی از مهم ترین تکنیک ها رو به نام divid and corquer(تقسیم کن و پیروز شو یا اختلاف بیانداز و برنده باش)که معمولا نیرو ها را تقسیم می کردند یا ارتش مقابل را سعی میکردند از وسط یک تعداد نیرو بفرستند و به دو بخش شکسته شود، دو بخش را به چند بخش و به همین ترتیب نهایتا در جنگ پیروز میشوند.

این یکی از اصلی ترین روش هایی بود که مسائل را با آن حل میکردند.

در فیزیک به همین مرحله ی decomposition می گویند تحلیل گرا یا reductionism، نگرشی که دکارت و نیوتون پایه گذاری کردند. یعنی می گفتند که ما (دکارت تو کتابش)روی آن عکس اردکی را کشیده بودند و می گوید که این اردک را میتوانیم درک کنیم و به اجزاش بشکنیم و بفهمیم که چه کار میکند و به سیستم بیولوژیکی و زیستی اش کاملا مسلط شویم.

نکته ی مهم این است که ما متوجه شویم سیستمی که میشکنیم، این اجزا، واقعا چه هستند؟ می توانند ساختار باشند، می توانند فرایند ها باشند، میتواند موجودیت های اشیا و اینکه ما یک سیستم را به چیزی میشکنیم خودش به این بستگی دارد که چارچوب ذهنی ما چیست! عینکمان چیست! به همین خاطر دو نفری که میخواهند سیستم را تحلیل کنن به دو شیوه ی مختلف این کار را برای ما انجام دهند ولی تا حدی هر دو به اندازه ی کافی به سیستم مسلط شوند.

۱. یکی دیگر از اصطلاحاتی که در سیستم و تفکر سیستمی زیاد استفاده می شود کلمه ی ماژولاریتی modularity است.

فرآیند تقسیم سیستم به ماژول هایی با اندازه تقریبا یکسان ماژول ها طراحی سیستم را برای ما ساده می کنند.

۲. جفت شدگی coupling: زیر سیستم هایی که به یکدیگر وابسته هستند با هم جفت می‌شوند و اتفاقا همین سیستم است که باعث می شود یک کل را به اجزا بشکنیم و دوباره اجزا را کنار هم بچینیم همان کاری که به آن سنتز می گویند و بتوانیم دوباره کل را بسازیم.

۳. انسجام cohesion:میزانی که یک زیر سیستم عملکرد واحد را انجام میدهد. یک زیر سیستم خودش شامل یک مجموعه ای از اعضا است. این زیر سیستم چقدر عملکرد واحد را انجام میدهد.

۴. توصیف منطق سیستم logical system description:یعنی منطق سیستم چیست! مثلا در یک سیستم بانکی کارمندانی که در بانک نشستند مثل رییس، معاون و تحویلدارها. اینها همه تفسیر فیزیکی سیستم می شوند. اینکه هر کارمند کجاست؟ چه کامپیوتری دارد؟ چه سطح دسترسی دارد ولی اینکه مثلا با چه شرایطی به ما وام میدهند بر اساس چه قوانینی ما میتوانیم تسهیلات را دریافت کنیم، میزان سود سپرده بر حسب چه قانونی محاسبه می شود. اینها همه به منطقی که در سیستم وجود داره بر میگردد و اصلا سرویس هایی که یک سیستم میتواند به ما ارائه کند را در غالب منطق سیستم ببینیم.

در توصیف فیزیکی ساختار فیزیکی را می بینیم، این اصطلاحات متفاوت در سیستم وجود دارد که خیلی از مواقع مورد غفلت واقع می شود. این زاویه ی دید متفاوت به سیستم (این زاویه ی دید منظور زاویه دید فیزیکی نیست)است.

مثلا: بانک فقط از دید مشتری به سیستم نگاه نکنید یعنی نگویید که بانک یک مکانیزمی هست که در آن میتوان افتتاح حساب کرد و سود سپرده گرفت و تسهیلات گرفت و انتقال وجه و... انجام داد.

- اما از دیدگاه مدیر سیستم بانکی: بانک خدمات دیگری مثل بحث مرخصی کارمندان شعبه را مدیریت می کند، اضافه کاری کارمندان را پرداخت میکند، گزارش های مختلف از سیستم میگیرد که مثلا در امروز چه میزان آورده ی مالی به شعبه داشتیم و چقدر پول افراد از اینجا برداشت کردند چه نوع تراکنش هایی بیشتر از بقیه انجام شد. اینها از دید مشتری اصلا وجود ندارد.
- از دید سرپرست شعبه های یک استان فرق میکند: اینکه چه شعبه هایی در سطح شهر زیر مجموعه ی این فرد مدیریت میشوند. بعضی از شعبه ها درجه بندی می شوند مثلا شعبه ی درجه یک و درجه دو و سه هر کدوم از این شعبه ها چند نیرو دارند و ماهانه چقدر اضافه کار پرداخت می شود؟ گزارش های مالی که داریم میگیریم کدام شعبه عملکرد بهتری داشته است؟

بنابراین در تحلیل یک سیستم لازم است که ما رول ها و نقش های مختلف را بگیریم.

انتزاع (Abstraction): پیچیدگی ها را در همان ابتدای کار همه را یکجا جمع نکنیم، اگر بخواهیم چنین کاری کنیم ذهن ما از نظر مطالعات شناختی ثابت شده که ذهن ما قادر نیست پیچیدگی های زیاد را هم زمان با هم درک کند. این پیچیدگی ها را باید یا در سلسله مراتب یعنی تیکه تیکه، اصلا همان decomposition کمک هست یا همان ساختاری که گفتیم رییس دانشگاه یعنی سطح بالاتر بعد معاون بعد مدیریت بعد گروه ها و....

انتزاع هم تقریبا یه چنین مفهومی دارد می گوید ما وقتی میخواهیم سیستم را توصیف کنیم در سطح انتزاع اول باید ببینیم در سطح دوم این خدمات آموزشی در چه سطوحی مثلا گروه مهندسی کامپیوتر، شیمی و مکانیک و... بیایم در هر کدام از این گروه ها بگوییم که دقیقا چه کسانی هستند و چه خدماتی را ارائه میدهند و شروع به بحث های تحلیل و مدل سازی می کنیم. ما به اینکه تمام پیچیدگی ها را در سطح اول نپردازیم انتزاع می گوییم.

۲_۱_ دانش سازمانی organizational knowledge

دانش سازمانی برای مهارت تحلیل سیستم به شدت مورد نیاز است. این سازمان را به عنوان یک نماد از سیستم هایی که قرار است آنها را تحلیل کنیم آوردیم یعنی نسبت به ساختارش دانش داشته باشیم و فرایندهایش چیست؟

دانش سازمانی به این میپردازد که ما (کسانی که در حوزه ی بیزینس فعالیت دارند organizational به معنای سازمان است و کسانی که در حوزه ی تحلیل سیاسی کار میکنند ممکن است سازمانشان یک کشور باشد، یک وزارت خانه باشد برای حوزه ی نرم افزار، کسانی که دارند یک سری خدمات را ارائه میدهند بنابراین سازمان میتواند هر چه باشد.)

نکته ای که وجود دارد این است که بفهمیم سازمان ها چگونه کار میکنند یعنی درک اولیه داشته باشیم و این درک اولیه کمک میکند وقتی تحلیل می کنیم. اصطلاحا روش های تحلیل سیستم می گوییم را به کار میگیریم که با افراد سیستم صحبت میکنیم، اطلاعاتشان را به دست می آوریم، داکیومنت ها را می گیریم میخوانیم فرم هایشان را بررسی میکنیم به ما کمک میکند که فرایند ها را بفهمیم.

دانش عملکردها و رویه های تعریف شده برای سازمان، اینکه بفهمیم یک سازمان چطور عملکردها و رویه هایش را انجام میدهد. تصور کنید اگر یک فرد نوجوان را در یک کارخانه عملکرد سیستم رو بخواهد متوجه شود، اصلا ممکن است درکی از اینکه مدیر عامل کیست، اعضای هیئت مدیره چه کسانی هستند، چگونه سلسله مراتبی وجود دارد، اصلا یک کارخانه معمولا چه خدماتی را ارائه میدهد اینها را نداشته باشه و هرچقدر با افراد آنجا صحبت کنند ممکن است نتواند به سادگی یک درکی از سیستم بدست بیاورد که نهایتا منجر به تحلیل سیستم شود.

بنابراین دانش سازمانی کمک میکند و ما هرچقدر بزرگتر میشویم و با بخش های مختلف اجتماع سر و کار داریم و خدمت میگیریم (البته اگر نگاه کنجکاوی داشته باشیم)دانش آن سازمان را بهتر می فهمیم و درک میکنیم.

ممکن است بگوییم که این دانش سازمانی بدون اینکه ما تحلیل را شروع کنیم در واقع همه جانبه نباشد کاملا درست است، چون سیاست های داخلی، آیین نامه ها و... همه وقتی به دست میاد که ما فرایند تحلیل سیستم را شروع کنیم و مورد مطالعه قرار بدهیم، اینکه محیط رقابتی و نظارتی سازمان را بتوانیم درک کنیم، استراتژی ها را بتوانیم درک کنیم، دانش سازمانی که هر کدام از افراد یک فراخور اینکه چقدر با یک سیستم یا سازمانی ارتباط داشتند یک درک اولیه از آن دارند هم خوب است و هم اشکالی دارد. احتمالا این را شنیدید که خلاقیت در کودکان به مراتب بیشتر از بزرگتر هاست مثلا یه شکلی بی ربط بکشید و به یک بزرگتر نشان بدهید احتمالا چند تفسیر بگوید ولی به کودکان که نشون بدهید ممکن است ده ها تفسیر نسبت بدهند چون ذهنشون هنوز چهارچوب بندی نشده است.

یک کسی که در دانشگاه درس خوانده وقتی از دانشگاه فارق التحصیل می شود می گویند شما سیستم دانشگاه را تحلیل کن، فکر میکند که دانش سازمانی دارد ولی اون نگاهی که این فرد در مدت که دانشگاه بوده، بیشتر از نگاه دانشجویی است. بنابراین فکر میکنه تمرکز اصلی سیستم همین است و متمرکز می شود روی رویه ها و قوانینی که معطوف به دانشجو هست.

شناسایی وتعریف مسئله

این موضوع مهم است و بعضی افراد که توانمندی خاصی داشتند روی این بخش توانستند بهره های زیادی ببرند.

مثال: ممکن است یک نفر در ایستگاه اتوبوس بایستاد و اتوبوس دیر بیاید زود میرود خیلی زمان بندی ندارد و خیلی این را به عنوان یک مسئله نمیبیند آن فرد یعنی فکر میکند که یعنی همین مدل سیستم را باید کار کند ولی کسی که چند تا سیستم درست را دیده باشد و متوجه شود که وقت ارزش دارد و افرادی که در ایستگاه می آیند باید مطلع باشند که کدام سرویس چه ساعتی به اینجا خواهد رسید کاری که در مترو انجام شده. وقتی این را دیده باشند می فهمند که اشکالی در سیستم وجود دارد و اشکال این است که وضع موجود با وضع مطلوب متفاوت است.

حال تصور کنید که وضع مطلوب (ما الان داریم راجع به موضوعی فکر میکنیم که زمان هدر رفتن است، اشکال هست)ولی ممکن است چیز دیگری اشکال داشته باشد که ما نمیبینیم یعنی ما هنوز وضع مطلوب را درست نمیشناسیم.

داستان اعرابی و کوزه آب:

یک اعرابی که تو صحرا زندگی میکرد یک مدت که اینور آنور جا به جا میشد میدیده که چند تا درخت هست که وسطش یک جوی آب هست و میگه این دیگه خود بهشته! تفسیر دیگه ای از بهشت نداشته چون هر چی میدیده بیابان بوده ولی از اون آب میخوره و میبینه به چه آبی و چه خوبه و درخت و اینا و میگه این آب بهشته کوزه رو پر میکنه و راهی بغداد میشه که آب رو به خلیفه نشون بده. میره و میگه که در کاخ که از بهشت اب آوردم میره داخل و اون اب رو میخوره و میبینه که اصن اون اب گندیده هست و وقتی میپرسه اینو از کجا اوردی و تعریف میکنه متوجه میشه که این دیدگاهش نسبت به غایت یک زیبایی و چیزی همینه که داره تعریف میکنه و اینیه که دیده و خوب ازش تشکر میکنه و به نگهبانا میگه نزارین دجله و فرات رو ببینه خجالت زده میشه.

مدل ذهنی open minded))و ابزارهای شبیه سازی کمک میکنند که ما وضع مطلوب و موجود را بهتر بشناسیم. شناسایی وضع موجود سیستم همیشه کار ساده ای نیست ولی معمولا جوری از آن حرف میزنند که انگار راحت است!

تحلیل و طراحی سیستم معمولا در چهار گام قابل بررسی است.

اطلاعات (intelligent): همه ی اطلاعات مرتبط جمع آوری میگردد.

طراحی: راه حل های مختلف را بررسی و فرمول بندی مدل می کنیم.

انتخاب: بهترین راه حل از نظر معیار هایی که داریم، هزینه زمان.... انتخاب می کنیم.

اجرا: راه حلی که از همه بهتر بوده است از هر نظر عملیاتی می کنیم و وارسی می کنیم. (اطمینان حاصل می کنیم اون چیزی که میخواستیم شده؟)

شناسایی و تعریف مسئله problem identification بعضی یک چشم خاصی دارند که میتوانند مسائل را خوب ببینند البته که با تمرین به دست میاد ولی در حوزه ی نرم افزار یک سری جوان را میشناسیم که توانستند بیزینس های خیلی بزرگ راه بیاندازند و سودهای بزرگی را بدست بیاورند.

یک بخش زیادی (بیش از ۵۰ درصد)به آدم ها و توانایی تشخیص و شناسایی و تعریف مسئله را داشتند، بر میگردد. یعنی فهمیدن یک مشکلی وجود دارد که این مشکل اگر توسط محصول حل شد یا دستگاهی که میسازند مردم از این دستگاه و محصول استقبال خواهند کرد و پولدار میشوند.

شناخت مسئله نیمی از راه حل هست. در بسیاری از مسائلی که در سطح کسب و کار با آنها سر و کار دارین شناسایی مسئله تقریبا تمام راه حل است یعنی خیلی اوقات راه حل در شناخت مسئله نهفته است اینکه ما چقدر خوب میتوانیم مسئله را شناسایی کنیم همین یعنی **تحلیل**. میخواهیم مسئله را شناسایی کنیم و به اجزا از زاویه های مختلف نگاه کنیم عینک های مختلف بزنیم، ارتباط بین اجزا را ببینیم اینها همه تحلیل مسئله است.

۲. مهارتهای فنی technical skills

نیاز یک تحلیلگر است. یکی از صنعت های جدی که خیلی از تحلیلگران ما به ویژه در تحلیل سیستم های اجتماعی دارند این است که مهارت های فنی شان کم است.

مهارتهای فنی یعنی مهارتهایی که ما دانش فنی نسبت به تکنولوژی هایی که آمده داریم، امکاناتی که این تکنولوژی ها معرفی میکنند ابزار های مختلف چه مزیت ها و چه معایبی دارند، هزینه ی هرکدام چقدر هست. اینا همه به مهارتهای فنی بر میگردد.

اینکه فناوری هایی که قابل استفاده توانستیم یا راه حل ما هستند و کداماشان را میتوانیم پیشنهاد بدهیم. هر کدام از این فناوری ها مزایا و معایبشون چیست؟ و نهایتا کدام را باید انتخاب کنیم؟ تصور کنید یک سیستم موجود را تقریبا شناختیم اما میخواهیم یک سیستم بهتر از آن طراحی کنیم یا سیستم را ارتقا بدهیم، تکنولوژی هایی که معرفی میکنیم را میدانیم که در هر بخشی کلی تکنولوژی وجود دارد، که کدام یکی از این تکنولوژی ها را باید انتخاب کنیم؟ مزایا و معایبشون چیست؟ به این بر میگردد که چقدر دانش فنی داریم.

اگر میخواهیم دانش فنی خود را بالا ببریم باید دائما تحولات فناوری رو پایش کنیم، یک تحلیلگر باید دائما این روز ها را شناسایی کند. (روزهای فناوری رو شناسایی کند.)

تحلیل پتنت ها کمک میکند که متوجه شویم چه فناوری هایی ایجاد می شود چون معمولا وقتی یک فناوری جدید یا یک فناوری تغییر یافته بهبود یافته ای شکل میگیرد بلافاصله صاحب این اثر پتنت را ثبت میکند به خاطر اینکه از فکر و ایده اش محافظت کند و اتفاقا خیلی با جزئیات بر خلاف مقالان علمی آنها را منتشر میکند و رایگان هم هستند. این تحلیل سطح پتنت برای شرکت هایی مفید است که در حوزه ی فناوری خیلی پیشرو هستند، به درد یک تحلیلگر صفر که میخواهد یک سری سیستم ها را برنامه ریزی و تحلیل کند همان فناوری هایی که در جریان روند هستند قرار گرفتند یا پویش محیطی هایی که انجام میدهد کفایت میکند.

لازم است که در حوزه ی تحولات فناوری به روز باشیم. نشریات مقالات تخصصی را دائما مطالعه کنیم در انجمن های علمی و حرفه ای عضو باشیم و تحولات آنها و کارگاه ها و همایش هایشان را رصد کنیم، دوره های آموزشی مختلف و در همایش ها و نشست های مختلف شرکت کنیم.

۳. مهارت های مدیریتی

۳-۱ مدیریت منابعresource management: منبع شامل افراد وداده ها و اعتبارات مالی و منابع فیزیکی می شود ما چگونه اینها را به کار بگیریم تا راه حلمان که همان مدل طراحی هست یک راه حل کم اشکال باشد و از سیستم قبلی بهتر باشد.

۳-۲- مدیریت پروژه project managementبرای انجام یک پروژه گام های اولیه که معمولا در خیلی از پروژه ها هم اینگونه تعریف می شود ما آن سیستم را باید تحلیل کنیم بعد مدل را طراحی کنیم بعد آن را اجرا کنیم و بعد مورد آزمون و وارسی قرار بدهیم. این فرایند که طی می شود را مدیریت پروژه میگوییم و یک تحلیلگر باید روی آن تسلط داشته باشد و بتواند کنترل کند که از زمان پروژه فراتر نرود یا از منابع مالی که پیشبینی شده برای پروژه فراتر نرود.

۳-۳- مدیریت مخاطرات Risk management ریسک ها را پیشبینی کند و به این مسئله برمیگردد که ما چقدر توانمند هستیم، در پیشبینی عواملی که میتواند پروژه را مخاطره کند وقتی این پیشبینی را انجام دهیم میتوانیم برای مخاطرات برنامه بریزیم. از بعضی از آنها را می شود اجتناب کرد، از برخی هم که اجتناب ناپذیرند می شود هزینه ی تخریب یا مخاطره را به حداقل رساند.

۳-۴- مدیریت تغییرات change management معمولا در سیستم هایی که درگیر طراحی آنها هستیم نیازمندی ها تغییر میکنند و ما این تغییر ها را میتوانیم مدیریت کنیم. این مدیریت تغییرات درون سیستم تحلیل و طراحی است و یک بخشی هم این است که چگونه سیستم موجود را به سیستم جدید انتقال بدهیم.

۴. مهارت های شخصیتی

یک تحلیلگر در بخش های مهارتی تحلیلی نیاز دارد که اطلاعات جمع کند، از زاویه های مشتری، کارمند، مدیر گروه سیستم را ببیند پس باید با آن افرادی که در سیستم هستند ارتباط بگیرد و با آنها صحبت کند و مصاحبه داشته باشد. مصاحبه ی فردی و مصاحبه ی گروهی آنها را ترغیب کند که اطلاعات بدهند اینها به مهارتهای ارتباطی یا communication skills برمیگردد که در چه سطحی هستند و این دسته مهارتها، مهارت های کلیدی هستند یعنی در موفقیت خیلی از افراد تاثیر گزار بوده است.

- مهارت های کار کردن با تیم و به صورت فردی: همیشه این دو مهارت، گاهی اوقات میگوییم که ما ایرانی ها بلد نیستیم تیم ورک کار کنیم فکر میکنیم ارزش تیم ورکینگ است، فقط در صورتی که خیلی از مواقع اتفاقا ما به مهارت کار به صورت فردی هم نیاز داریم یعنی بتوانیم تکی از عهده یک سری کارها بربیایم، یک جایی بتوانیم با تیم کار کنیم.
- یک تحلیلگر خوب کسی است که نه فقط بتواند با تیم کار کند بلکه در مواقع لازم بتواند فردی هم کار کند.

- درسیستم های پیچیده که بعدا توضیح داده می شود که خیلی از اوقات فرد تحلیلگر باید فرصت به کسانی که به ما اطلاعات میدهند، بدهند که به عنوان مشارکت کنندگان سیستم یا کاربران سیستم که میخواهند اطلاعات بدهند. حرفاشان را بزنند نقش ما در این نوع مصاحبه های گروهی بیشتر facilitatg groups یعنی تسهیل ساز گروه ها دارد.
- اینکه انتظارات کاربران از سیستم را مدیریت کنید، فکر نکنند که قرار است همه‌ی مشکلات توسط یک پروژه حل بشود.

این ها به اینکه ما چقدر میتوانیم انتظارات افراد را مدیریت کنیم بر میگردد که بلوف زدن اصطلاحیه که اینجا گاهی اوقات مخاطره ایجاد کند یعنی اینقدر بلوف زدیم که انتظاراتی را در مخاطب ایجاد کرده که فکر میکند ما و پروژه ی ما و درک ما از سیستم کلی مسائل داخل سیستم رو حل میکند.

فصل ۲

اهداف فصل: بخشی از این اهداف به نگاه سازمانی که ما باید به یک ساختار و سازمان مورد بررسی داشته باشیم، برمیگردد.

- با فرایند برنامه ریزی راهبردی آشنا می شویم.
- با تحلیل swot آشنا می شویم و ابزار هایی که برای این کار استفاده می شوند.
- با مفهوم business case یا همان مورد کسب و کار آشنا می شویم.
- دلایلی که پیش میاد که ما بخواهیم سیستم ها را توسعه بدهیم و مسائل را حل کنیم.
- فاکتور هایی که روی پروژه تاثیرگذار خواهند بود (فاکتور های خارجی و داخلی)
- فرایند درخواست ارتقا سیستم یا توسعه سیستم به چه شکلی است.
- مفهوم مهم امکان سنجی یا feasibility study بر اساس اطلاعاتی که از مشتریان، محیط کسب و کار، رقبا و قسمت های دیگر استخراج می کنیم. باید مطمئن باشیم که طرح ما از نظر عملیاتی، تکنیکی، زمانبندی و مالی قابل انجام است.

برنامه ریزی راهبردی

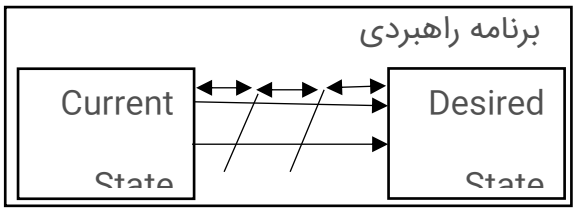
Problem یا مسئله، گپ بین حالت موجود و حالت مطلوب است، یعنی سازمان ما در حال حاضر در حالت موجود است ولی خیلی دوست داریم به حالت مطلوب برسیم.

Desired

Current

مثلا شهرداری همدان در حال حاضر تعداد اندکی از خدماتش را به حالت الکترونیکی ارائه می کند و بخش زیادی از خدمات حضوری است و فرایند انجام کارها ترتیبی مشخص دارد. ایده آل این است که تمام فرایند های شهرداری و سرویس هایی که ارائه می دهند از طریق اینترنت برای مردم قابل دسترس باشد. بعضی فرایند ها هم اصلاح شوند و باعث شوند که انجام کارها در زمان کوتاهتری انجام شود.یک سری دیتا که شهرداری تولید می کند و می تواند در اختیار استارت اپ های مختلف قرار دهد که بتوانند اپلیکیشن های متفاوتی بنویسند و به همین ترتیب ما بتوانیم راجب وضع ایده آل سیستم شهرداری صحبت کنیم.

برنامه ریزی استراتژیک به این می پردازد که چگونه از وضع موجود به وضع مطلوب، بر اساس چه برنامه‌هایی برسیم. این ها همان برنامه های راهبردی هستند، برنامه ی راهبردی میتواند ۵ ساله باشد. به طور مثال خود برنامه راهبردی به ۴ برنامه کوتاه مدت شکسته می شود و برای هر کدام ازاین برنامه‌های کوتاه مدت یک مجموعه برنامه اقدام یا action plan می نویسیم که باید در سال چه کارهایی انجام دهیم، سال دوم چه کارهایی انجام دهیم و به همین ترتیب تا سال اخرو در نهایت هم باید بررسی کنیم که به ان وضع مطلوب رسیده ایم یا نه.



واقعیت این است که ما این داستان را خیلی ساده در نظر گرفتیم، احتمالا نقد های زیادی راجع به برنامه های راهبردی میخوانیم. خیلی از مواقع به گونه ای صحبت می کنیم که گویا به این وضعیت مطلوب سیستم حالت ایستا را دارد و ما فقط کافیه بر اساس یک سری action plan به وضعیت مطلوب برسیم، ولی واقعیت این است که در عمل بسیار پیچیده است و کاملا محیط دینامیک است.

عوامل مختلفی برنامه ما را تحت تاثیر قرار می دهد. در نهایت این وضعیت مطلوب یا target که تعریف کردیم، یک target ایستا نیست و به هر حال تحلیل محیط،فرایند خیلی پیچیده ای است که به راحتی می شود برنامه ریزی را به این ترتیب هدایت کرد.

کلیت ماجرای برنامه ریزی راهبردی شامل فرایند تعریف اهداف بلند مدت سازمان، برنامه های راهبردی بلند مدت سازمان، منابع شان و این را معمولا از بیان vision یا بیان ماموریت سازمان شروع می‌کنیم.

در بیان ماموریت ما به vision یا چشم انداز سازمان، اهدافی که دنبال می کند، ارزش‌هایی که داخل سازمان جاری است، عوامل کلیدی موفقیت سازمان، اهدافی که الویت دارند میپردازیم. برای نوشتن برنامه راهبردی، تحلیل swot یکی از تکنیک های مرسوم است. Swot به نقاط قوت و ضعف سازمان و فرصت هایی که در بیرون مهیا است، میپردازد.مواردی که ما را تهدید می کند را شناسایی کنیم و براساس آن تحلیل swot را انجام دهیم.

نقاط قوت و ضعف، فرصت ها و تهدید ها را (threats, opportunities, strength, weaknesses) می نویسیم، بررسی می کنیم چگونه می توانیم نقاط قوت را تقویت کنیم و نقاط ضعف را کاهش دهیم و از فرصت ها استفاده کنیم و تهدید ها را بتوانیم مهار و کنترل کنیم. به طور کلی تعدادی استراتژی می نویسیم.

	Strengths
weakn	esses

مثال: برای شرکتی قوی بودن در حوزه وب و داشتن نیرو های قوی یک نقطه قوت است. و استفاده از سیستم های قدیمی legacy system))نقطه ضعف است. فرصت موقعیت های خوبی دارد این شرکت تا بتوانند کارها را توسعه دهند.در بیرون این فرصت ها مهیاست و میتوانند استفاده کنند.رقبای وب به شکل حرفه ای مشغول

کار هستند و این یک رقابت شدیدی را در حوزه وب شکل می دهد بنابراین یک تهدید برای این شرکت به حساب می اید، باید بتوانند حوزه اثر گذاری خودشون را زیاد کنند.

شرکتی در حوزه پتنت کار میکند:

نقطه قوت: پتینت های ما عمدتا فناوری های با ارزش را پوشش می دهند.

نقطه ضعف: پتینت های ما طول عمر کوتاهی دارند. (این ویژگی ها می تواند یکی باشد یا چندتا، می‌توانیم برحسب اطلاعاتی که از افراد یا خبرگان داخل سیستم گرفتیم دسته بندی و امتیازبندی کنیم و نهایتا بر اساس این اطلاعات که به دست میاوریم نقاط ضعف، قوت، تهدیدها و فرصت ها را می‌شناسیم.)

این مسئله تمرکزش روی سازمان یا organization است ولی میتواند فردی هم باشد. ما می توانیم برنامه ریزی برای کسب و کار و زندگی را کاملا برحسب swot انجام دهیم.

در برنامه ریزی استراتژیک نقش گروه IT سازمان این است که بتواند business strategy ها را ساپورت کند و نیازهای عملیاتی را پشتیبانی و حمایت کند. محدوده پروژه هایی که می خواهیم انجام دهیم مشخص و دقیق تعریف شوند. در فصل اول گفتیم یکی از ویژگی های سیستم مرز است و وقتی در حال انجام یک پروژه هستیم عملا این پروژه قرار است روی سیستم سوار شود.

شناخت این مرز یا boundary برای ما خیلی مهم است و گروهIT میتواند دراین کار به ما کمک کنند و یا حتی برای ما برنامه استراتژیک بنویسند و یا هر کار دیگری در حوزه اطلاعات و توسعه اطلاعات را انجام دهند.

تا حد ممکن اهداف باید واقع گرایانه باشند نه خیلی ایده ال و نه خیلی سطح پایین باشند، توضیحات کافی در مورد اهداف داشته باشیم. فرضیات،محدودیت ها، فاکتور ها و سایر ورودی ها دقیق باشند.

طیف مختلفی از ابزارها را برای برنامه ریزی راهبردی میتوانیم استفاده کنیم:

- مایکروسافت افیس به طور کلی ورد و اکسل
- سایر case tools که برای بحث برنامه ریزی استراتژیک استفاده می شوند.
- Mind map tools مثل mind note, miro, edrow max ابزارهایی هستند که به اسم ابزارهای mind map یا conceptual map یاbrainstorming شناخته می شوند. این‌ها ابزارهایی هستند که به ما امکاناتی را می‌دهند که اطلاعاتی را که داریم جمع اوری کنیم و به راحتی دسته‌بندی کنیم. (یک ساختار درختی میتوانیم ایجاد کنیم)
- Balanced scorecards یا کارت های نمره دهی متعادل شده. ابزارهایی هستند برای جمع‌آوری و سازمان دهی اطلاعات که در برنامه ریزی های راهبردی استفاده می‌کنیم، مورد استفاده قرار میگیرند.

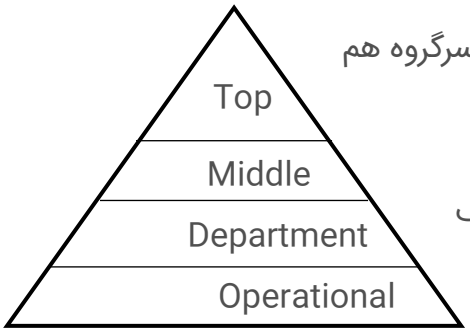
- Gap analysis تحلیل شکاف (شکل حالت مطلوب و وضع موجود، بیایم فاصله بینشان را برحسب شاخص هایی دسته بندی کنیم، وضعیت مطلوب و وضعیت حال را هم شاخص بندی کنیم و فرق این دو شاخص با آن شاخص چقدر است.)

The business case

بیزینس کیس همان مشکلاتی است که در کسب و کار وجود دارد و قرار است به پروژه تبدیل شوند و ما برای این پروژه یک پروپوزال proposal))بدهیم. عملا شناسایی این بیزینس کیس ها موضوع مهمی است.

معمولا پروژه هایی که اعلام می شوند به روش‌های متفاوت می تونایم این بیزینس کیس ها را شناسایی کنیم.

این هرم، هرم سازمان است. در بالای هرم مدیران سطح بالا قرار دارند.



تعدادی مدیران میانی در سطح پایین تر قرار دارند و تعدادی سرکارگر یا سرگروه هم در سطح پایین تر قرار دارند و پایین ترین سطح operational است.

ممکن است ۳ لایه، ۴ لایه و ۵ لایه از اینها داخل دسته بندی های مختلف باشند.

همچین هرمی را می‌توانیم برای کارخانه متصور بشویم.

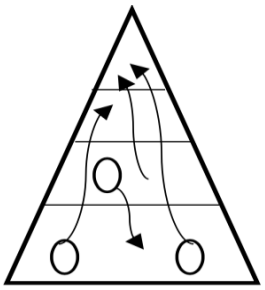
بعضی از پروژه ها از top down اعلام می شوند یعنی از بالا به پایین که برحسب برنامه راهبردی است.

اول action plan ها در واقع از بالا به پایین اعلام می شوند. میگوید بیزینس کیس شما می‌تواند پروژه شماره ۲ باشد.

در قسمت operational نیروهایی درحال کار هستند. مثلا با این ابزارهایی که کار می‌کنیم این مشکلات را داریم یا اطلاعاتی را لازم داریم که در حال حاضر موجود نیست و به صورت پراکنده است.

این اطلاعات را می توان از بخش های دیگر جمع کرد، برای این کار لازم است پروژه تعریف شود یا اصطلاحا یک business case شکل گرفت.(این توضیحات همه در سطح operational اتفاق میافتد). این اطلاعات را به بالادستی ها می دهند و سپس مدیران بررسی می کنند اگر درست باشد یا اگر راه کار ساده ای داشته باشند ممکن است بتوانند در همین مرحله حلش کنند، اگر نه! به مدیران بالاتر منتقل می کنند و همین چرخه تکرار می شود.

یعنی اگر مدیران میانی راه حلی داشته باشند توصیه می کنند و اگر نه به مدیران سطح بالا اعلام می‌کنند و مدیران سطح بالا ازبخش های مختلف. یک سازمان را در نظر بگیرید که بخشهای مختلف دارد و از بخش های



مختلف یک تعداد پروژه آمده است. پروژه ها را مدیران سطح بالا الویت بندی می‌کنند، براساس اینکه هر پروژه چقدر هزینه دارد، الویتش چقدر است و هر پروژه چقدر دست آورد دارد و برحسب مسائل مختلف اینها را دسته بندی می‌کنند، مثلا پروژه بخش فروش در الویت قرار دارد و مثلا n دلار اعتبار هم در نظر گرفته شده است، این پروژه را شروع کنید و قرار داد ببندید و کارهایش را انجام دهید و بعد از اتمام این پروژه سراغ پروژه دیگری می رویم. به این ترتیب ها می گوییم bottom-up

خیلی از مواقع ممکن است شرکتی از بیرون محصولی را توسعه داده و این محصول را مثلا جاهای مختلف نصب کردند. این محصول را برای مدیران یا نیروی های عملیاتی یا مدیران میانی ارائه می دهند و آن ها متوجه می شدند که درست است که به این مسائل و به این بیزینس کیس خودشان داخل شرکت فکر نکردند ولی واقعا یکی از مسائل جدی است، راهکار این شرکت هم یک راهکار مناسبی است و بعد از ارزیابی اولیه آنها را میخرند، یا حتی ممکن است این نمونه را نداشته باشند، صرفا میگویند چنین مسئله‌ای در سازمان شما وجود دارد، یعنی این بیزینس کیس را یک شرکت بیرونی شناسایی کرده و پیشنهاد میدهد و نهایتا میپذیرند یا رد میکنند و انجام نمی شود.

شرکت هایی هستند که تعدادی developer دارند و تصمیم می گیرند هر پروژه یا بیزینس کیسی را شناسایی کنند که خود developer ها ای پروژه ها را برای شان انجام می دهند که به این نوع پروژه ها in house میگویند.

مثالا شهرداری: فرض کنید شهرداری به این نتیجه می رسد که یک پروژه نرم افزاری برای صدور حساب مشتری هایش انجام بدهد، اینکه نیروهای یک بخشی داخل سازمان فناوری اطلاعات داشته باشند، تعدادی نیرو نرم افزاری و برنامه نویسی کار می کنند و اگر خودشان انجام بدهند in house است.

تجربه ی جهانی نشان داده است که خیلی از پروژه های in house، پروژه های موفقی از کار در نمی‌آیند. معمولا استراتژی in house برای مراکزی خوب است که اطلاعاتشان خیلی مهم است و نمی خواهند شرکت های بیرونی در جریان قرار بگیرند.

استراتژی out source

یعنی پروژه را به شرکت های بیرونی بدهیم تا انجام بدهند. اینکه چطور پروژه را به شرکت های بیرونی بدهیم یک مسئله جدی است.

یک سری اصطلاحات را باید یاد بگیریم مثلا REP، اینکه چطور بین درخواست های مختلفی که برای ما ارسال می شود انتخاب کنیم (selection)، بخش قرار داد یا contract را چطور منعقد کنیم، بحث نظارت بر پروژه یا

supervisor اینکه مطمئن شویم پروژه چقدر خوب پیش میرود، بچث های نگه داری و پشتیبانی و همچین مسائلی را در out source داریم.

پروژه های open source

یعنی پروژه هایی که از قبل نوشته شده است، source هم در اختیار قرار دارد و کافی نصب کنیم.

BPM (business process management)

این ابزار ها ابزار هایی هستند که در واقع برای ما این امکان را فراهم می کند که تعدادی از بیزینس‌هایی که داریم را با این ابزار، یک نرم افزار توسعه بدهیم.

فرض کنید ما یک ابزار داریم که اگر فرایند کار را دقیق در بیاوریم که چه مراحلی را طی می‌کند، اول چه اطلاعاتی را از کدام بخش ها می‌گیرد و بعد این اطلاعات را چگونه به واحد های بعدی میفرستند، چه کسی این اطلاعات را تایید می‌کند، بعد از تایید باید به کدام بخش فرستاده شود، اگر تمام این اطلاعات را دقیق استخراج کرده باشیم، اصطلاحا دقیق پروژه را تحلیل کرده باشیم،BPM این اجازه را به ما می‌دهد که چطور این فرایند را بسازیم داخل نرم افزار و پروژه را انجام دهیم. BPM ابزاری است که ما میتوانیم processها یا work flow را مدل کنیم. قابل استفاده برای پروژه های خیلی پیچیده است.

Business case مشکلاتی است که ما استخراج کردیم و انتظار داریم که پروپوزال برایش بنویسیم و در پروپوزال مشخص شود که این مسائل دقیقا چه هستند و چه راهکارهایی وجود دارد و فناوری، منابع و زمان مورد نیاز را باید توضیح بدهیم، باید خیلی جامع باشد و فهمش هم خیلی راحت باشد و خیلی از سوالات را بتواند جواب بدهد.

اینکه چرا این پروژه را انجام میدهیم؟ چقدر هزینه و زمان از ما خواهد گرفت؟ چه ریسک هایی در انجام این کار وجود دارد؟ معیار های موفقیت پروژه چیه؟ اگر این راه را انجام دهیم چه راه حل های alternative یا جایگزینی وجود دارد؟

ما می‌توانیم تقاضای راه اندازی سیستم یا انجام پروژه را از بخش های مختلف بگیریم. در system request یا تقاضای سیستم باید مشخص باشد که چرا این بیزینس کیس پیشنهاد می شود یا راه حلی یا پروژه ای پیشنهاد می شود. آیا این پروژه باعث می شود ما کنترل بیشتری روی سیستم داشته باشیم؟ باعث می شود که اطلاعات بیشتری را از سازمان بدست بیاوریم؟ آیا باعث می شود کارایی یا عملکرد سیستم بهتر شود؟ آیا باعث می شود که سرویس های ما بهبود پیدا کنند؟ آیا باعث می شود که پشتیبانی محصولات جدید و خدمات پس از فروش بهتر شود؟ و...

میتوانیم خیلی دلایل دیگر بیاوریم و قرار نیست همه اتفاق بیافتد، یعنی وقتی پروژه ای را پیشنهاد می‌دهیم، این پروژه قرار نیست همه ی مسائل سازمان را حل کند، همین که باعث کنترل بیشتر شود، و البته بستگی به الویت های سازمان می شود. یا مثلا باعث می شود کارایی سیستم بالاتر برود.

دلایل دیگریست، هزینه ها کاهش پیدا می کند یا خدمات بهبود پیدا می کند. این ها همه عواملی هستند که ما میتوانیم برجسب این دلایل پیشنهاد بدهیم که سیستم را راه اندازی کنند. تعدادی از این عوامل داخلی هستند و تعدادی خارجی. اینکه برنامه راهبردی ما چیست طبیعتا تاثیر گذار است بر پروژه هایی که تعریف میکنیم. آن جایی که برنامه راهبردی را تقسیم کردیم به action plan یا برنامه سالانه، اینکه ما چه پروژه ی راهبردی داریم تاثیر گذار است که ما چه پروژه های سالانه ای را تعریف کنیم. اینکه مدیران ما چه کسانی هستند و چگونه فکر می کنند، نیاز های کاربران چه هست؟ گروهIT، سیستم ها و اطلاعات موجود به چه شکل هستند؟ و چگونه کار میکنند؟ و منابع مالی چیست؟ این ها همه عوامل داخلی هستند. ممکن است ما به این نتیجه برسیم که ۱۰ تا پروژه باید تعریف کنیم که کارمان بهتر شود، خدماتمان بهتر شود ولی واقعا منابع مالی این اجازه را نمی‌دهد.

عوامل بیرونی هم شامل دولت و قوانینی است که وضع می‌کند و نگاهش به توسعه، اینکه چه فناوری هایی در اختیار ما است. فناوری های جدید چه هستند و تولید کنندها و تامین کنندها و ورودی هایی که ما احتمالا در کارخانه داریم یا سازمان، مشتری های ما چه کسانی هستند؟ رقبا چه کسانی هستند؟ وضعیت اقتصادی جامعه؟...

برای مثال شرکتی برنامه ریزی می کند و در حوزه ای شروع به کار می کند و محصولی را توسعه می‌دهد و دستگاهی را میسازد اما دلیل اینکه فکر می‌کرد میتواند موفق شود این بود که تا الان وارداتی از بیرون نداشتیم اما دولت قانونی وضع میکند که واردات داریم، این محصول که داخل تولید می شود نمیتواند خوب بفروشد. پس این قانون دولت روی موفقیت یا عدم موفقیت شرکت تاثیر می‌گذارد.

برای مثال ما برنامه ریزی کردیم دستگاهی بسازیم که بخشی از قطعات از خارج ایران وارد شود، وضعیت اقتصادی وثبات اقتصادی الانوجود ندارد و به شدت گران می شود، بنابراین محصول پیش بینی شده ما هم قیمت تمام شده اش خیلی بالا خواهد بود و احتمالا دیگر مشتری نخواهد داشت. به همین سادگی این عوامل می‌توانند روی شکست یا موفقیت تاثیر گذار باشند، البته نه در واقع کاملا بلکه شبکه ای از این عوامل هر کدام سهمی در موفقیت شرکت دارند.

معمولا کمیته ای تشکیل می شود و پروژه ها و تقاضاهایی که می‌دهیم را بررسی میکنند که کدام پروژه واقعا مورد نیاز است و کدام پروژه در الویت قرار ندارد و برای چه پروژه ای اعتبار داریم. معمولا یک فرم طراحی می شود که معمولا تحت وب است که مشخصات کسی که ایده مطرح می‌کند و توضیحات لازم در خصوص توضیح مسئله و اینکه چه مشکلی یا چه بیزینس کیسی در سیستم وجود دارد و برای ان یک RFP تنظیم کنیم.

وقتی یک مشکل داخل یک سازمان یا یک شرکتی شناسایی می شود و قرار است راه حلی داشته باشیم و یک فرم نرم افزار برایش بنویسیم، یک request proposal تهیه می شود که تا حدی مشخص می‌کند چه کسانی از این مسئله تاثیر میپذیرند، چقدر باعث شده این مسئله روی گردش مالی ما اثر بگذارد.

یعنی مسئله را تا حد ممکن از زوایای مختلف بهش میپردازیم و منتظر هستیم شرکت ها پیشنهاد های خود را برای ما ارسال کنند. شرکت های مختلف هم این request proposal را می بینند و شروع میکنند راه حل ارائه می‌دهند و ارسال می‌کنند. یک کمیته ای این پروپوزال ها را برسی می‌کند و می‌گوید فلان شرکت پیشنهاد خوبی داده است و قیمت و زمانبندی خوبی دارد پس انتخاب می شود این فرایندی است که معمولا برای انتخاب پروژه ها طی می شود.

Feasibility study: بخش مهم

هر پروژه ای را که می خواهیم انجام دهیم، یا خودمان یک عضو و یا مدیر یا کارمند یک سازمانی هستیم که تقاضای پروژه را مطرح کرده است، و یا شرکتی هستیم که میخواهیم یک پروژه را اجرا کنیم. حتما feasibility study باید انجام شود.

Feasibility studyبه این مفهوم است که ما باید از جوانب مختلف برسی کنیم که این پروژه انجام است یا نه!

از چه زوایایی معمولا برسی می شود؟

Economic (financial): یعنی از نظر مالی این پروژه چقدر عملیاتی است؟آیا اعتبار مورد نیاز را داریم؟ مثلا شهرداری یک دیتاسنتر راه اندازی کند خیلی پروژه خوبی است ولی چقدر هزینه میخواهد؟ بعد از برسی متوجه شدند ۱۲ میلیارد هزینه میخواهد. آیا این مبلغ را شهرداری میتواند پرداخت کند؟ آیا انقدر هزینه کنند سرور بدست می اید(نه لزوما)بهره ای که میبرد، آیا این دو با هم، همخوانی دارند؟ پس گفتیم هم از نظر اینکه چقدر منابع مالی لازم داریم برای پروژه. حالا هم منابع و هزینه های ملموس و هم نا ملموس. چقدر اعتبار نیاز دارد؟ چقدر پول باید هزینه کنیم؟ چقدر آورده دارد؟ یعنی چقدر پول به شرکت دوباره برمیگردد؟

برای مثال در شهرداری پروژه ای را هزینه کردند ۱ میلیارد، از این پروژه ۱ میلیارد خدمت بدست نمی‌آید در عرض یکسال ولی در عوض باعث شده کارها سریع تر انجام شود، رضایت مشتری بیشتر شود، از طرفی تعداد کارمندهایی که قبلا برای انجام این خدمت داشتند ۵ نفر بودند حالا می شوند ۲ نفر. مثلا حقوق ها ماهی ۳ نفر کمتر باید حقوق بدهند. اگر این جمع و تفریق ها را حساب بکنیم متوجه میشویم آورده‌ی خوبی داریم.

Tangible cost: یعنی هزینه های ملموس، یعنی هزینه های خرید خرید نرم افزار و سخت افزار و ابزار و حقوق نیرو انسانی

Intangible cost: یعنی هزینه های نا ملموس یعنی فرض کنیم باعث می شود ما اعتبار کاربرها را به یک موضوعی کم شود، اینجا ما داریم هزینه پرداخت می‌کنیم.

Tangible benefits: مزایای ملموس یا سودها همان هایی که میتوانیم حساب و کتاب کنیم که ما ظرف یکسال از هر مشتری قراره ۱۰ دلار بگیریم و پیش بینی می‌کنیم که ۱۰۰ نفر این را بخرند بنابراین ۱۰۰۰۰ دلار می شود درامد یا سود یا منفعت ملموس.

Intangible benefits: مزایای نا ملموس مثلا باعث می شود برند خودمان را تقویت کنیم، این یک مزیت ناملوس است، اینکه مشتریان ما باعث می‌شوند که یک محصول با کیفیت بالاتری را داشته باشند، رضایتشان بیشتر می شود.

Technical: ما از نظر فنی امکانش را داریم که این پروژه را انجام دهیم؟ چه برای شرکتی که مجری طرح است و فناوری در اختیارش است و چه ما که میخواهیم بهره وردار‌ی کنیم، فناوری های مورد نیازش را داریم یا نه؟

امکان سنج فنی یعنی اینکه ما باید برسی کنیم که ایا از نظر فنی امکانات لازم را داریم یا نه؟ آیا اصلا با این دستگاه ها و ابزارها و با زبان های برنامه نویسی که میخواهیم این کار را انجام دهیم افرادی را داریم که بتوانند باهاش این کار ها را انجام دهند؟

Schedule: آیا داخل تابعی که در نظر گرفته شده، آیا این زمان مورد پذیرش ما است؟ ممکن است پروژه‌ای طی دو سال انجام شود و به درد ما نخورد در حالی که ما میخواهیم پروژه طی ۴ ماه مسئله را حل کند.

امکان سنجی زمانبندی یا برنامه ریزی است که باید به این بپردازیم که این پروژه ما برامون خیلی است که توی ۳ ماهه انجام شود. آیا این پروژه با این برنامه ریزی هایی که داریم در ۳ ماه قابل انجام است؟ چه عواملی روی برنامه ریزی ما تاثیر می گذارند؟ چه عواملی ممکن است برنامه ما را عقب بیاندازد.

Operational: (موضوع مهم)(عملیاتی)آیا پروژه ای که ما داریم انجام می‌دهیم از نظر عملیاتی در آینده مورد استفاده قرار می گیرد؟ خیلی از تیم های استارت اپی برای خودشان یک مسئله تعریف میکنند. دقیقا این **feasibility study** را باید انجام دهند. به طور مثال من میخواهم یک اپلیکیشن بنویسم که مردم خریدهای های خود را از طریق اپلیکیشن انجام دهند. حال ما باید برسی کنیم از نظر اقتصادی چقدر هزینه لازم دارد و چقدر آورده دارد، مدل اقتصادی چیست؟ درآمد چگونه است؟ هم سود و هزینه اش برای مشتری چگونه خواهد بود؟ برای ما چطور؟ از نظر فنی زبان برنامه نویسی که می خواهیم کار را انجام دهیم امکانش را داریم؟آیا مردمی که میخواهند این اپلیکیشن را نصب کنند زیر ساختش را دارند؟ مثلا گوشی smart دارند؟اینکه برنامه ما قراره روی یک ابزارهای خاصی نصب شود؟ از نظر زمانبندی این پروژه طی چه مدت انجام می شود؟ آیا اطلاع داریم رقبامون محصول را در مدت کوتاه تری میخواهند به بازار عرضه کنند یا نه؟ این ها را تحلیل میکنیم. فرض میکنیم ما این اپلیکیشن را نوشتیم واقعا کسی خرید میکند؟ نصب میکند؟ این ها **مسائل امکان سنجی** هستند.

Legal: قانونی هم میتوانیم به چهار مورد بالا اضافه کنیم.

امکان سنجی عملیاتی یعنی وقتی توسعه پیدا کرد آیا این ابزار اصلا مورد استفاده قرار میگیرند یا نه؟

حال اگر قرار است این پروژه در سازمان اجرا شود باید به فرهنگ سازمانی توجه کنیم، اگر در جامعه قرار است پروژه استفاده شود باید به فرهنگ مردم توجه شود. در سال ۱۹۹۵ شرکت آمازون کار خودش را شروع کرد به عنوان یک فروشگاه اینترنتی ولی در آن زمان خیلی خرید انجام نمی‌شد چون فرهنگ مردم، فرهنگ خرید اینترنتی نبود. یا زمانی که دیجیکالا راه اندازی شد، قبل از آن تعداد زیادی شرکت هایی که کار فروش اینترنتی را انجام میدادند شروع کرده بودند ولی چون فرهنگ مردم، فرهنگ اینترنتی نبود یعنی به خرید اینترنتی باور نداشتند بنابراین خیلی استفاده نمیکردند. این ها همان بحث های operational feasibility است. حال باید چیکار کنیم؟

طبیعتا باید برنامه ریزی داشته باشیم، تحلیل می کنیم، بررسی میکنیم اگر شدنی بود روی آن فرهنگ مورد نظر آموزش می دهیم یا مشوق هایی را قرار می دهیم که از ابزار ما استفاده کنند.

یک پروژه را تعریف می کنیم اگر مدیران بالا دستی و کاربران واقعا از پروژه حمایت نکنند عملیاتی نمی شود، منظور از عملیاتی شدن این نیست که پروژه را بتوانیم بنویسیم و اجرا کنیم. اینها بحث های تکنیکی هستند، عملیاتی یعنی اینکه استفاده کنند. اگر ما پروژه ای را نوشتیم حالا به هر دلیلی در یک سازمانی هیچ کدام از کاربرها اعتقادی به استفاده از پروژه ما ندارند عملا این پروژه ما شکست خورده است، باید براش برنامه ریزی داشته باشیم. آیا باعث می شود یک سری کاهشی در جریان آن چرخه گردش کار چیزی داشته باشیم یا نه!

اگر کارشناس فناوری و اطلاعات یک سازمان شویم، یکی از کارهایی که انجام دادنش ضرورت دارد، استخراج مشکلات سازمان و اولویت بندی آن ها است. در نهایت باید برنامه ریزی جهت تبدیل آن مشکلات به پروژه هایی با اولویت پیاده سازی توسط کارشناس IT صورت گیرد.

ازجمله کارهایی که باید حوزه IT صورت بگیرد می توان به بروزرسانی نرم افزار ها و زیر ساخت سیستم اشاره کرد (این کار اجتناب ناپذیر است و همه سال انجام می شود)اما بعضی پروژه ها برای بهبود فعالیت ها است برای مثال خدمات با کیفیت بهتری ارائه شوند یا برخی کار ها که پیش از این به صورت دستی انجام می شدند با روش های الکترونیکی انجام گیرند.

این پروژه ها به دو شکل Top Down و Button Up اولویت بندی می شوند. در واقع به این شکل که، کاربرانی که در لایه ی نهایی محصول در حال کار کردن هستند مشکلات موجود در سیستم را گزارش می دهند که انتظار می رود راه حل مناسبی برای آن ها پیدا شود و نهایتا در قالب پروژه راه حل ها پیاده سازی شوند.

در مباحث پیشین به مطالعات امکان سنجی به عنوان یکی از موضوعات مهم اشاره کرده ایم و گفتیم که زوایای مختلف را باید بررسی کنیم. برای مثال به پرسش آیا این پروژه قابل پیاده سازی است یا خیر، حداقل با چهار معیار پاسخ دهیم.

از این معیار ها می توان به موارد زیر اشاره کرد:

- بررسی منابع مالی موجود برای پروژه، اطمینان یافتن از عملی بودن زمانبندی انجام شده (که می توان در مدت معین شده پروژه را به پایان رساند)
- از لحاظ تکنیکال مورد بررسی قرار دهیم (دانش فنی مورد نیاز تجهیزات و تکنولوژی و زیرساخت لازم برای پیاده سازی پروژه موجود باشد.
- از نظر operational (بررسی این که محصول آماده شده آیا کارایی لازم را برای مشتریان و کاربران نهایی را دارد و مورد استقبال ان ها قرار خواهد گرفت؟)

برای مثال در بازدیدی که در دهه ۸۰ از غار علی صدر داشتیم پروژه ای برای فروش بلیط توسط شرکت مطرح شد. ما پروپوزال کامل و راه حلهای مبتنی بر استفاده از بستر اینترنت برای فروش اینترنت را ارائه دادیم. در این ارائه مشتریان می توانستند تاریخ و روز بازدید و انواع سانس ها را انتخاب کنند. اطلاعاتی نظیر تعداد سانس های موجود نیز در اختیار آن ها قرار می گرفت. آن ها می توانستند از طریق اپلیکیشن و وب سایت و کیوسک هایی که در سطح شهر نصب میشد از این خدمات استفاده کنند.

موضوعی که اهمیت داشت این بود که شرکت سیاحتی علی صدر به عنوان بخش خصوصی مدیریت این مکان توریستی آیا واقعاً تا آن روزبه فکر راه اندازی یک سیستم فروش بلیط نبوده است؟

پس از بررسی هایی که انجام دادیم و مصاحبه هایی که با مدیران شرکت داشتیم متوجه شدیم که آن ها در اوایل دهه ۸۰ حتی سامانه ای برای فروش باید به صورت آفلاین و بر اساس کد ملی افراد ترتیب داده بودند تا از مشکلات فروش بلیط بازار سیاه در فصول پر بازدید سال با، این محدودیت ها جلوگیری کنند. و پروژه ها به خاطر اینکه قایقران ها را در نظر نگرفته بودند عملیاتی نشد.

اما چه ارتباطی بین قایقران ها و پروژه وجود داشت؟!

چون اقوام آن ها کسانی بودند که در بیرون غار بلیط ها را میخریدند و میفروختند و درآمدی خوبی کسب می کردند اما بعد پیاده سازی پروژه از نظر operational زیر سوال رفت. چون بدون حضور قایقران ها در غار عملا کسب و کار توریستی بازدید از غار کاملا کنسل می شد.

آنچه گفته شد به این معنی نیست که بعدا دیگر با این مشکل مواجه نشدند، باید راه کاری برای حل این مشکل پیدا میشد.

اگر پروژه ای از لحاظ **operational** قابل اجرا نباشد به اصطلاح، **خلاء تحلیل مشتری** می گوییم.

اگر تعداد پروژه هایی که به بخش فنی واگذار می شود خیلی زیاد تر از توان تیم برای پیاده سازی آنها باشد، باید پروژه ها اولویت بندی شوند. این اولویت بندی بر اساس فاکتور هایی که در ادامه توضیح خواهیم داد صورت میگیرد.

باید به این موضوع توجه داشته باشند که کار اولویت بندی داینامیک است، یعنی سالیانه احتمالا نیاز به بازبینی دارد. یکی از فاکتور هایی که می تواند در اولویت بندی موثر باشد این است که پروژه پیشنهاد شده چقدر می تواند موجب کاهش هزینه ها شود؟ یا اینکه چقدر می تواند بازدهی سیستم را بالا ببرد؟ آیا می تواند در بدست آوردن اطلاعات بیشتر یا بهبود نتیجه ها موثر باشد؟

یا اصطلاحا آیا مزایای این پروژه ملموس است یا نیست؟ گاهی وقت ها این نا ملموس بود اهمیت بسیاری دارد. یعنی می توان همه فاکتورها را در غالب اندازه گیری و عدد و رقم و سود مالی محاسبه کرد.

باید پرسید که آیا پروژه می تواند سیستم بهتری از خدمات به مشتری ارائه دهد؟و یا می تواند سازمان بندی بهتری را معرفی کند؟

این سوال ها به بهبود نتیجه اولویت بندی کمک خواهد کرد. باید در نظر داشت که منابع مرتبا بررسی شود برای مثال پروژه ای با وجود تمام ویژگی ها و فاکتور های مثبتش دریک بازه زمانی ۵ ساله قابل انجام است. پس باید از زمانبندی مطمئن بود. منابع انسانی و مالی و فنی همه در اولویت بندی تاثیر گذار خواهند بود.

انجام دادن روتین یک سری از طرح ها اجتناب پذیر است برای مثال بروز رسانی.

فرض کنید سامانه ی اتوماسیونی را خریداری کرده ایم. شرکت سازنده update جدیدی را ارائه می دهد که ما باید آن را خریداری کنیم. پس باید هزینه مربوط به بروز رسانی نیز در نظر گرفته شود. احتساب مالیات و در نظر گرفتن بخشی از درآمد برای پرداخت ان ضروری است. و همچنین تغییرات فصلی که می تواند روی زیر ساخت ها و سامانه مرتبا تاثیر گذار باشد.

برخی از این پروژه ها توسط تیم مدیریت انتخاب می شوند و به عنوان پروژه با بیشترین اولویت به تیم معرفی می شوند. اگر می خواهیم اولویت بندی به درستی صورت گیرد باید با برنامه ریزی پیش برویم و دقیق باشیم. برای این کار لازم است مدیران کلیدی را ببینیم و با کاربران صحبت کنیم. از افرادی که در حوزه IT کار می کنند درباره ی پروژه توضیح بخواهیم. برای مثال این که هر یک از پروژه ها چه اهدافی را دنبال می کنند یا چه سوالاتی را پاسخ می دهند یا چه کارهایی را باید انجام دهند. تمام این نکات باید در برنامه ریزی ما وجود داشته باشد. آنچه گفته شد یکی از کارهایی است که تحلیل گران در جلسه مصاحبه بر عهده دارند.

از جمله مهارت های کلیدی مورد نیاز تحلیلگران می توان به مهارت پیش بردن مصاحبه های فردی و گروهی اشاره کرد. لازم است که در صورت امکان تحلیلگران جلسه ای برای مصاحبه با افراد نام برده ترتیب دهند و برای

ان ها نقششان را توضیح دهند و سوال های گفته شده را مطرح کنند و ان ها را تشویق به طرح نظریه کنند و در نهایت با جمع بندی اطلاعات بدست آمده نتیجه گیری صحیحی برای اولویت بندی ها کنند.

توصیه می شود اولویت های بالاتر به پروژه هایی که باعث بهبود عملکرد سازمان می شود داده شوند. می توان در این موارد خیلی روی مشکلات موجود درباره ان که پیش از این به ان ها اشاره کرده ایم، تمرکز نکرد.

مسائلی مانند هزینه، سود، فرصت هایی که توسط پروژه ایجاد می شود، محدودیت های موجود را می توانیم از گزارش های مدیریتی استخراج کنیم.

به طور کلی گزارش طرح ما باید دارای ساختار زیر باشد:

۱. نشان دادن اینکه مسئله به خوبی فهمیده درک شده است.
۲. باید به خوبی قوانین و محدودیت های پروژه شرح داده شوند.
۳. حتما اینکه اطلاعات مناسبی از سازمان بر حسب فاکتور های زیر استخراج شده است نشان داده شود:
   - بررسی چارت سازمانی
   - مورد مطالعه قرار گرفتن مستندات
   - ساختار سیستم و نحوه عملکرد ان به طور دقیق مورد بررسی قرار گرفته است
   - یک سری سرور و پیمایش هایی با کاربران سیستم صورت گرفته است و اطلاعاتی از آنها بدست آمده است
۴. مطالعات امکان سنجی صورت گرفته است :
   - Operationalاز نظر عملیاتی از مورد استفاده قرار گرفتن پروژه، اطمینان حاصل کنیم.
   - Technical از نظر فنی مطمئن شویم تکنولوژی لازم برای توسعه محصول موجود است.
   - Economic منابع مالی موجود هستند.
   - Schedule باید در نظر داشته باشیم IDE که پروژه برای به ارمغان خواهد آورد قابل توجه باشد و زمانبندی ان مناسب باشد.
   - ساختار پروپوزالی که باید ارائه شود:
     - Study, Usability, Cost, benefit, Schedule data

باید به خوبی نشان دهیم که مسئله کاملا فهمیده شده است و اگر مشکلی وجود دارد چه تاثیری می تواند به روی افراد مختلف، مشتریان، مدیر سیستم داشته باشد. مسائل و مشکلاتی که قرار است به عنوان یک business case به آنها بپردازیم و اولویت پروژه را تعیین کنیم.

برای ان که نشان دهیم که مسئله به خوبی فهمیده شده است باید با قابلیت های نرم افزاری که قرار است بسازیم و قابلیت های سیستم های دیگر به خوبی آشنا باشیم که به این کار Scope یا boundary سیستم می گوییم.

یکی از ۹ ویژگی سیستم علاوه بر روابط بین اجزا محدوده ی سیستم است. در مورد سیستم های فیزیکی بر خلاف سیستم های نرم افزاری تشخیص scope پروژه سخت نیست. در سیستم های نرم افزار باید تمرکز تان را روی قابلیت ها و ویژگی های مورد انتظار که قرار است اتفاق بیافتد بگذارید. یعنی مشخص کنیم چه مواردی جز پروژه ما هستند و چه مواردی نیستند.

مثالی از انواع محدودیت ها:

محدودیت های ما در بخش خارجی-فرض کنید شرکتی در حوزه توریسم فعالیت دارد. قوانینی که توسط میراث فرهنگی، سازمان اطلاعات کشور و وزارت ارشاد... گذاشته می شوند **قوانین اکسترنال** نام دارند.

اما بعضی از قوانین داخلی هستند مانند محدودیت های کسب و کار که خود دارای آیین نامه ها و شرایطی است. بعضی از محدودیت ها مربوط به زمان حال و بعضی دیگر مربوط به آینده است. محدودیت های آینده بیشتر آگاهی است که یعنی باید به عنوان شرایط مطلوب در نظر گرفته شوند.

برخی قوانین مانند قوانینی که سازمان میراث فرهنگی گذاشته است اجباری هستند در صورت عدم رعایت منجر به تعطیلی کسب و کار می شوند.

Fact finding

یعنی پیدا کردن حقایقی از سیستم که طی فرایند زیر صورت میگیرد:

جلساتی را با مشتریان هماهنگ می کنیم. دراین جلسات اطلاعاتی از طریق مصاحب ها پرسش نامه کسب می کنیم، کار آنها را مشاهده می کنیم. مستندات و اسناد رئیس های سازمان را مطالعه می‌کنیم. در نهایت اطلاعاتی درمورد اینکه فعالیت سازمان به چه شکل است، چه مواردی برای آن اهمیت دارد، ارزش های آن چیست، مشتریان آن ها چه چیزی نیاز دارند را استخراج می کنیم.

در نهایت باید گزارشی شامل موارد، خواسته های سیستم، مشکلات سیستم، یافته های ما از مطالعه آن، توصیه های ما، نقش های پروژه در صورت انجام گیری، چه افرادی با چه وظایفی را آن کار خواهند کرد، تخمینی از زمان بندی و هزینه ی پروژه، مزایای آن و... تحویل دهیم.

قالب پروپوزال

Life cycle سیستم های توسعه نرم افزار را می توان در چند گام خلاصه کرد.

- گام های مربوط به planning
- گام های مربوط به A&D

- گام های مربوط به Implementation و گاهی این گام با Test پیاده سازی می شود.
- گام Maintenance

نگهداری	پیاده سازی	تحلیل و طراحی	برنامه ریزی

در نهایت مشکلات سازمان تبدیل به پروژه هایی می شوند که طی یک زمانبندی و برنامه ریزی دقیق انجام می شوند و باعث بالا رفتن بهروری سیستم می شوند.

فصل ۳

در این فصل درباره‌ی نحوه‌ی مدیریت پروژه های IT و نرم افزاری را می‌پردازیم.

ممکن است وظیفه مستقیم یک تحلیلگر مدیریت پروژه نباشد، معمولا افرادی تجربه ی خوبی در تحلیل پیدا می‌کنند و مدیر پروژه موفقی می شوند که بدانند مدیر پروژه چه وظایفی دارد و چه کارهایی را باید انجام دهد، اینها از جمله مواردی‌ست که نیاز است تحلیلگر به همه آن اشراف داشته باشد.

اهداف فصل: یادگیری مثلث سه گانه‌ی پروژه که شامل هزینه، زمان و محدوده‌ی پروژه می باشد، که به بررسی ساختار پروژه و ارتباطی بین آنهاست، می‌پردازد.

- مفاهیمی مانند برنامه نویسی، زمان بندی، پایش و گزارش گیری از پروژه می باشد که نیاز به تسلط کافی دارد.
- با یک مفهوم مهم و کلیدی این فصل آشنا می‌شویم، WBS(work breakdown struct)یعنی ما ساختار پروژه را بشکنیم و به ساختار یا کارهای کوچکتر تقسیم کنیم.
- به الگوی taskها و ارتباط بین آنها می پردازیم. ارتباط بین taskها ممکن است حالتی باشد که برای شروع taskام i باید taskهای قبلی را انجام داده باشیم، یعنی taskام i نیاز به پیش نیاز داشته باشد و همچنین اگر taskها که تمام نشده باشند، نمی توانیم taskهای بعدی را شروع کنیم که با مفهوم مسیر بحرانی پروژه آشنا می‌شویم.
- بدانیم وضعیت پروژه را چگونه گزارش دهیم.
- پروژه نرم افزاری چگونه باید توضیح دهیم و چگونه باید مدیریت شان کنیم.
- در خصوص مفهوم برنامه ریزی(مانند مدیریت مخاطرات) صحبت کنیم.

مدیریت پروژه

این تعریف کلی ست و ارتباطی با پروژه های نرم افزاری ندارد. می‌توان در هر پروژه ی حوزه ی مهندسی این تعریف را داشته باشیم و توسعه دهیم(extend)و از این تعریف استفاده کنیم.

فعالیت‌هایی مربوط به برنامه ریزی، زمانبندی، پایش، کنترل و گزارش گیری از پروژه است که بعنوان پروژه توسعه سیستم های اطلاعاتی، بر روی آن کار می‌کنیم.

پس مدیر پروژه وظیفه اش این است که اگر قرار است یک پروژه انجام شود، برنامه ریزی پروژه را انجام دهد بتواند زمان بندی کند، بداند زمان کلی پروژه چگونه است، آگاهی داشته باشد که در چه مرحله ای از پروژه هستیم و چه کاری را انجام می‌دهیم، توانایی کنترل داشته باشد اگر پروژه در معرض تاخیر افتادن بود بتواند

فعالیت‌ها را طوری باز طراحی کند که تاحد امکان آن تاخیر راجبران کند ودر مراحل مختلف پروژه بتواند وضعیت پروژه را گزارش دهد.

دست آورد مدیر پروژه موفق این است که پروژه در زمان خودش با بودجه ایی که در نظر گرفته شده است تمام کند و دقیقا تمام خواسته ها را اجرایی کرده باشد و رضایت کاربران نیز کسب شده باشد.

مثلث پروژه

مثلث پروژه شامل **محدوده**، **هزینه** و **زمان** می باشد.[مهم]

هرچه محدوده پروژه بیشتر باشد هزینه و زمان طولانی‌تر خواهد بود و هر چه محدوده‌ی پروژه کوچکتر باشد، زمان وهزینه کمتر خواهد بود.

محدوده (space) پروژه ی نرم افزاری، همان ویژگی هایی ست که ما از نرم افزار انتظار داریم و ویژگی هایی که خارج از تصور هست در محدوده پروژه ما نیست. ممکن است بگوییم اگر قابلیت x در پروژه باشد خیلی خوب است اما باعث تغییر چشم گیر هزینه و زمان پروژه ما خواهد شد.

بنابراین اگر توافقی بر بروی هزینه و زمان پروژه شده تغییر spaceمی تواند هزینه را بطور کلی تغییر دهد.

- مدیر پروژه باید بتواند برنامه ریزی پروژه را انجام دهد در برنامه ریزی پروژه مشخص می شود که taskهای لازم برای تکمیل پروژه چیست و زمان و هزینه مورد نیاز را بتوان تخمین (estimating)بزند.
- بتواند پروژه را زمان‌بندی کند. جدول تنظیم کند و مشخص نماید که این taskها چه ارتباطی باهم دارند. هر taskرا چه کسی و چه موقعی از پروژه باید شروع کند تا در نهایت پروژه در زمان مشخص شده تمام شود و باید taskهای بحرانی یا کلیدی را شناسایی کند و اگر معیار تمام شدن پروژه در موعد باشد، باید کاملا تمرکزش بر روی این taskها باشد تا این taskها به هیچ عنوان به تاخیر نیافتند.
- یکی دیگر از وظایف project monitoring می‌باشد، یعنی بتواند پایش کند و بررسی کند کارهایی که برای اعضای تیم در نظر گرفته شده آیا به موقع پیش می‌روند؟

در Project reporting باید دائما وضعیت پروژه را گزارش کرد، چند درصد از پیشرفت داشته؟ چه مقدار از پروژه باقی مانده؟ الان در چه وضعیتی هستیم؟ در اینده به چه منابعی نیاز داریم؟ چه میزان از منابع را هزینه کردیم؟

work breakdown struct(ساختار شکست پروژه)

می خواهیم یک وب‌سایت برای یک رستوران درست کنیم، که یکی از کلیدی ترین کارهایی که میخوایم انجام بدهیم پذیرش سفارش آنلاین غذاست.

بگوییم برای این پروژه به دو ماه وقت نیاز داریم.(البته زمان را بعدا پیش بینی می کنند)

سراغ taskاصلی می رویم باید یک interview با کارکنان، مدیر رستوران و حتی با مشتریان داشته باشیم به عنوان مثال برای این کار دو هفته زمان در نظر می گیریم. بعد از انجام کار در خصوص اینکه نیازمندی ها چه چیزهایی هستند، تحلیل انجام می دهیم. مثلا ۴هفته برای طراحی مدل‌مان در نظر میگیریم. ۲ هفته زمان برای شروع بیاده سازی در نظر می‌گیریم، حداقل کل زمان تخمین زده شده برای taskهایی که استخراج کردیم ۴ الی ۵ هفته می‌باشد.

یعنی taskکلی پروژه را (مثلا)به۲۰ taskکوچک‌تر می‌شکنیم. برای این taskها نیاز به نیروی انسانی داریم تا این interview نیازمندی ها را به تحلیلگر بسپاریم. یک کار را طراح انجام دهد ویک کار دیگر را پیاده ساز انجام می دهد.

معمولا این ساختار شکست پروژه ورودی‌ای برای pert/cmp charts می باشد.

Pert/CMP chart

در نمودار گانت که به آن نمودار میله ای نیز گفته می شود، محور افقی زمان و محور عمودی task هاست.

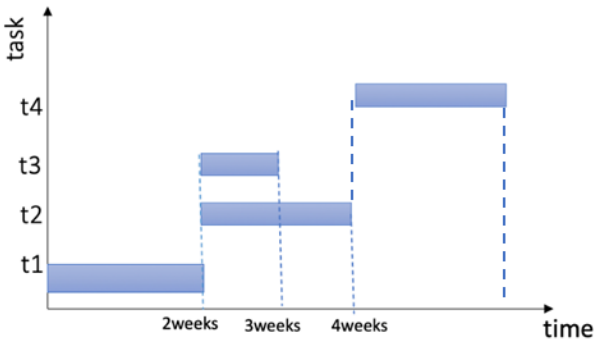
مثلا T۱دو هفته زمان می برد.

برایT۲ که مثلا ۲ هفته زمان میبرد باید اول T۱ تموم شود تا بتواند شروع کند. هم زمان با T۳, T۲ را هم شروع می‌کنیم، مثلا یک هفته زمان میبرد و نهایتا پروژه ما به یک نمودار تبدیل می شود.

که ترتیب و توالی taskها نمایش می‌دهد و اینکه هرکدام چقدر زمان میبرد تا تمام شود.

اما در نمودار شبکه ایی نشان می‌دهد کهtask ها به شکل گراف چه ارتباطی باهم دارند.

در WBSوقتی داریم یک پروژه را به کارهای کوچکتر تقسیم می‌کنیم، باید مشخص کنیم که این task یا activlyما چه چیزهایی هستند، از آغاز تاپایان پروژه چه کارهایی راباید انجام دهیم. هیچ کاری نباید از قلم بیافتد حتی اگر یک کار گزارش گیری هست، جلسات کار با پیمانکار یا کارفرما را باید بدانیم، هر کدام از اینها زمان، انرژی و هزینه خودش را می‌خواهد. اگر چیزی را از قلم بیاندازیم عملا در تعیین زمان و تخمین زمان پروژه و در تخمین هزینه های لازم برای انجام پروژه احتمالا با مشکل مواجه می‌شویم.

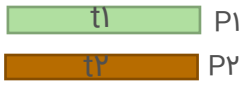


در ساختار WBSما سعی می‌کنیم taskها را لیست کنیم و زمان هرtaskرا برحسب ساعت، هفته، ماه و... محاسبه کنیم.

فاکتورهای که در طول پروژه تاثیر گذارند؛ اندازه پروژه و منابع انسانی در دسترس می باشد.

گاهی اوقات ممکن است چون دو نفر نیرو هستیم یک پروژه کوچک زمان زیادی را از ما بگیرد، این کار در استارتاپ ها خیلی معمول است. یک کار خیلی بزرگ ممکن است کلا ۳ ماه زمان از ما بگیرد چون ۱۰۰ نفر نیروی فعال داریم که در پروژه ما را می توانند کمک کنند بنابراین اندازه پروژه و منابع انسانی که در اختیار داریم بسیار تاثیر گذارند.

ما فرض می‌کنیم که هر taskرا یک نفر انجام می دهد و اگر دو task را موازی در نظر بگیریم به شرطی که پیش نیاز هم نباشند حتما دوتا نیروی انسانی فعال داریم که میتوانیم آن را انجام دهیم، اگر نیروی انسانی نداشته باشیم، عملا این دو به عنوان sequential می‌باشد:



- task ۱، ۳ هفته زمان برای انجام لازم دارد که یک نفر انجام می‌دهد.
- Task۲، ۳ هفته زمان نیاز دارد. (ارتباطی با task۱ ندارد.)

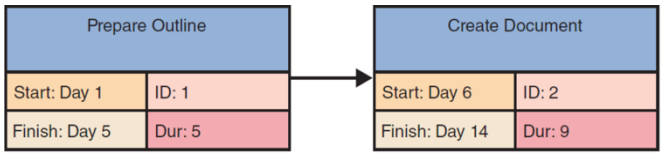
اگر دو نیروی انسانی فعال و آزاد داشتیم، task۱ را بر عهده people۱ می‌گذاریم و task۲ را بر عهده people۲ می‌گذاریم ولی اگر یک نفر را دارم که وقتش آزاد است، task۲را نمیتوان انجام داد باید صبر کرد تاtask۱تمام شود بعد task۲را شروع کنم.

۳ هفته زمان task۱و ۳ هفته هم زمان task۲ پس این کاری که می‌توانستیم در ۳ هفته انجام دهیم حال در ۶ هفته انجام می شود. بنابراین اندازه پروژه و منابع انسانی که در اختیار داریم بسیار تاثیر گذارند.

همچنین اینکه ما چقدر روی پروژه‌های مشابه قبلی تجربه داریم، چقدر کامپونت‌های مشابه را develop کردیم، چقدر توانستیم زمان فعالیت‌های قبلی راحدس بزنیم، چقدر توانستیم محدودیت ها را خوب شناسایی کنیم و... نیز تاثیر گذار است.

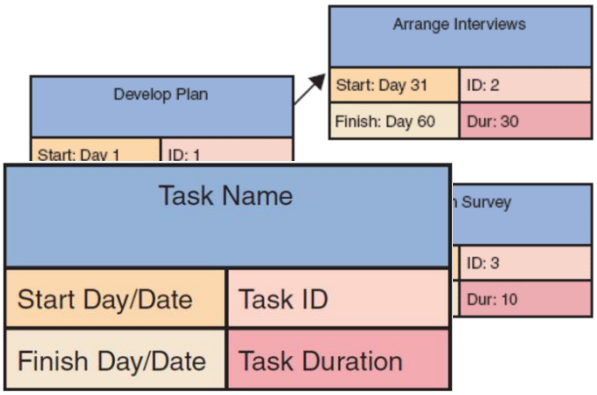
Task pattern

Taskها کارهای کوچکتری هستند که حاصل شکست پروژه اصلی می باشد. برای پیش بینی از نظر زمان و هزینه، این شکستن را ادامه می‌دهیم که بتوانیم پیش بینی کنیم.



الگوهایی که برای Taskها می توانیم متصور شویم این است که این Taskها می توانند به همدیگر وابسته despendent باشند. (یعنی تا زمانی که task۱ را تمام نکرده باشیم نمی توانیم task۲ را شروع کنیم.)

- Multiple Successor Task: چندین Taskباید تمام شود تا این Taskرا بتوان شروع کرد.
- Multiple Predecwssor Task: یک Taskخودش عامل این است که چندین Taskدیگر را بعد آن شروع کنیم.



در نمودار هایی که برای گانت یاpertرسم می‌کنیم معمولا از چنین چیزی استفاده می کنیم.

Despendent

برای مثال این دو taskوابسته هستند و تا task۱تمام نشود task۲شروع نمی شود.

Multiple Successor

task۱باید تمام شود تا بتوان task۲، task۳ را شروع کرد.

(یعنی task۲,task۳به task۱ وابسته هستند.)

Multiple Predecwssor

Task۱

- task۱وtask۲باید تمام شوند تا بتوانیم task۳ را شروع کنیم.

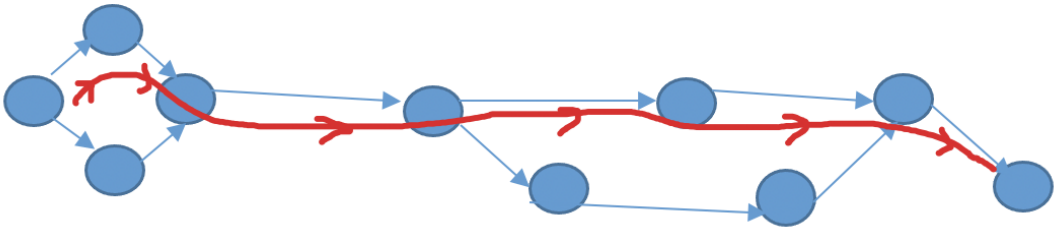
Task۳

Task۲

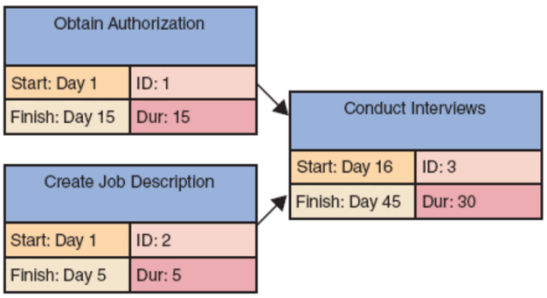
The Critical Path

نمودار مسیر بحرانی یا مسیر کلیدی روی پروژه ها مسیری است که فعالیت های روی آن مسیر قرار دارند. مسیر بحرانی مسیری ست که تمامی فعالیت هایی که بر روی این مسیر قرار دادند نباید به تاخیر بیافتند اگر هر کدام از اینها به تاخیر بیافتند، به همان اندازه که آن فعالیت به تاخیر افتاده است؛ کل پروژه دچار تاخیر می شود.

(با توضیحاتی که دادیم میتوانیم چنین نموداری را داشته باشیم.)



Project Monitormg



برای پیمایش پروژه چنین ساختاری مثل نمودار گانت را به صورت کامپیوتری دائما هر روز در سیستم نگاه می‌کند و printمی‌کند به صورت خیلی بزرگ بر روی دیوار اتاق میزند و حواسش هست که الان چند روز از پروژه سپری شده؟ چه فعالیت هایی را انجام دادیم؟ چه فعالیت هایی به موقع تمام شدند؟ ورودی ها چه چیزی بودند؟ وخروجی ها چه چیزی بودند؟ و....

Reporting

در این مورد توضیح دادیم که مدیر پروژه وظیفه اش این است که دائما بر روی پروژه خودش تسلط داشته‌باشد و در جلسات مختلف گزارش دهد که چقدر از منابع را هزینه کردیم! چه کسانی کمک کردند! چند درصد از پروژه پیشرفت کرده! چه قابلیت هایی از سیستم را پیاده سازی کردیم و کدامشان را تست کردیم.

یکی از کارهایی که در ابتدای پروژه بعد از برنامه ریزی و آغاز پروژه باید انجام بدهیم، این است که بتوانیم زمانی را برای پروژه تخمین بزنیم و هزینه انجام پروژه را پیش بینی کنیم. مواردی هست که ما در پیشنهاد پروژه یا پروپوزال پروژه اشاره می کنیم.

برای بحث تخمین و کنترل پروژه از نمودارهای Gantt و PERT استفاده می شود.

Grantt: نمودار میله ای که معمولاً با ابزار هایی مثل project Microsoft استفاده می شود.

PERT: نمودار شبکه ای است.

هدف این بخش: چگونه یک پروژه را با استفاده از این دو نمودار ترسیم کنیم و کار های کنترل را انجام بدهیم و تخمینی برای زمان انجام پروژه و هزینه داشته باشیم.

اولین قدم این است که بتوانیم موضوع پروژه را به چند بخش کوچک تقسیم کنیم. اصطلاحاً work breakdown structure یا WBS.

این فرایند را تا جایی باید ادامه داد که task ی که شکستیم به راحتی قابل پیش‌بینی باشد و مشخص باشد که چه ورودی هایی دارد، چه خروجی هایی دارد، چه نیرویی رو قرار است به آن اختصاص بدهیم و کار را برای ما انجام بدهد.

روش‌های مختلفی برای تقسیم بندی وجود دارد.

۱. یک عده بر اساس گام های پروژه تقسیم بندی می کنند که مثلا میانگین در فاز تحلیل سیستم چه کارهایی داریم؟ در فاز طراحی سیستم چه کارهایی داریم؟ در فاز پیاده‌سازی چه کارهایی باید انجام بدهیم؟ در فاز تست و استقرار چه کارهایی باید انجام بدهیم؟
۲. پروژه را بر حسب شناختی که از ابتدا تا انتهای پروژه لازم است انجام بدهیم و با شناختی که از قبل داریم تقسیم می کنیم.

Activity	هفته (روز) Time estimate			ET (extended time) (o+۴r+p)/۶
	o	r	p	
جمع آوری نیازمندی ها	۱	۵	۹	۵
طراحی واسط کاربری	۵	۶	۷	۶
طراحی گزارش	۳	۶	۹	۶
طراحی پایگاه داده	۱	۲	۳	۲
مستند سازی کاربران	۲	۶	۷	۵/۵
برنامه نویسی	۴	۵	۶	۵
تست (آزمون)	۱	۳	۵	۳
نصب و راه اندازی	۱	۱	۱	۱

توضیحات یک ID برای هر تست در نظر می گیریم

- Activity یا آن فعالیت هایی که نیاز است برای یک پروژه را بتوانیم شناسایی کنیم.
- ممکن است بگوییم مثلاً طراحی واسط کاربری را تقسیم بندی کنیم. مثلاً یک UI برای مدیر می‌خواهیم طراحی کنیم، اول دو روز وقت نیاز داریم، برای کاربران ممکن است چند تا صفحه جدا بخواهیم مثلاً پنج روز زمان نیاز داشته باشه..... این گونه حتی می شود این موارد را شکست.
- -یک تخمینی از زمان انجام پروژه بزنیم میتواند روز، هفته، ماه باشد. روز سخت ترست و ماه خیلی دقیق نیست، هفته بهتر است.
- یکی از روش های تخمین پروژه روش خوشبینانه بدبینانه هست: (o + ۴r + p)/۶
- O: خوشبینانه ترین حالت چند روز نیاز است برای انجام کار (یعنی همه چیز مرتب باشد نیروها آماده باشند و منابع آماده باشد پول در اختیار باشد و هیچ مشکلی نداشته باشیم، کارفرما ها و کاربران با ما همکاری کنند)

- P: بدبینانه همه شرایطی گفتیم نیاز داشته باشد هماهنگ کنیم و مثلاً سخت افزار آماده کنیم و...
- r: واقعی با توجه به تجربه‌هایی که داشتیم چقدر زمان میبرد که در معادله به آن ضریب ۴ دادیم.
- این فرمول میتواند تغییر کند چیز ثابت نیست تجربه به دست آمده می توانیم در جریان انجام پروژها عوض کنیم مثلا به r ضریب ۲ بدهیم در اینصورت باید تقسیم بر ۴ کنیم (o + ۲r + p)/۴ یا مثلاً (o + ۵r + p)/۷
- -Time estimate را به p، o، r تقسیم می کنیم. ما وقتی مدیر یک پروژه هستیم، باید بتوانیم این حالت ها را حساب کنیم.
- -کار روی اینکه اون ET یا (expected time)یعنی اون اعداد داخل time estimate رو حساب می کنیم.

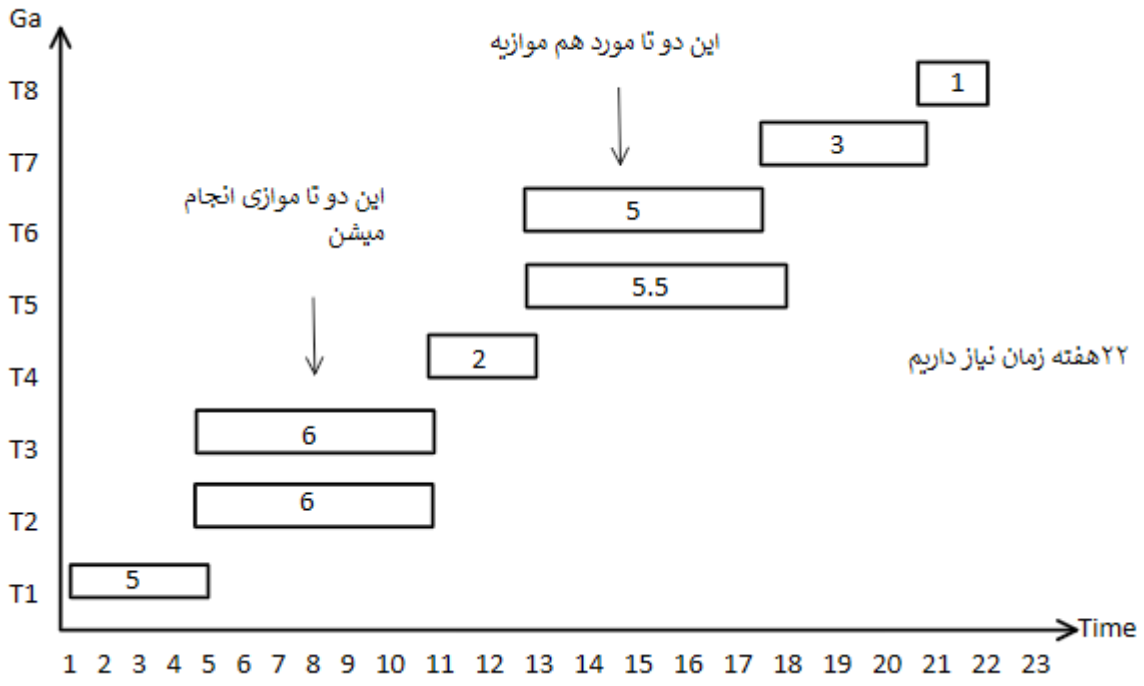
۳- روش های مبتنی بر user case، تجربی، line of code و... هم وجود داره برای تقسیم بندی این روش خوش بینانه، بدبینانه یک روش پرکاربرد است.

- قسمت ET اولین چیزی ست که برای انجام نمودارها مورد نیاز است.
- دومین چیزی که نیاز داریم فعالیت‌های هرکدام پیشواز هایشان چیست.
- منظور از پیش‌نیازهای یعنی اینکه اگر ما می‌خواهیم T۱ را شروع کنیم چون اولین کار هیچ پیشنیازی ندارد مستقیم به این مسئله مصاحبه می‌کنیم.
- برای اینکه T۲ را انجام بدهم لازم است که اول اطلاعات را جمع کرده باشم و بدونم نیازهای کاربر چی هست پس T۱ را به عنوان پیش نیازی برای T۲ اعلام می‌کنیم.

ID	Activity	Preceding activity
T۱	جمع آوری نیاز ها	-
T۲	طراحی واسط کاربری	**T۱**
T۳	طراحی گزارش	**T۱**
T۴	طراحی DB	**T۳وT۲**
T۵	مستند سازی کاربران	**T۴**
T۶	برنامه نویسی	**T۴**
T۷	تست	**T۶**
T۸	نصب و راه اندازی	**T۷وT۶**

برای طراحی گزارش هم من باید جمع آوری نیازها را انجام داده باشم و از روی نیازمندی‌هایی که کاربران داخل سیستم یا کاربرانی که در شرکتی که قرار است نرم افزار برایش بنویسیم اعلام کردند باید اول اطلاعات را جمع کنیم و طراحی گزارش را انجام بدهیم.

- وقتی که ریپورت UI را جمع می‌کنیم می‌توانیم طراحی DB را انجام بدیم پس T۱ و T۲ را به عنوان پیش نیازهای T۴ اعلام می کنید
- برای اینکه بتونیم مستند سازی کاربرد داشته باشیم ما باید T۴ را بگذاریم، ممکن است یکی بگوید T۲ , T۳ , T۱ پیش نیاز است، درست است ولی چون قبلاً T۲,T۳ را به عنوان پیش نیاز T۴ اعلام کردیم نیاز نیست آنها را اینجا بنویسیم یعنی برای T۵ ما باید T۲,T۳ اول باید انجام شوند برای T۴ بعد T۴ را شروع کرده باشیم و تمام کرده باشیم تا بتوانیم T۵ شروع کنیم.
- T۶ را می شود هم زمان با T۵ شروع کرد ولی باید T۴ ما انجام شده باشد پس T۴ را به عنوان پیش نیاز T۶ اعلام می کنیم.
- برای T۷ لازم است که برنامه نویسی تمام شده باشد پس T۶ پیش نیاز می شود.
- برای T۸ باید آزمون نرم افزار را تمام کرده باشیم و هم بحث های مستند سازی کاربر را انجام داده باشیم پس T۵,T۷ الان دومین عاملی که برای نمودار ها لازم بود رو پیدا کردیم preceding activity حال از روی این دو باید سراغ طراحی نمودار گانت و پرت برویم.



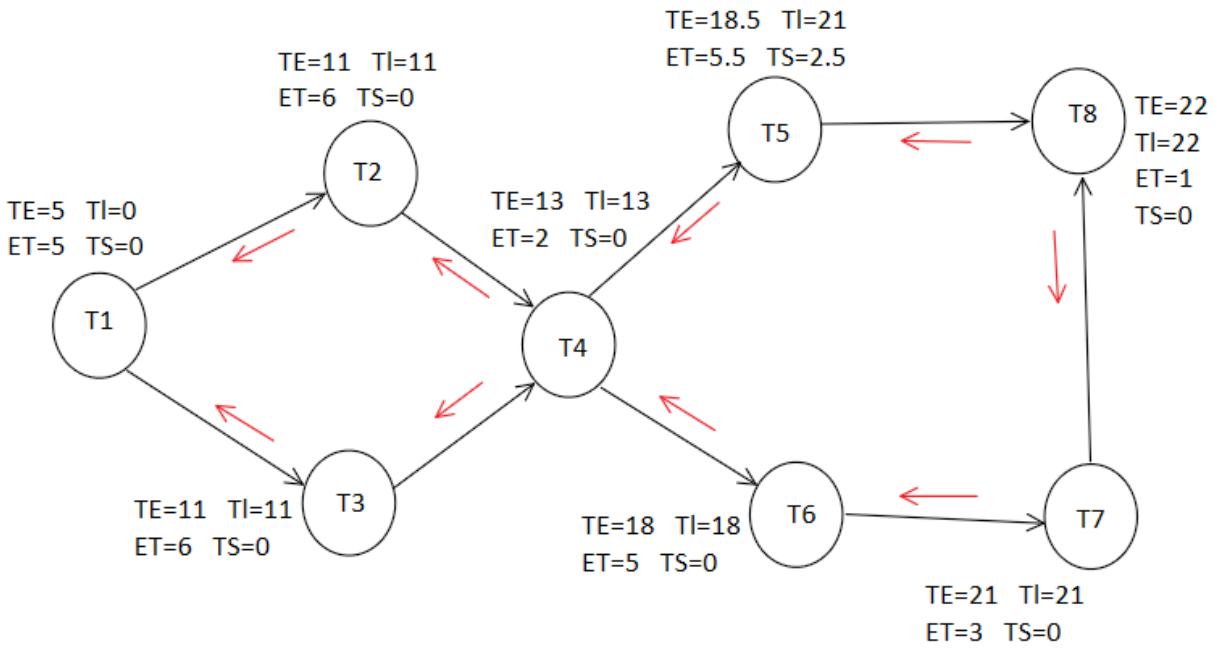
- T۱ را از همان اول میتوانیم شروع کنیم، برای T۲,T۱ باید تمام شود. T۳هم وضعیت مشابه T۲ را دارد.
- T۴ باید T۲,T۳ تمام شده باشد، برای T۵,T۴ تمام شده باشد. برای T۶ کافیه T۴ تمام شده باشد

- T۷ هم باید T۶ تمام شده باشد،T۸ هم T۵,T۷ باید تمام شده باشد.
- پس ما پیش‌بینی می‌کنیم که هفته بیست و دوم پروژه تمام شده باشد
- برای قسمت هایی که موازی است باید نیروی کافی داشته باشیم اگر نداشته باشیم نمی توانیم موازی انجام بدهیم، باید یکی از کارها تمام شود و بعد یکی دیگر را شروع کنیم.
- ما هرچه پروژه را زودتر تمام کنیم بهتر است حتی اگر نیروی کار نداریم بهتر از نیروهای کار ساعتی استفاده کنیم تا این task ها را به صورت موازی پیش ببریم.
- این نمودار را معمولا پرینت می‌کنند و به دیوار می زنند و میگویند کجای کاری هستیم و چه کردیم، آیا کارها بر اساس پیش بینی پیشرفتند یا نه! مدت کاری که باقی کار جمع می شود یا نه! اگر نه که باید فکری کنیم.

مزیت های نمودار گانت:

- به راحتی می تونیم بفهمیم در طول مدت زمان انجام کار چقدر است
- کارهای موازی را به سادگی میتوان تشخیص داد
- کنترل پروژه رو به سادگی میتوانیم انجام بدهیم (مهم)

نمودار PERT یا شبکه ای



- بدون پیش نیاز میتونه شروع بشه چقدر زمان لازم داشت؟ ET=۵هفته
- برای اینکه بتوانیم T۲ شروع کنیم باید T۱ تمام شده باشد و مدت زمان task۲=ET هم خودشان ۶ هفته هستند.
- T۳ هم همین طور
- برای T۴ هم T۲,T۳ بود و زمانش ۲ هفته بود
- T۵,T۶ هردو T۴ پیش نیازشان است ۵,۵,۵, هفته زمانشان است.
- T۷ هم T۶ پیش نیاز است و ۳ هفته زمانشان است.
- T۸ هم T۵,T۷ و ۱ هفته زمانشان است.

erliest time:TE زودترین زمانی که یک فعالیت طول می کشد تا خاتمه پیدا کند.

latest time:TL دیر ترین زمان انجام فعالیت

slack time:TS زمان لختی

TE: از سمت چپ نمودار حرکت می کنیم TE=ET فعالیت قرار می دهیم (T۱)

(T۲)می گوییم ۵ هفته لازم داریم تا T۱ تمام شود و ۶هفته هم خودش زمان نیاز دارد که ۱۱=۵+۶ هفته زمان نیاز است.

(T۳)۱۱=۵+۶

(T۴)فعالیتی که از دو مسیر به آن می رسیم اون زمان رو انتخاب می کنیم که بزرگتر است، چون تا زمان بزرگتر تمام نشده باشد نمی توانیم فعالیت را شروع کنیم البته در اینجا هر دو ۱۱ هفته هستند ۱۳=۲+۱۱

(T۵)۱۸/۵=۵/۵+۱۳ هفته

(T۶)۱۸=۱۳+۵ هفته

(T۷)۲۱=۱۸+۳ هفته

(T۸)دو مسیر دایره دار بکشید ۱۸/۵ هفته دیگری ۲۱ هفته پس ۲۱ انتخاب می شود پس ۲۲=۱+۲۲ هفته، پس این پروژه ۲۲ هفته در زودترین زمان ممکن بر حسب پیش بینی زمان میبرد تا تمام شود.

TL: این بار از سمت راست شروع می کنیم TL آخرین فعالیت را برابر TE اش قرار می دهیم بعد از هفته‌ی خودش کم می کنیم و به سمت چپ حرکت می کنیم.

(T۸): TE=TL=۲۲

(T۷): ۲۲-۱=۲۱

(T۶): ۳-۲۱=۱۸

(T۵):۲۲-۱=۲۱

(T۴):اینجا هر دو بهم رسیدند آن که از همه کوچیکتر است را انتخاب می کنیم بین ۱۸و۲۱=> ۱۳=۵-۱۸

(T۳): ۱۳-۲=۱۱

(T۲): ۱۳-۲=۱۱

(T۱): باز دو مسیر را مقایسه می کنیم ۵=۶-۱۱

|TE-TL|=Tl-TS:Ts (همیشه TL ممکنه بزرگتر از TS دربیاد)

(T۱)=۰ (T۲)=۰ (T۳)=۰ (T۴)=۰ (T۵)=۲٫۵=۱۸-۲۱ (T۶)=۰ (T۷)=۰ (T۸)=۰

مفهوم TS: میگوید میزان سختی و میزان زمان مجاز برای به تاخیر انداختن یک فعالیت برای این که کل پروژه به تاخیر نیافتد این امکان را داریم که فعالیت ۵ تا ۲٫۵ هفته دچار تأخیر شود یعنی اگر تاخیر داشته باشد به پروژه ما لطمه وارد نمی شود و می توانیم سر ۲۲ هفته تمام کنیم اما فعالیت T۱ حتی یک روز هم عقب بیافتد ما ۱ روز تاخیر در کل پروژه داریم یا فعالیت ۶و۷

Critical path:

- طولانی ترین مسیر در شبکه
- سیری که فعالیت های آن همه بحرانی باشند

فعالیت بحرانی: (کلیدی)فعالیتی که TS=۰ باشد

برای این مثال ۲ مسیر بحرانی داریم.

T۱-> T۳->T۴->T۶->T۷->T۸

T۱-> T۲->T۶->T۶->T۷->T۸ ,

- یعنی فعالیت های T۱,T۲,T۳,T۴,T۵,T۶,T۷,T۸ بحرانی هستند و نباید دچار تاخیر شوند.
-
- وضعیت نمودار PERT
- مسیر بحرانی را نشون می دهد و کمک می کند پروژه را کنترل کنیم.

فصل ۴

در ابتدای ترم که شروع شد یه سری مفاهیم و سازه ها رو توضیح دادیم.مهارتهای مورد نیاز برای تحلیلگر را گفتیم. مدیریت پروژه که مجموعه مدیریت زمان و هزینه و... را گفتیم.

انجام تحلیل را می شود در سه گام خلاصه کرد:

۱- جمع‌آوری نیازها
۲- سازماندهی یا مدلسازی و نیازها(مدلسازی تحلیل)
۳- ارائه راه حل و پیشنهاد

مدل سازی تحلیل: اطلاعاتی که ما درباره سیستم جمع آوری کردیم را در قالب مدل نمایش بدهیم. اما برای چه این کار را می‌کنیم؟ برای اینکه فهم بهتری از حل مسئله داشته باشیم.

مدل طراحی: راه حلی که ما برای سیستمی که شناختیم پیشنهاد می‌کنیم.

به راه حل جمع آوری نیاز ها و سازماندهی یا مدل سازی نیازمندی‌ها (مدل سازی تحلیل) Requirement Engineering (مهندسی نیازها)می‌گوییم.

اصطلاح مهندسی که قبلا اشاره شد: یعنی فرایندی که شامل چند گام هست و آن ها را طی می‌کنیم تا قابلیت اطمینان محصول را افزایش و هزینه محصول را کاهش بدهیم.

پس شامل یک فرایندست که نیاز به محاسبات، ابزار و روش تحلیل داده دارد. که به آن **مهندسی‌نیازها** می‌گوییم.

System Requirement

مثلا سیستم بیمارستان، بخش های مختلفش را مثل رادیولوژی، داروخانه، پذیرش، بستری و... را بهم متصل کنیم. یک دیتا بیس یکپارچه داشته باشند و کاراشان را بصورت paperless و اتوماسیونی انجام دهند.

این نیاز نیاز کلی هست و ما به آن stack holder request می‌گوییم.

اما لازم است که ما متوجه شویم داخل رادیولوژی چه کارهایی انجام می شود، چه نیازهایی دارند، در داروخانه چه نیازهایی دارند، چه کارهایی در پذیرش انجام می شود، این شناختی است که قرار شد بدست بیاوریم؛

نیازمندی آنهاست، یعنی سازمان نیاز دارد این قابلیت ها و این کار ها تبدیل به نرم افزار کند، که ما به آنها system requirement یا نیازمندی های سیستم می گوییم.

نیازمندیهایی که درموردشان صحبت می‌کنیم یک Bench marks برای پذیرش سیستم است؛ یعنی قرار است یک نرم افزار را تحویل بدهیم، پس باید یک سری ویژگی های این سیستم را داشته باشد.

این ویژگی ها System Requirement است که در مرحله آن ها را شناسایی می‌کنیم.

البته یک تصویر کلی از آنها را در مرحله ی پروپوزال ارائه دادیم.

System Requirement: اون ویژگی هایی که ما باید در سیستم نهایی که می خواهیم تحویل بدهیم وجود داشته باشد.

فعالیت مهندسی نیازها شامل موارد زیر است:

۱. جمع آوری نیازمندی‌ها (داده ها)
۲. نمایش نیازمندی‌ها (مدل سازی)
۳. ارائه پیشنهاد در انتهای کار (یعنی بگوییم این سیستم چه مشکلاتی دارد و احتمالا چه راه‌حل‌هایی دارد، ما باید راه حل ها را در مرحله طراحی بررسی کنیم و راه حل نهایی را طراحی کنیم)به زبان دیگر ۳ وارسی و تایید نیازمندی ها لازم است. یعنی معمولاٌ این مورد داخل دو گام اول انجام می شود.

فرض کنید که با یک کارمند بخش رادیولوژی یک بیمارستان صحبت کردیم و گفته شده که سیستم باید فلان قابلیت ها را داشته باشد. اگر قبول کنیم و همه این ها را ملاک قرار میدهیم و پیش میرویم احتمالاً وقتی پروژه را تحویل بگیرد بگوید این چیست؟ ما چنین چیزی را نمی‌خواستیم.

برای اینکه مطمئن شویم ویژگی را درست استخراج کردیم لازم است آنها را وارسی و تایید کنیم.(**Validate & Verify**)

روشهایی که معمولاً نیازمندیها را می‌پردازیم این است که انواع نیازمندی‌هایی که سراغشان میرویم، دسته های **نیازمندی های وظیفه مندی** و **غیر وظیفه مندی** هستند.

نیاز وظیفه مندی: آن دسته از نیازهایی که ما انتظار داریم مستقیماً داخل سیستم وجود داشته باشند، عملکرد سیستم عملاً براساس این نیازها مشخص می شود مثلاً در سیستم HIS فرض کنید یک بیمار میخواهد پذیرش شود، اینکه اطلاعاتش ثبت شود و بعد دکمه پذیرش زده شود:*نیاز وظیفه‌مندی* است.

در یک سیستم بانکی افتتاح حساب بصورت الکترونیکی اتفاق بیافتد.

اینکه دانشجو باید لیست دروس را ببیند و بین آنها تعدادی را انتخاب و ثبت کند. اینکه سیستم ما چک کند که این دانشجو مجاز به اخذ کدام واحد هاست؛ نیاز وظیفه مندی می شود.

نیاز غیر وظیفه مندی: نیازهایی که عمدتاً به کیفیت نرم افزار برمیگردد مشخصاً ممکن است در قرارداد یا System Requirement ای که می گوییم ذکر نشده، مثل امنیت، اینکه امنیت سیستم باید بالا باشد. سیستم واسط کاربری یا UI باید User friendly باشد، اینکه performance سیستم خیلی خوب باشد یعنی اگر قرار است یک بیماری را ثبت کنیم و اطلاعاتش را که وارد کردیم و دکمه ثبت را که زدیم، دو دقیقه بعد ثبت نکند، باید کمتر از دو ثانیه این ثبت اطلاعات اتفاق بیافتد.

دسته دیگر نیازمندی ها که داریم براساس ویژگی هایی که نرم افزار دارد، دسته بندی می‌کنیم.

چالش‌هایی که در نیازمندی با آنها سروکار داریم:

Imprecision: غیر دقیق بودن نیازمندی‌هایی که مشخص می شوند.

Agreement: آیا توافقی روی آنها وجود دارد یا نه!

Creep: مشاجره و عدم توافق (البته به معنی خزیدن و آدم مرموزست!)منظور این است که ما با آدم های متفاوتی سروکار داریم، خود این یک چالش هست، یک کارمند ممکن است یک سری اطلاعات را به ما بدهد، یک کارمند ممکن است ندهد، ترسی کارمند دارد که اگر اطلاعات را به ما بدهد ممکن است جای سیستمی که توسعه می دهیم را بگیرد، یا اطلاعاتی که درباره نحوه عملکردش بدهد منجر شود دیگران تسلط بیشتری روی کارش پیدا کنند و احتمالاً یک سری دور زدن قانون و تخلفاتی که انجام داده لو برود، این موضوعاتی هست که در تعیین نیازمندیها با آن مواجه هستیم.

Cognitive bias (سو گیری های شناختی): همیشه فکر می‌کنیم چیزی که می‌بینیم احتمالاً درست است، چیزی می‌شنویم درست است. ذهن ما ذهنی است که عملکردش را درست و میزان است ولی واقعیت اینگونه نیست! ما آدم ها پر از سوگیری های شناختی و متفاوتی هستیم. اینها باعث می شود شناخت ما نسبت به پدیده ها همواره نادقیق باشد و یک موردش که در حوزه کسبو کار مورد توجه قرار می‌گیرد اثر هاله ای هست.

اثر هاله ای: ما آدم ها وقتی یک ویژگی مثبتی را در یک نفر می‌بینیم آن را به موارد دیگر توسعه‌اش میدهیم. مثلاً افرادی که میان برای مصاحبه اگه توی همون جلسه اول کارفرما را جذب کنند، کارفرما این آدم حتماً کارش هم درست است، حتماً آدم مرتب و منظمی هم هست، حتماً کاربلد هم هست. یعنی ویژگی رفتاریش را به ویژگی های مهارتیش که ممکنه لزوماً درست نباشد وصل کردیم ولی واقعیت این است که اسیر این اثر هاله ای هستیم.

Scability : یعنی ما بتوانیم کسب و کارمون اگر در این مقیاس به خوبی کار می‌کند، اگر مقیاس کسب و کار بزرگتر شود، با افزودن یک سری زیر ساخت ها به درستی کار کند.

مثلاً اسنپ توی یک شهری خیلی خوب کار می‌کند، حال قرار است تعمیم به شهر های دیگر بدهیم. قرار نیست برویم خودرو بخریم، محصولمان و یا نرم افزارمان را عوض کنیم. کافی ست زیر ساخت سیستم یا سیستم را طوری تغییر بدهیم که اپلیکیشن ما با همان ویژگی ها روی این سیستم گونه ای کار کند که وقتی به هزاران کاربر پاسخ میداد حال بتواند به ۲۰ هزار کاربر پاسخ دهد. در استارتاپ ها خیلی با این موضوع سروکار داریم.

Security: اینکه سیستم خود را از نفوذهایی که بصورت غیر مجاز وارد سیستم شوند، محافظت کند و هزینه های مستقیم و غیر مستقیم ای که در سیستم هست باید در System Requirement دیده شود.

تکنیک های جمع‌آوری نیازها

۱. Team-based
۲. Individual

Team-based: به صورت تیمی باید کار را پیش ببریم. تعدادی کاربر را جذب می‌کنیم. مثلا در بیمارستان یک کارمند بخش داروخانه است. یک نفر از رادیولوژی ست، یک نفر از بخش پرستاری ست و... این افراد داخل یک تیم جمع می‌کنیم و سوالاتی از آنها می‌پرسیم و مصاحبه های گروهی انجام می‌دهیم و برای هر کدام ازاینها یک نقش تعریف می‌کنیم. جلسه هم می تواند بصورت formal و informal اداره شود و با استفاده از ابزار ارائه شود.

IBM یک ابزاری را برای این کار توسعه داده که یک تعداد کامپیوتر در شبکه قرار دارند و نرم افزاری روی این کامپیوتر ها نصب هستند و هر فردی پشت یکی از این کامپیوتر ها می نشیند و تحلیلگر ارشد یک سوال را مطرح می‌کند و همه در سیستم هایشان می‌بینند و می‌توانند بصورت شفاهی و تایپی پاسخ دهند.

مثلا اگه مدیر کنار ما نشسته باشد و من کارمند بخواهم حرفی بزنم که به مذاق مدیر خوش نیاید می‌توانم بنویسیم و هدف این است که سیستم را تا جایی که ممکن است بهتر و بهتر بشناسیم و مشکلات آن را استخراج کنیم.

این روشی که IBM توسعه داده و ما افراد را کنار هم بچینیم را JAD (Joint Application Development) می گوییم.

مزایا و معایب JAD:

نسبت به روش های مرسوم و سنتی که مصاحبه ی فردی انجام می شود و ما با تک تک افراد حرف می زنیم، هزینه بیشتری می برد.

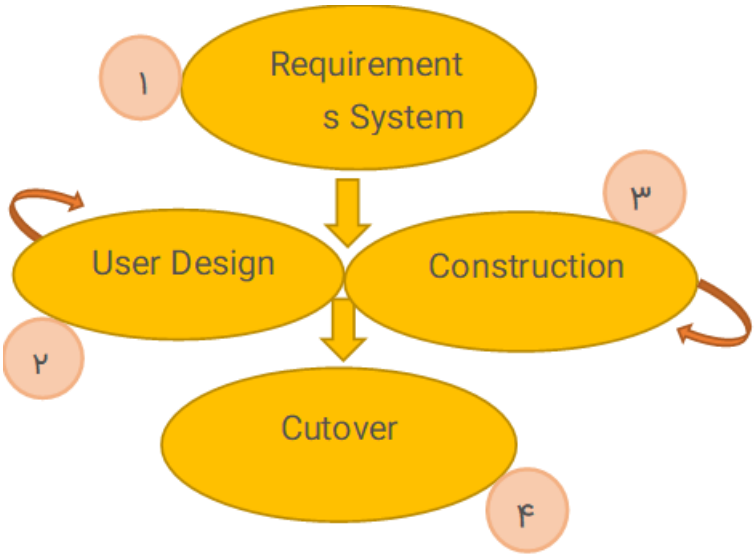
مدیریت کردن گروه های بزرگ کار سختی است.

اجازه میدهد که کاربران بخش های مختلف که اتفاقاً در پذیرش سیستم، در ارائه نیازمندیهای سیستم مهم و موثر هستند، حضور داشته باشند و مشارکت کنند.

کاربران سیستم احساس می‌کنند خودشان مالک این سیستمی هستند که قرار است نوشته شود، بنابراین اهمیت پیدا می‌کند که مشکلاتشان را مطرح کنند.

معمولا نیازمندیها در این روش به صورت دقیق تری بیان می شود، چون اگر تضادی وجود داشته باشد، مدیریت شود. مثلا کارمند۱ می گوید نیازها فلان و فلان است و کارمند ۲ ممکن است بگوید نه و این دو ممکن است با هم بحث کنند و ما نظاره گر هستیم تا به یک اجماعی می‌رسند. بنابراین چالش Imprecision بهتر مدیریت می شود.

Rapid application Development:



مثلاً تحت عنوان مدل Prototyping توضیح داده شده که یکی از روشها تعیین کردن نیازمندی ها و جمع آوری مهندسی نیازمندی ها و جمع‌آوری اطلاعات روش Prototyping است که حالا روش RAD شناخته می شود.

یکی از نمونه ها مشابه شکل رو به رو می‌باشد.

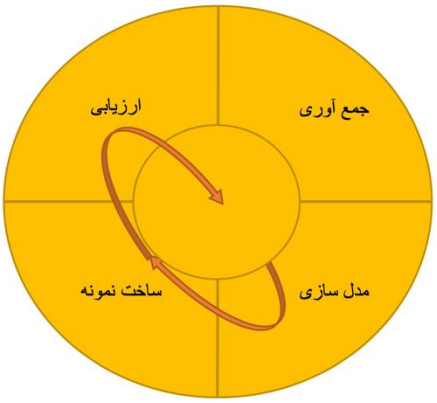
برنامه ریزی می‌کنیم نیازمندیها رو جمع آوری کنیم.

۱. یک طراحی اولیه از چیزی که ما تا الان برداشت کردیم به کاربران ارائه می‌دهیم.
   - و از کاربر feed back می‌گیریم. با آن تعامل
   - داریم و سعی می‌کنیم نمونه اولیه را بسازیم.
۲. نمونه اولیه را به کاربر نشون بدهیم.

این چرخه بین مرحله ۲و۳ تا جایی ادامه پیدامی‌کند تا روی بخشی که کار توسعه رو انجام می‌دادیم، کاربر ها بگویند درست همان چیزی است که ما می‌خواهیم.

یعنی روی نیازمندیها توافق حاصل شده باشد، حالا به سمت cutover می رویم.

یعنی در این قسمت محصول را پیاده سازی می‌کنیم، ساخته شد، به مرحله بعدی پروژه می‌رویم. اینارو توی cut های مختلف می‌بینیم.



مدل دیگری برای RAD قبلاً توضیح داده شده است.

مزایا و معایب Tram-based (RAD)

- توسعه سیستم را با سرعت قابل قبولی جلو می‌برد.
- تاکید زیادی روی نیازهای استراتژیک ندارد و خیلی متمرکز بر نظارت کاربران است، خیلی مهم است که کاربران بگویند دقیقا چیزی است که میخواهند و تا جایی اصلاح کنند تا مورد توافق طرفین شود. کاربرانی که نظر میدهند ممکن است خیلی به موضوعات راهبردی و استراتژیک سازمان توجهی نداشته باشند.
- به بحث کیفیت و سازگاری بیشتر توجه می شود و مهم است که سیستم کار کند.

روش های دیگرAgile : Team-based

ما در این روش ها سعی می‌کنیم توسعه محصول رو به صورت گام به گام داشته باشیم. به صورت مجموعه ای از Prototype ها ایجاد کنیم. این ها را دائماً به کاربران سیستم نشون بدهیم و از آنها feed back بگیریم و اصلاح را انجام بدهیم. روش های Agile ایده شان را از روش های RAD می‌گیرند.

Scrum: این روش یکی از معروفترین ها می‌باشد. در این روش معمولاً فرایندی تعریف می‌کنند. هر فرایند در ابتدای جلسه، افرادی که در تیم میخواهندکار کنند، یک warm up دارند.

یک جلسه ی کوتاه درباره ی اینکه امروز قرار است چه کار کنیم و task ها را مدیر از قبل تقسیم بندی کرده و آنها را به افراد میدهد و در قالب یک سری بلاک ها بصورت task هایی که در دوره های زمانی مشخصی تعریف شده اند، به برنامه نویس ها،طراح ها، تحلیلگران میدهد و این taskها را انجام میدهند و ما به کاربران نمایش میدهیم. یعنی کسانی که ناظر نهایی ما هستند و feed back می‌گیریم.

و در نهایت در تیم میاوریم و اصلاحاتی را انجام می‌دهیم و آنقدر این چرخه ادامه پیدا می‌کند تا ورژن نهایی نرم افزار تکمیل شود.

مزیتش این است که خیلی انعطاف پذیرست و برای تغییرات خیلی خوب کار می‌کند. یعنی مدیریت تغییرات را خیلی خوب می‌توانیم انجام بدهیم چون عملاً نمونه هایی از محصول را در جریان تحلیل سیستم، طراحی سیستم تولید می‌کنیم. این درست است که ممکن است همه قسمت ها لزوماً کار نکنند، اما نمونه ای است که توسط کاربرهای نهایی قابل دیدن می‌باشد.

اشکال این است که نیاز داریم که افراد تیم مهارتهای کافی داشته باشند، برای اینکه بتوانند بصورت تیمی کار کنند. و ابزارهای مختلفی که در Scrum مورد استفاده قرار می‌گیرد را به کار بگیرند.

برای جمع آوری نیازمندیها باید به این سوال جواب بدهیم که؛ چه کسی؟ چه کاری؟چه موقع؟ چطور؟ و کجا؟ در سیستم انجام میدهد و ما این اطلاعات را از روش های مختلف مثل مصاحبه مثل مطالعه اسنادی که آیین نامه ها و قوانینی که در سازمان وجود دارد مشاهده میکنیم و مستقیم کار کاربران یعنی در محل حضور داشته باشیم و ببینیم آنها چگونه کار میکنند و پردازش فرم ها و اطلاعات و گزارش هایی که تا الان در سیستم تولید شده دستی است یا کامپیوتری.

برای اینکه بتوانیم این اطلاعات را به دست آوریم یعنی بفهمیم چه کسی چه کاری را چطوری و چه موقعی و کجا انجام میده روش های مختلفی را میتوانیم به کار بگیریم:

The interview process

برای روش مصاحبه ما میخواهیم با یک سری از افرادی که تاثیر گذار هستند و اطلاعات کافی دارند، در جریان نیاز های سیستم هستند و در جریان اینکه سیستم چگونه کار میکنند هستند صحبت کنیم.

برای این کار باید به یک سری موارد توجه کنیم:

افرادی که میخواهیم با آنها مصاحبه کنیم را مشخص کنیم. برای همین باید درکی از سازمانی که میخواهیم تحلیل کنیم داشته باشیم.ما در مهارت های مورد نیاز تحلیلگر گفتیم که دانش سازمانی یکی از مهارتهایی هست که در تفکر سیستمی به کار ما میاید. برای دانش سازمانی باید بفهمیم که چندتا معاون و رییس دارد و چه معاون هایی ! هر کدوم ازاین معاون ها چند مدیر دارند هرکدوم از این مدیرها چه نیروهایی زیر دستشان کار میکنند و وظایف هر بخش چیست؟

به همین خاطر داشتن چارت سازمان یا دانش سازمانی که به دست میاریم کمک میکند که یک تعداد افراد رو از طیف های مختلف انتخاب کنیم و برنامه ریزی کنیم که با آنها مصاحبه کنیم و اطلاعات راجع به سیستم کسب کنیم.

نکته ی مهم این است که ما قبل از مصاحبه نگوییم که قرار است با این افراد مصاحبه کنم پس بریم اتاقشان و صحبت کنیم! این مصاحبه ی غیر روشمند است.

باید برای مصاحبه بگوییم که هدف من از مصاحبه با فرد شماره ۱ چیست؟ با فرد شماره ۲ چی؟ و.... پس باید سوالات از قبل مشخص شده باشد و یک جا یادداشت کنیم. حتما با کسی که میخواهیم مصاحبه کنیم وقت بگیریم.

اگرچه گفتیم سوالات مشخص است ولی موقع مصاحبه ممکن است یک سری سوالات جدید هم به ذهنمان برسد و هم اینکه نکاتی اضافه شود که سبب شود سوالات بیشتری نسبت به مجموعه سوالاتی که آماده کردیم به ذهنمان برسد.

برای مصاحبه آماده باشیم. مصاحبه را سعی کنیم هدایت کنیم. اگر سندی فایلی گزارشی چیزی لازمه همراه خودمان داشته باشیم و حتما از قبل آماده کنیم که کمترین زمان را برای مصاحبه صرف کنیم و سعی کنیم متن مصاحبه را بنویسیم.

تحت عنوان مستند سازی متن را پیاده سازی کنیم و مصاحبه را ارزیابی کنیم. آیا با این مصاحبه‌ای که انجام دادیم به خواسته ی مان رسیدیم؟! آیا اطلاعاتی که راجع به پرستاری بیمارستان داشتیم و سوالاتی که درباره اش داشتیم به آنها پاسخ داده شد؟ سوالاتی که درباره‌ی داروخانه داشتیم پاسخ داده شد؟ و....

چطور افرادی که میخواهیم مصاحبه کنیم را از قبل انتخاب کنیم؟

- از سطوح مختلف مدیریت و... افرادی رو انتخاب کنیم وقت بگیریم و صحبت کنیم.
- به صورت رندوم یک سری کاربران رو دعوت کنیم.
- مصاحبه فردی یا گروهی باشد.
- سعی کنیم کاندیدهایی که انتخاب می کنیم از همان کاندید هایی باشند که از بخش های مدیریت و کاربری و... حتی اگر سیستم قرار است با مشتری های خارج از مجموعه کار بکند، مثل بیمارستان بانک و...بهتر طیفی از مشتریان هم مورد توجه قرار بدهیم.
- اهداف مصاحبه را مشخص کنیم.
- اینکه انتظار داریم بعد از مصاحبه چه اطلاعاتی را بدست آورده باشیم.
- سوالات را از قبل آماده کنیم.
- خیلی مهم است که چگونه سوال بپرسیم و چه بپرسیم!
- از پرسیدن سوالات خیلی کلی مثل: چه خبر اجتناب کنید.
- سوالات تشریحی (سوالات که پایانشون باز هستند مثلا نحوه ی پذیرش بیمار در بیمارستان به چه ترتیب هست. فرد شروع به توضیح دادن میکند، این open ended question می شود.)
- سوالات انتخابی یا چند گزینه ای (تستی – غلط x)سوالاتی که گزینه ها رو از قبل انتخاب کردیم پاسخ محدود هست و باید یک یا چند تا ازین گزینه ها رو انتخاب کنند close ended question
- چگونه خود را برای مصاحبه آماده کنیم؟
- آماده سازی یک کار ضروری است.
- وقت مصاحبه را حتما از قبل مشخص کنیم از چه زمانی تا چه زمانی؟ (حداکثر یک ساعت باشد)
- دائم زمان را چک کنیم ببینیم آن فرد وقت دارد و آزاد است؟ استرس نداشته باشد و چک کنیم از آن فراتر نرویم.

- مکان مصاحبه را مشخص کنیم.
- طول زمان را مشخص کنیم.
- اینکه بگوییم میخواهم درباره سیستم صحبت کنیم عبارت درستی نیست. موضوع را مشخص کنیم یعنی بگوییم میخواهیم درباره فرایند پذیرش صحبت کنیم. یا اینکه در داروخانه چه کارهایی انجام میدهید.
- موضوع مشخص کنید و حتی خیلی کمک کننده است که سوالات را برای اون شخص ارسال کنیم قبل از مصاحبه اجازه بگیریم و در طول مصاحبه صدا را ضبط کنیم مستند سازی انجام بدهیم.
- به صحبت های کسی که صحبت میکند واکنش نشان بدهیم. نشان بدهیم که حرف هایش برای ما مهم است.

در جریان خود مصاحبه یک برنامه داشته باشیم:

- مثلا بگوییم در لحظه ای که من میخوام با فرد مصاحبه کنیم۱۰ دقیقه اول فلان بعد ۱۰ دقیقه ی بعدی فلان ۵ دقیقه بعدی فلان و.... یعنی یک برنامه برای مصاحبه داشته باشیم.
- در ابتدا خود را معرفی کنیم پروژه را توضیح بدهیم و اهدافی از مصاحبه دنبال می کنیم را بگوییم.
- شنونده ی فعال باشیم نشان بدهیم که صحبت های فرد برای ما مهم است.
- برای اینکه فکر کند و پاسخ بدهد به فرد وقت بدهیم.
- بعد از مصاحبه اطلاعات مصاحبه را خلاصه سازی کنیم، پیاده سازی کنیم که چیزی از قلم نیافتد.

مستند سازی مصاحبه:

- زمان مصاحبه حداقل مورد نیاز باشد.
- اگر فکر میکنید مستند سازی در طول برنامه زمان بر هست میتوانیم اجازه بگیریم و ضبط کنیم و بعد پیاده سازی و مستند کنیم.
- در صورت نیاز می شود با هماهنگی خود فرد چیزی که پیاده سازی کردیم را از مصاحبه برای خودش هم ارسال کنیم تا مورد وارسی و ارزیابی قرار بگیرد و در انتها مصاحبه را ارزیابی کنیم. یعنی بگوییم اهدافی که از مصاحبه برسیم را با سوالات رسیدیم یا نه.

روش های دیگه برای جمع آوری اطلاعات:

- داکیومنت ها را مطالعه کنیم. بالاخره هر سازمانی یک سری شرح وظایف برای کارمندانش دارد، یک سری آیین نامه،یک سری قوانین دارد که این ها را باید مطالعه کنیم.
- Observation:
- خودمان در محیط قرار بگیریم و طبعا با هماهنگی و اجازه ببینیم که مثلا در بخش‌های داروخانه چطور یک بخش بیمارستان اطلاعات را به صورت دستی یا سیستمی ارسال میکنند؟؟ داروخانه نسخه را آماده میکند و به بخش های مختلف بیمارستان بفرستد یا مثلا در بانک ببینیم کار تسهیلات بانکی چگونه انجام می شود؟ وام چطور است؟ افتتاح حساب چطور است؟

Questionnary (پرسش نامه یا پیمایش):

- مجموعه سوالات را به یک تعداد از کاربران بدهیم. معمولا سوالات close ended هستند ولی این امکان هم وجود دارد که یک سری open endedباشد و کاربران نظراتشان را بدهند.
- مقایسه مصاحبه و پرسش نامه:
- در پرسش نامه میتوانیم از تعداد بیشتری از کاربران اطلاعات بگیریم ولی تعداد سوالات محدود هستند. اما در مصاحبه یک تعداد سوال از قبل آماده کردیم اما ممکن است در طول مصاحبه در جریان مصاحبه سوالات جدیدی به ذهنمان برسد و عملا اطلاعات بهتر و دقیق تری از سیستم بدست میاوریم.
- چیزی که در انجام تحلیل پروژه های نرم افزاری معمول است، ترکیب روش هایی ست که میگوییم. یعنی از پرسش نامه و مصاحبه و هم مطالعه داکیومنتری استفاده میکنیم. همobservation داریم و هم از روش brain stormating استفاده میکنیم.

طوفان ذهنی – ذهن انگیزی Brain stroming

یعنی ما گروهی از افراد را درست کنیم و بگوییم که شما هر ایده ای راجع به موضوع دارید مطرح کنید. ما این ها را روی یک تابلو مینویسیم و بعد شروع میکنیم ایده ها را بر اساس نظر تعدادی دیگر که در مجموعه هستند امکانش هست خط بزنیم – اصلاح کنیم – دقیق کنیم و....

میتواند به صورت unstructured , structured باشد:

Structured روش مند و مشخص باشد. چه کسانی چه حرفایی را باید بزنند و راجع به چه موضوعاتی باید نظر بدهند و گامها را از قبل مشخص کرده باشیم.

در جمع آوری داده ها لازم است با مفهوم sompling هم آشنا باشیم.

Sompling (نمونه گیری):

در آمار خواندیم وقتی میخواهیم تحقیقی انجام بدهیم مثلا یک نظرسنجی راجع به اینکه فلان امکاناتی که در دانشگاه فراهم شده خوب انجام می شود یا بد.

- یعنی بیایم یک تعداد از دانشجویان را انتخاب کنیم و از آنها سوال میپرسیم که میتوان به صورت مصاحبه و هم به صورت پرسش نامه پرسید. اینکه چگونه انتخابشان کنیم مربوط به بحث sompling می شود.
- اگه همه را بگوییم که شرکت کنند systematic sample می شود.
- اگه تعدادی را بر حسب کد پستی انتخاب کنیم یا برحسب شماره دانشجویی یا بر حسب اینکه ۳ نمونه دانشجو مربوط به یک ورودی هستند بحث (startfield sample)(نمونه گیری هدفمند – طبقه بندی شده) می شود.
- میتوانیم رندوم (Random sample)انتخاب کنیم. یکی را میفرستیم در کانال دانشگاه و هر کسی جواب داد.
- یا اینکه میتوانیم نمونه ای که میخواهیم انتخاب کنیم کاملا هدفمند باشد و یک گروه خاصی را مد نظر هست را انتخاب کنیم. (objective of sample)

یک دسته از روش هایی که میتوانیم برای جمع آوری اطلاعات استفاده کنیم روش research (پژوهش کردن)است.

مثلا اینترنت را چک کنیم که کلا سازوکارهای (مثال ما HIS بوده. سیستم های اطلاعات بیمارستانی.) ببینیم چه بیمارستان هایی در کدام کشورها چه کارهایی کردند مجلات IT را دنبال کنیم. کتاب ها را دنبال کنیم. اگر راه حلی توضیحی دادن نکته ای را گفتند راجع به آن و اطلاعاتی را راجع به موضوعی که داریم کار میکنیم از اینترنت و سایت های مختلف شرکتایی که این کارو انجام دادند به دست بیاریم.

Agile

قبلا گفتیم که هدف و تمرکز روی این است که (مخصوصا روش xp)روی روایت هایی از کاربران از سیستم برای ما تشریح می کند. ما روایت ها را بشنویم سناریو های مختلف را بشنویم سعی کنیم این سناریو ها را کنار هم بگذاریم حالت های مختلفی که سیستم احتمالا در آن وضعیت قرار میگیرد را استخراج کنیم و ویژگی هایی که نیاز هست تا این سیستم خوب کار کند را دربیاوریم و به اطلاع کاربرامان برسانیم و آنها مورد ارزیابی قرار بدهند.

تا اینجا فرض بر این است که ما اطلاعات را به روش های مختلف جمع آوری کردیم. حال میخواهیم سازمان دهی کنیم مستلزم این است که یک زبان مدل سازی داشته باشیم روش های مختلفی برای این کار وجود دارد.

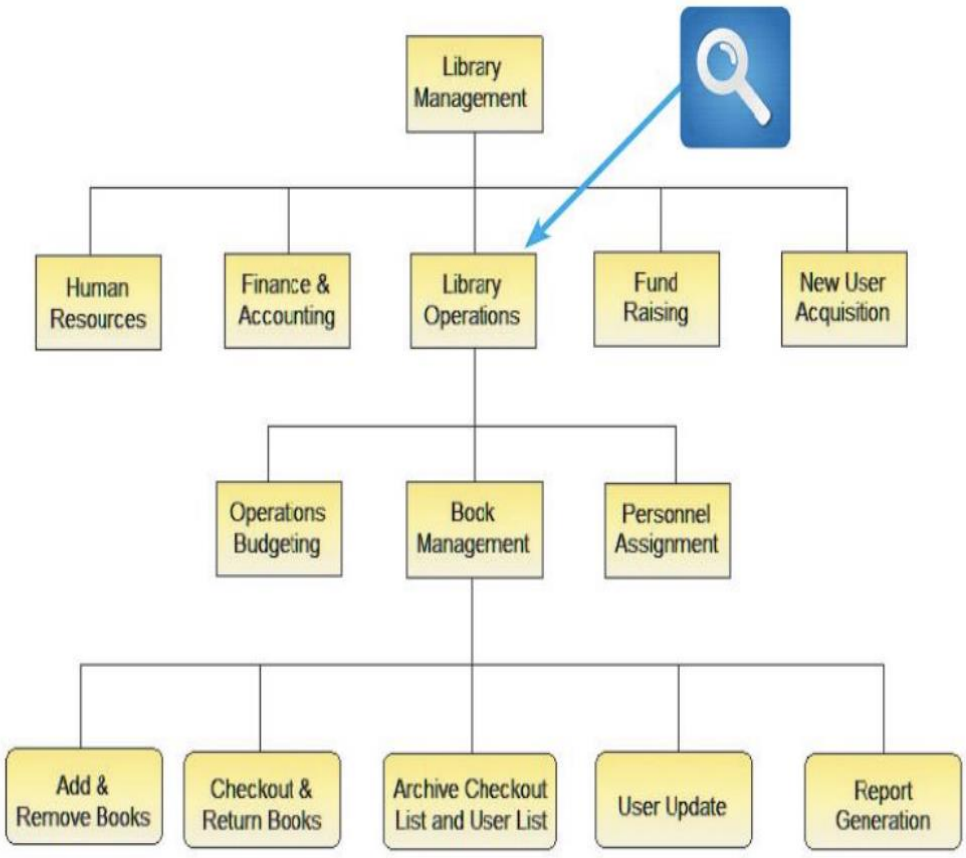
استفاده از زبان طبیعی:

برای اینکه اطلاعاتی را که جمع آوری کردیم را Represent کنیم.

Diagram:

از دیاگرام ها استفاده میکنیم. مثلا در فلوچارت که در مبانی کامپیوتر خونادیم که برای شروع فلوچارت از دایره استفاده میکنیم. برای انجام محاسبات از مستطیل برای خواندن از متوازی الاضلاع و... کاری که اینجا انجام میدادیم مدل سازی یا representation هست.

- در مورد مستند سازی هم گفتیم که سعی کنید اطلاعاتی که به دست میاوریم را ثبت کنیم که این ثبت میتواند ضبط صوتی باشد و یا به صورت تایپ باشد.
- از روش های ساده استفاده میکنیم که اطلاعات را نگه داریم.
- سعی کنید اطلاعات را گونه ای بنویسیم که قابل فهم باشد.
- اطلاعات را سازماندهی کنیم چه به صورت فنی چه با استفاده از زبان های مدل سازی بنویسیم.



در این مثال نمایی از اینکه سیستم library managemet شامل چه زیر مجموعه هایی هست بعد این زیر بخش ها شامل چه فعالیت هایی هستند.

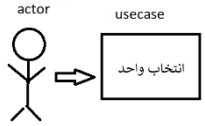
قسمت های مختلف باید اطلاعاتش را بدست بیاروییم و میاییم توضیح میدیم که مثلا فعالیت شماره ۱۴ که ورودی هایش فلان است و خروجی اش هم فلان توسط فلانی انجام شود.

نمودار فرایند کسب و کار:

مثلا توی مثال صفحه ی قبل user update یک فعالیتی است که در سیستم به این نوع فعالیت ها process یا فرایند میگوییم.

Work flow هم گفته می شود (جریان کار).

مثلا انتخاب واحد یک جریان کار است. دانشجو اول در سیستم log in میکند بعد بخش ثبت نام شماره ترم را وارد میکند لیست دروس را میبیند بعد کد درسها را جایی وارد میکند بعد ثبتشان میکند و بعد نهایی شان میکند این فرایند برای اینکه بتوانیم نمودار فرایند را نشان بدهیم.یکی از روشها bpmn است و روش دیگر data flow diagram است و روش دیگر .uml



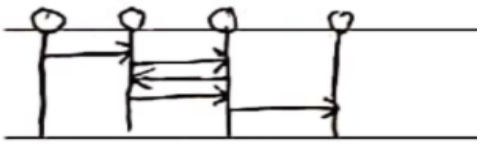
Use case diagram زیر مجموعه umlاست.

اینکه کاربر(در uml به او میگویند actor) از چه سرویسی استفاده میکند یا ارائه میکند. مثلا سرویس انتخاب واحد یک نمودار میکشیم و نشان میدهد که به موضوع از این زاویه نگاه میکنیم که سیستم ما چه سرویس ها و قابلیت هایی را ارائه میکند یا چه کسانی از این سرویس استفاده میکنند.

هستumlهم زیر مجموعه Sequence diagram

توالی انجام کارها در سیستم نشان میدهد و تعاملی که بین اشیا وجود دارد.

مثلا شی ۱ ۲ ۳ ۴ شی ۱ اول یه پیام میفرسته به شی ۲ بعد ۲ به ۳ و بعد ۳ به ۲ و بعد ۲ به ۳ و.... این توالی انجام کار را استفاده میکنیم.Sequence diagram میخوایم نشون بدیم



Validating and verifying

مرحله اخر، نیازمندی هایی که استخراج کردیم را validate کنیم. خیلی مهم است که اطمینان حاصل کنیم از اطلاعاتی که به دست آوردیم و مدلی که تهیه کردیم.این اطلاعات اطلاعات درستی است!! این چیزیی ست که مجموعه ی مدیریت سیستم را میخواهد، باید پیاده سازی کند توی این مرحله لازمه آنها را validate کنیم که اطلاعات درست است؟ و ما به درستی بیان کردیم؟

ابزارها Tools

ابزارهای مختلفی وجود دارد و ما به عنوان مهندس حتما از مجموعه ابزارها باید استفاده کنیم. ابزارهایی که برای گام جمع آوری نیازها وجود دارد. word مثلا برای مستند سازی spreadsheet software ها مثل اکسل که بتوانیم داده ها را در آن بگذاریم. ابزارهایی که برای بهره وری استفاده میشوند. ابزاری که برای مدیریت اطلاعات فردی استفاده می شود. مدیریت و زمان بندی و مصاحبه ابزارهایی برای مدیریت دیتابیس ها ابزار هایی برای نمایش مدل گرافیکی ابزار هایی که برای بحث های تعاملی استفاده می شود. اینکه ما کار مدل سازی و مستند سازی را کار مدیریت مصاحبه و طراحی پرسش نامه و... این ها را با استفاده از ابزارها انجام بدهیم.

مروری بر کارهای استارت آپی

مدل کسب و کار و توسعه مشتری:

استارت آپ ها بر حسب یک ایده شکل میگیرند ایده ای که ممکن است در ذهن یک فرد به عنوان موسس این استارتاپ شکل بگیرد و بعد شروع به پیاده سازی این ایده را میکنیم و تبدیل به یک نرم افزار میکنیم ولی یادمان باشد که در طرف دیگر مشتریان نهایی هستند که قرار است از محصول ما استفاده کنند.

بنابراین اگر من یک چیزی را در ذهن خودم توسعه بدهم وقتی به مشتری عرضه میکنم عملا مشتری بگوید نه این چیزی نبود که من میخواستم،**من موفق نشدم**!

مدل کانواس یکی از روش های توسعه بیزینسی هست: ما یک مفروضات و حدس های اولیه را نوشتیم ولی سوال اصلی این است که چطور مفروضات را به fact تبدیل کنیم.یعنی مطمئن شویم این اطلاعات که ما داریم اطلاعات درستی هست؟ و به همین خاطر لازم است که به بیرون از شرکت برویم و با یک سرس مشتریان مصاحبه کنیم واین حدسیات و مفروضات خودمان را عملا به fact تبدیل کنیم.

ما در جریان تکمیل اطلاعات تاکید داشتیم که روی چه کسی؟ چطور؟کجا؟چگونه؟چه چیزی؟ که باید این اطلاعات را در بیاوریم برای همین فرآیندCDP یا Customer Development Process در حوزه ی کسب و کار تعریف می شود. این فرآیند در استارت آپ ها بسیار مهم است.یکی از دلایل شکست استارت آپ ها این است که به این موضوع کمتر توجه کرده اند.از نظر فنی شکست نخورده ولب به این فکر نکرده که مشتری چه چیزی میخواهد.یعنی وقتی محصول را به بازار عرضه میکند مشتری استقبال نمیکند به خاطر این که به این قضیه از قبل فکر نکرده اند.

CDP فرآیندی است که طی میکنیم تا مفروضات مان را درمدل بیزینسی به یک سری fact تبدیل کنیم.

معمولا دو مرحله ی کلی را میتوانیم متصور شویم که هر کدام دو مرحله است که کلا چهار مرحله می شود search و execution.ما باید یک ساختار روشمند داشته باشیم تا یک تجربه را طراحی کنیم، آزمایش کنیم و نسبت به آن داده به دست بیاوریم که این داده ها بحث جمع آوری اطلاعات میشوند.

بنابراین چقدر مهم است ما اصول پژوهش را یاد بگیریم؟چطور مسئله را تعریف کنیم؟با چه روش هایی آزمون کنیم؟ مثلاً یکبار یک ایده را در ذهنمان مطرح میکنیم و یک سری داده جمع کنیم و می گوییم بله! مشتری های من به شدت نیاز دارند به این محصول و حتما موفق می شوم و شروع میکنیم محصول راDevelop می کنیم اما بعد که می آید بیرون میبینیم که کسی نمیخرد دلیلش را بررسی می کنیم و می بینیم احتمالا در بخش فنی هست یعنی همان تصوری که از اول داشتیم در محصول نهایی وجود دارد اشکال میتواند اطلاعاتی که بدست آورده‌ایم و بعد بر اساس آن ها قضاوت کردیم که قطعا مشتری از ما خواهد خرید باشد که اصلا روش عالی ای نبوده، یعنی آن روش پژوهش را درست انجام ندادیم.به همین خاطر برای اینکه یک کسب و کار موفق شود احتمالا مجموعه ای از افراد دارند کار میکنند و مهارت های مختلفی را باید مسلط باشند.

مرحله جستوجو و اجرا

مرحله اول توسعه مفروض است،اینکه درمورد مشکل مشتری چه فکری میکند و نیازهایش چه چیزهایی هستند و ما جواب این ها را بدهیم،customer discovery و customer validation است.

مجموعا قرار است در این مرحله ببینیم که آیا آن ایده ای در ذهن ما هست با آن ایده ای در ذهن مشتری هست مطابقت دارد یا نه؟به این مرحله Product Market Fit می گوییم.یعنی میخواهیم مطمئن شویم که بازاربه آن نیازما fit هست پس customer discovery و customer validation مرحله ی search می شود.

در search لازم است با مشتری حرف بزنیم و آن ها را بشناسیم و دسته بندی کنیم و بفهمیم که چه شخصیت هایی دارند و چه ویژگی هایی دارند و نیازهایشان را بهتر بشناسیم و اصلا آن چیزی که به عنوان مشکل فکر

کردیم از نظر مشتری هم مشکل به حساب می آید؟! اگرهست این مشکل چقدر برایش دردسر ایجاد کرده؟اگر دردسر ایجاد کرده چقدر حاضر است هزینه کند تا ابزار یا دستگاهی بخرد که مشکلش برطرف شود؟ و این حرف ها را در یک چرخه تکرار میکنیم و تا جایی این جستوجو را ادامه میدهیم تا مطئن شویم که بخش زیادی از مشتری هایمان را شناسایی کردیم و آن ها را مورد ارزیابی قرار دادیم ومطمئن شویم که ایده ی ما به کارشان می‌آید.

بخش دوم، customer creation و company building بخش اجراست.

در customer creationعملا می آییم اطلاعاتی که از مشتری جمع کردیم را دسته بندی میکنیم، ویژگی هایشان را طبقه بندی می کنیم ومشخص می کنیم که هر نوع مشتری چه نیازهایی دارد و مشکلاتشان چیست؟چقدر هزینه می کنند؟حقوق ماهیانه شان چقدر است؟!

درcompany building شروع به تقویت تیم و پایه گذاری شرکت،بخش توسعه فنی و بازاریابی و فروش و... می کنیم.

فصل ۵

فرض کنید اطلاعات را جمع آوری کردیم این اطلاعات احتمالا به یک حجم زیادی کاغذ که در آن مقدار زیادی آیین نامه و قانون و روش انجام کار و شرح وظایف بخش های مختلف می باشد، تبدیل شدند.

در مهندسی همیشه دنبال این هستیم که بتوانیم کار ها را با قابلیت اطمینان بالا و هزینه کم انجام بدهیم، مدل سازی تکنیکی است که به مهندسان کمک می کند تا بتوانند یک درک روشن از اطلاعاتی که تا کنون به دست آوردند یا راه حلی که طراحی کردن به دست بیاورند.

اهداف فصل:

بعد از این درس ما باید بتوانیم:

- رابطه بین مدل های Logicو مدل های Physical را توضیح بدهیم. (بگوییم منظور از مدل منطقی و مدل فیزیکی چیست)
- Data Flow Diagram: (نمودار جریان داده)
- توضیح چهار علامت استفاده شده درون نمودار داده
- فهم خطوط راهنمایی استفاده شده برای طراحی نمودار
- ترسیم نمودار بالاترین سطح یا نمودار زمینه ای، نمودار سطح صفر را بکشیم، نمودار سطوح پایین تر در DFD را بکشیم.
- Context Diagram: ۰_level، ۱_level، ۲_level و... به سمت جزییات بیشتر حرکت می کنیم یعنی به راه حل جزییات بیشتری اضافه می کنیم.
- ویژگی تعادل در سطوح مختلف: به طور مثال سطح دوم و سطح سوم (۳_level , ۲_level)باید باهم در تعادل باشد و همین طور سطح دو با سطح یک و الی آخر.
- با ابزار هایی که فرآیند را توضیح می دهند آشنا می شویم.

مدل سازی

مهندسی: تکنیکی هست که ما یک فرآیند را دنبال می کنیم. مجموعه ای از گام ها، که یک محصولی را توسعه بدهیم (در نرم افزار این گام ها را توسعه می گوییم و در صنایع دیگر تولید گفته می شود.)این محصول باید دو ویژگی داشته باشد:

۱- تا حد ممکن قابلیت اطمینان به محصول بالا باشد.
۲- هزینه تولید و نگهداری پایین باشد.

برای این کار از تکنیک های مختلفی استفاده می کنیم که یکی از آن ها مدل سازی است.

مدل سازی چیست؟

تعریفی که از مدل ارائه می شود؛

مدل: ساده سازی واقعیت، یعنی ما هر آن درکی که از محیط بیرون داریم را در قالب یک مجموعه‌ای از علامت ها نشان می دهیم. مدل صورت های مختلفی دارد یعنی وقتی می گوییم

E= mc^2

$$sim(q.d_j) = \frac{\sum_{i=1}^{n} w_{i,j} * w_{i,q}}{\sum_{i=.}^{n} w_i * j^2 * \sum_{i=.}^{n} w_i * q^2}$$

این یک مدل است که از یک سری علائم استفاده می کنیم، مثلا $\sum$ یا W_{ij} یعنی وزن کلمه i ام در سند jام.

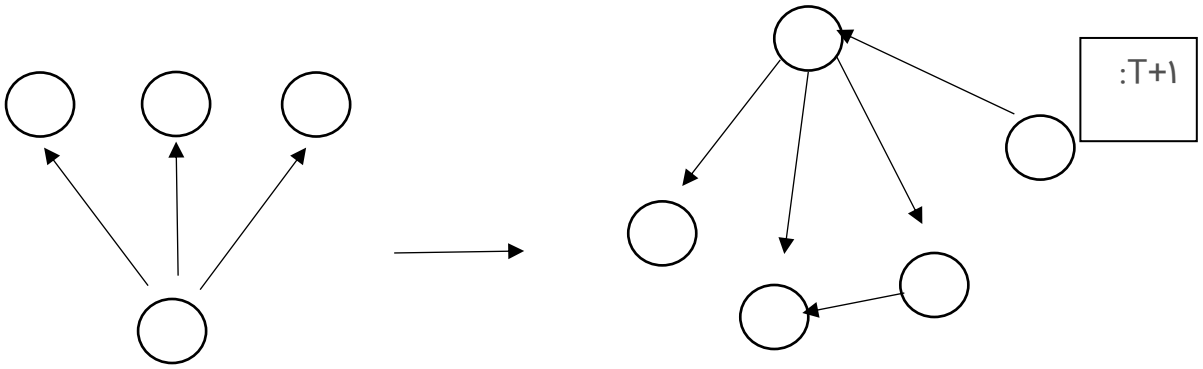
این یک شیوه مدل سازی است که یک واقعیت را در قالب زبان ریاضیات تشریح می کنیم.

در یک مدل دیگر از واقعیت ممکن است ماکت بسازیم. به طور مثال وقتی می خواهند یک مجتمع طراحی کنند با یونولیت نشان می دهند یا محل پارک و جاده و.... این مدل می تواند بیانگر آن چیزی است که هست یا آن چیزی ست که میخواهیم عملیاتی کنیم. مثلا راه حل ما باشد، اینجاست که مدل تحلیل یا مدل کسب و کار داریم. آن جا که راه حل را طراحی می کنیم مدل طراحی یا مدل راه حل گفته می شود.

ممکن است کامپیوتری (به وسیله اتوکد)یا با انیمیشن باشد.

مدل سازی محاسباتی

در این زمینه هم از کامپیوتر استفاده می کنند که در واقع یک مجموعه از علائم را بر حسب یک سری از قوانین و قواعد و ارتباطاتی که برایشان تعریف می کنند مثلا از حالت T به حالت T+۱ برود یعنی بگویند این شبکه ای که اینجا می دیدیم ساختارش عوض شده، اول به آن یک گره اضافه شد در ثانی از گره شماره یک به دو و سه و چهار یال وجود داشت حال یال ها تغییر پیدا کردند و ساختار شبکه عوض شده است.



پس مدل هر چیزی ست که از واقعیت سعی می کنیم بسازیم، چه آن واقعیتی که ما درکش می‌کنیم (ممکن است سوال پیش بیاید که چرا آن چیزی که وجود دارد را مدل می کنیم؟! به خاطر اینکه آن چیزی که وجود دارد پیچیدگی هایی دارد که ما برای اینکه آن را در تصویر درک کنیم از آن مدل می سازیم.)بیشتر اوقات برای اینکه مسئله یا مشکل را بفهمیم لازم است که یک مدل از آن بسازیم که مدل به ما کمک می کند درک یک باره از مشکل داشته باشیم.

همچنین ممکن است مدل ما مدل طراحی باشد یعنی راه حل را مدل می کنیم؛ پس مدل سازی می تواند در مراحل مختلف اتفاق بیافتد که ما در مهندسی نرم افزار حداقل از این دو مرحله استفاده می کنیم. حتی ممکن است مدل تست داشته باشیم، یعنی بگوییم که تست نرم افزار بر اساس چه ساختاری باید انجام شود.

چون هنوز به طراحی نرسیده ایم مدل هایی که از آن ها صحبت می کنیم مدل های مرحله تحلیل است، وقتی صحبت از مدل کسب و کار می کنیم یعنی کسب و کاری که الآن کار می کند به چه شکل است.

فرض کنید در حوزه نرم افزار هر کدام از سازمان ها، شرکت ها، فروشگاه ها و هرجایی که می خواهیم برایش نرم افزار بنویسیم را به شکل یک کسب و کار ببینیم.

مثال: فرض کنید برای یک رستوران یک نرم افزار طراحی می کنیم، اولین کاری که می کنیم این است که بفهمیم کسب و کار رستوران با چه منطقی انجام می شود، چگونه انجام می شود،) ممکن است بگوییم می دانیم مشتری به فروشنده مراجعه می کند، غذا برایش سرو می شود و هزینه آن حساب می شود و خارج می شود.)

اگر چنین تصوری از رستوران داشته باشیم تنها بخش کوچکی از کسب وکار را داریم توصیف می‌کنیم آن بخش که مواد غذایی تامین می شود و چگونه به دست رستوران می رسد؟ کجا انبار می شوند؟ تقاضای مواد غذایی روزانه است؟ یعنی هر روز مجموعه ای از مواد برایش می آیند یا انبار دارد و برای یک هفته انبار می کند؟ چگونه نظارت می کند که چقدر از کالا ها را داخل انبار دارد؟ این سوالات از زاویه دید مدیریتی بود. از زاویه دید صندوق دار به این صورت است که چگونه چک می‌کند تا الان چند نوع غذا فروش رفته است؟ از کدام نوع بیشتر فروش رفته؟ برای مشتری های بعدی غذا موجود است؟

اینکه از زوایای مختلف متوجه شویم چیزی است که اسمش را کسب و کار چیست، گذاشتیم. می شود همان اطلاعاتی که باید از یک کسب و کار بدست بیاوریم و بعد آن ها را در قالب مجموعه‌ای از علائم مدل کنیم. این مدل کسب و کار می شود، یعنی متوجه شویم کسب و کار چگونه کار می‌کند.

مدل منطقی(Logical model)

سیستم چه کار هایی را باید انجام دهد صرف نظر از اینکه ساختار پیاده سازی سیستم چه هست.

مدل فیزیکی(Physical model)

سیستم چگونه باید ساخته شود.

به عنوان مثال سیستم آموزش دانشگاه: در این سیستم یک مجموعه ای داریم مثلا مدرس، دانشجو، کلاس هایی که ارائه می شوند، بخش کارمند های آموزش و یک سری فرآیند هایی داریم مثل انتخاب واحد، حذف تک درس، سیستم از چه اجزایی درست شده و هر کدام از این اجزا چه ارتباطی باهم دارند و این افراد باید بتوانند از طریق یک سری دستگاه به یکدیگر متصل شوند و آن شبکه ای که وجود دارد و ارتباطی که بین بخش های فیزیکی سیستم ما وجود دارد بخش فیزیکی می شود و ما در این بخش قرار است توضیح بدهیم فیزیک سیستم چگونه است و یا چگونه باید باشد.

اما برای اینکه انتخاب واحد را دانشجو انجام دهد یک سری قواعد داریم اینکه دانشجو های ورودی چه سالی امروز نوبت آن هاست که انتخاب واحد کنند، شرط انتخاب واحد چیست؟ درس های قبلا گذرانده باشند، اگر کسی معدلش بالاتر از ۱۷ باشد می تواند بیشتر از ۲۰ واحد بردارد و... فرآیند اخذ واحد چیست؟ اینکه لیست دروس را می بینند بعد درس ها را انتخاب می کنند و به مجموعه دروس آن ها را اضافه می کنند، بعد اعمال تغییرات را می زنند، بعد چک می شود که این واحد ها مجاز هست یا نه بعد اتفاق می افتد.

این مراحل می شود منطق سیستم فارغ از اینکه با چه تکنولوژی پیاده سازی شود و چه افرادی با هم دیگر قرار است ارتباط داشته باشند تا این منطق اتفاق بیافتد.

- آن منطقی که ما برای اجرای فرآیند در نظر می گیریم مدل منطقی نامیده می شود.

ما حداقل ۴ دسته نمودار می توانیم تصور کنیم؛ دو دسته مربوط به سیستم فعلی است که به آن ها مدل تحلیل می گویند؛ و دو دسته هم مربوط به سیستم آتی است که به آن ها مدل طراحی می‌گویند.

مدل تحلیل: یعنی ما سیستم موجود را بشناسیم، اول باید متوجه شویم سیستم موجود چگونه کار می کند تا بعد مشکلات آن را در بیاوریم و آن را به یک سیستم جدید تبدیل کنیم و به صورت فیزیکی و منطقی سیستم را مدل می کنیم.

مدل طراحی: ما یک سیستم جدید پیشنهاد می دهیم.

اگر یک سیستم خوب کار کند و اجزایش خوب کنار هم چیده باشند و به صورت دستی کارش به درستی انجام شود که ما وقتی می خواهیم این را به سیستم جدیدی ببریم احتمالا تغییرات زیادی نمی خواهیم تنها کاری که می خواهیم انجام بدهیم این است که بخش ها را که قبلا به صورت دستی انجام می داد الان از طریق شبکه و کامپیوتر انجام شود، احتمالا سیستم جدید با سیستم قبلی خیلی تفاوتی نخواهد داشت اما اگر سیستم موجود مشکلات زیادی داشته باشد باید آن سیستم را تغییر دهیم، یعنی تحلیلگر سیستم آن ویژگی هایی که فصل قبل گفتیم در اینجا به کارش می آید، جایی که می خواهیم اشکالات سیستم موجود را حل کنیم و با آن یک سیستم جدید طراحی کنیم.

ممکن است هفتاد درصد سیستم قبلی تغییر کند و امیدوار باشیم که مجموعه سیستمی که طراحی کردیم یک سیستم کارآمدتری نسبت به قبل است.

آیا از زبان ریاضی می توان در مدل سازی کامپیوتر استفاده کنیم؟!

بله مثل زبان z با این زبان ها سیستم را توصیف و وارسی می کنند مثلا وقتی می خواهند سیستمی تحت عنوان Enroll برای ثبت نام دانشجو از یک سری علامت ها استفاده می کنند، به زبان ریاضی مثلا x student ∈و... ادامه می دهند که سیستم باید چه ویژگی هایی داشته باشد و چگونه عمل کند.

چه زبان های دیگری موجود است؟ notation راجع به DFD که صحبت شود نوع خاصی از علامت گذاری هست و یا مثلا در زبان UML – flowchart

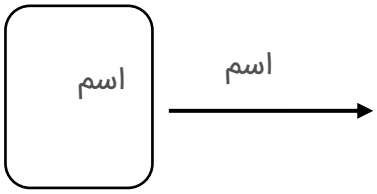
وقتی می خواهیم یک سیستم را مدل کنیم چند نکته برای ما مهم است؛ **زاویه نگاه**: ما می توانیم عینک های مختلفی به چشم بزنیم و سیستم را با آن ببینیم یکی از عینک ها که می توانیم بزنیم *Functional* است یعنی سیستم را به صورت مجموعه ای از فرآیند ها می بینم مثال: DFD

اگر عینک *Object Oriented* را بزنیم آنگاه سیستم را به صورت مجموعه ای از Object ها می بینیم، آنگاه زبان مدل سازی ما UML است.

ممکن است عینک *Agent Oriented* بزنیم آنگاه سیستم را به شکل مجموعه از Agent ها می بینیم و زبان مدل سازی Net Logo است.

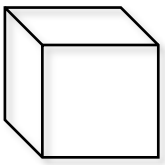
DFD یک نمودار جریان است که توضیح می دهد که دیتاها چگونه در این سیستم جریان پیدا می‌کنند، علامت گذاری های مختلفی دارد؛که در خود DFD افراد شروع کردند علامت گذاری معرفی کردند از جمله gane and sarson symbol و Yourdon چیزی که مرسوم است و امروزه استفاده می شود gane and sarson است.
برای مدل کردن process علامت آن:

Data Flow: یک پیکانی که روی آن اسم جریان داده نوشته شده‌است.



Data Store:

Entity: یک مستطیل سایه دار



Process: یک کار که در سیستم انجام می شود، حتما باید عملیاتی اتفاق بیافتد.

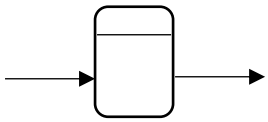
Data Flow: یک دیتایی که از نقطه A جابجا شود و به نقطه B می رود.

Data Store: همان دیتابیس (محلی که اطلاعات در آن نگهداری می شوند)است.

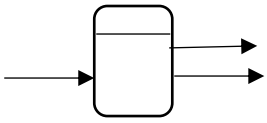
External Entity: موجویت های بیرونی هر چند در سیستم یکه اطلاعاتش برای ما مهم است درون سیستم نگهداری کنیم می شود موجودیت درونی و External می شود آن موجودیت هایی که خارج از Scope ما هست.

قوانین DFD:

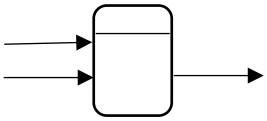
یک Data Flow وارد یک Process شود، پردازش روی دیتا انجام شود ویک خروجی به ما بدهد.



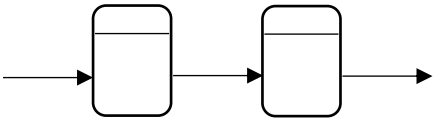
یک دیتا وارد شود پردازش انجام شود و دو دسته دیتا به ما بدهد.



دو دسته وارد و یک دسته خارج شود.

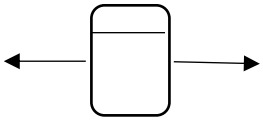


یک داده وارد Process شود و یک خروجی بدهد که خروجی ورودی Process بعدی بشود.

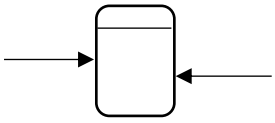


این موارد در DFD اشتباه است:

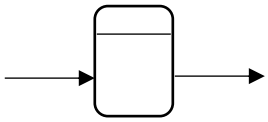
Miracle: جادو نمی کند، یعنی فرآیندی که ورودی نگیرد نمی تواند خروجی تولید کند، فرآیندی که فقط خروجی تولید می کند غلط است.



Black Hole: سایه چاله، فقط ورودی بگیرد و خروجی ندهد.



Gray Hole: یک سری داده هایی که کافی نیستند، از آن ها داده ای تولید کند که ارتباطی بین آن ها وجود ندارد. به طور مثال از روی تاریخ تولد یک فرد تاریخ مرگ او را نمی توانیم بگوییم.



Data Store: جاهایی که داده ها را نگهداری می کنیم، یک سری Table داریم که هر Entity که نرمال شده است را نگاشت می کنیم به این Table و در این Table یک سری Field داریم که مثلا: Name، ID و... و شروع می کنیم آن ها را پر می کنیم و عملا این Entity ها نگاشت می‌شوند.

می توانیم با DFD تنها سیستم را مدل کنیم؟!

خیر، ما DFD + ER Diagram می خواهیم؛ ER Diagram نشان می دهد که مدل معنایی داده هایی که در Data Store باید قرار بگیرند به چه ترتیب و چه شکلی باید باشد.

یک موجودیت باید با پروسه ای یا فرآیند در ارتباط باشد یا بلعکس، فرآیند با موجودیت در ارتباط باشد یا بصورت رفت و برگشتی باشد.

یک فرایند می تواند اطلاعاتی را به موجودیت بفرستد و موجودیت دوباره آن اطلاعات را به فرایند بفرستد.

مواردی که اشاره کردیم **نمونه های درست** از مدل سازی می باشد.

نمونه های غلط

یک موجودیت نمی‌تواند با موجودیت دیگری در ارتباط باشد یعنی یک موجودیت نمی‌تواند اطلاعاتش را به دو موجودیت دیگر مستقیماً بفرستد.مثل این که مشتری اطلاعات چیزی را که میخواهد را مستقیما به صندوق دار نمی‌فرستد،بلکه از طریف فرآیندی به صندوقدار یا آشپزخانه یا مدیر ارسال می شود.

یک موجودیت با یک data sture نمی‌تواند ارتباط مستقیم داشته باشد، یک data sture با موجودیت نمی تواند ارتباط مستقیم داشته باشد.

یعنی چه؟ فرایندی بین این دو قرار بگیرد مثلاً فرایندی که اطلاعات دریافتی از موجودیت یک را پردازش می‌کند و بعد به موجودیت دو می فرستد یا از موجودیت می‌گیرد اطلاعات را پردازش می‌کند به data sture می‌دهد یا از data sture اطلاعات را می‌گیرد پردازش می‌کند به موجودیت می‌دهد.

قوانین کلی

فرآیند با فرآیند ، فرآیند با موجودیت(درDFD)، فرآیند با data sture امکان پذیر است.

موجودیت با موجودیت، موجودیت با data sture، data sture با data sture امکان پذیر نیست.

DFD

ما مدل گرافیکی که بر اساس اطلاعاتی که از سیستم جمع‌آوری کردیم را آماده می‌کنیم مدل سازی مثل یک زبان برنامه نویسی نیست خیلی مشخص نیست که بگوییم حتماً دستور پرینت فلان مدل نوشته شود و... بنابراین ما به تعدادی خطوط راهنما اکتفا می کنیم.

خطوط راهنما

اولین کاری که DFD انجام میدهیم بعد از اینکه اطلاعات را جمع‌آوری کردیم سعی می‌کنیم یک context diagram رسم کنیم، context diagram در واقع کلی‌ترین و کلانترین زاویه ی ما با سیستم است. این که سیستم ما چه پروسه مهمی را قرار است انجام بدهد و با چه موجودیت هایی در ارتباط است در واقع بالاترین سطح DFDست، در این پروسه بزرگ که اسمش را پروسه کلان گذاشتیم در واقع به ما می گوید اسم سیستم اطلاعاتی را که را در فرایند context diagram قرار بده، مثلاً سیستم اطلاعاتی که میخواهیم تهیه کنیم سیستم سفارش آنلاین غذاست می گوید اسمی که برایش انتخاب کردی را در فرآیندی که در context diagram طراحی می کنی قرار بده.

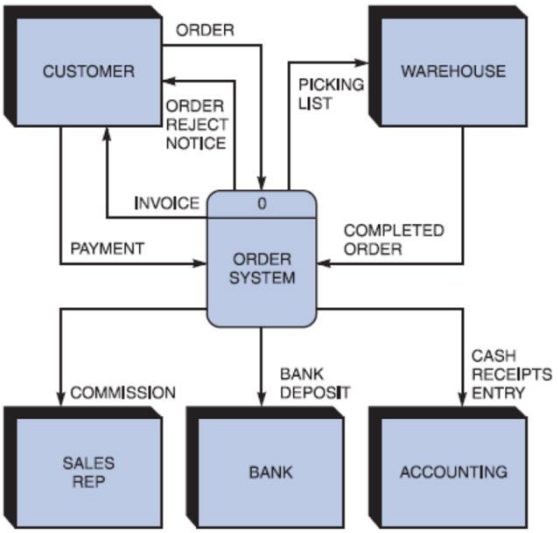
- اسم موجودیت ها، اسم پروسه ها و همه اسم ها که در طول سطوح مختلفی که داریم مدل سازی می کنیم باید منحصر به فرد باید داشته باشد.
- وقتی داریم خطوط جریان را میکشیم این خطوط همدیگر را قطع نکند.

- برای هر فرایند شماره گذاری کنیم و شماره منحصر به فرد داشته باشند.
- مطمئن شویم مدلی که می‌خواهیم تهیه کنیم مدل دقیقی است، اطلاعات را تا حد ممکن اضافه کنیم و اطلاعاتی از دست نرفته باشد و گم نشده باشند.
- نمودار را می کشیم که فهمش برای همه راحت باشد و کاربرانی که قرار است از این سیستم استفاده کنند همه نیازهایشان را در نمودار ما ببینند.

قرار نیست ما فرایند ها و جزئیات نمودار را در یک نمودار بکشیم ما اینها را در سطوح مختلف می کشیم ما یک فرآیند را در context diagram در نظر می گیریم ارتباطش را با موجودیت های دیگر مثلاً اینکه در سیستم ما ۵ موجودیت بوده یا ۵ External entity بوده آنها را در context diagram گذاشتیم و گفتیم این سیستم سفارش ما سیستمی است که یک سری اطلاعات را از مشتری می‌گیرد حال می‌پرسیم آن اطلاعات چیست؟ اطلاعات سفارش، اطلاعات پرداخت، یک سری اطلاعات به مشتری می‌دهد که شامل قبض است؟ یا اینکه سفارش پذیرفته شده؟ و با موجودیت انبار، موجودیت بخش فروش، موجودیت بانک،موجودیت حسابداری در ارتباط است؟

که در این مرحله شماره context diagram صفر می باشد.

به نمودار بعدی نمودار سطح صفر می‌گوییم همان نموداری که کشیدیم را میشکنی فرآیندهای کوچکتر که در dfdیک تکنیکی به اسم Decompostion است، یعنی شکستن یک فرآیند به فرآیندهای کوچکتر،شکستن یک بخش به بخش های کوچکتر.همین سیستمی که راجع به سیستم سفارش صحبت می‌کنیم با مشتری و بانک در ارتباط است این سیستم خودش در سطح ۰ شامل ۳ فرآیند می باشد:



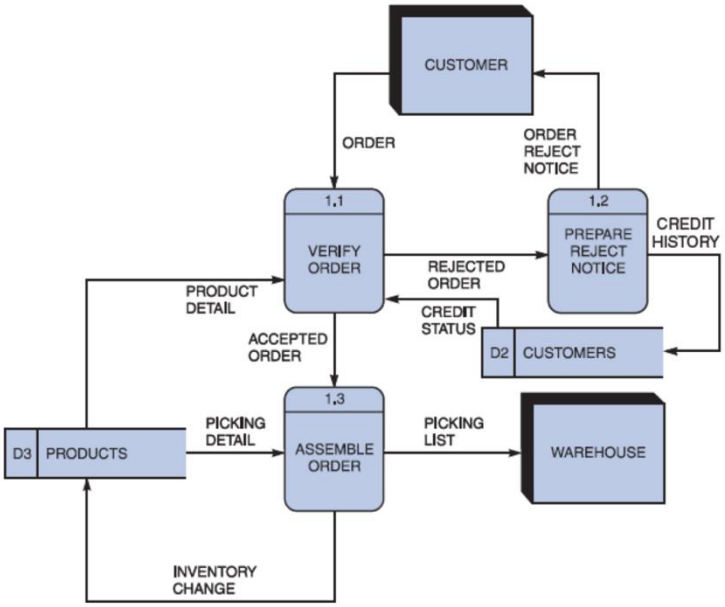
۱. Fill order
۲. Create invoice
۳. Apply payment

بعد شماره گذاری می کنیم باید دقت داشته باشید در سطح ۰ شماره های‌مان یک رقمی است.

مشتری مستقیماً سفارش را به فرایند Fill order می‌فرستد Fill orderاطلاعات مشتری را که گرفت به بخش انبار می‌فرستد انبار یک سری اطلاعات را به Create invoiceمی فرستد و Create invoiceقبض را می‌سازد و به مشتری می‌دهد و می‌گوید سفارش شما پذیرفته شده و یک کپی از همان را به database می فرستد که این database دوباره اطلاعات جزئیات قبض را به اطلاع apply paymentمی‌فرستد و یکی از روش های accounting , bank , sales dept کار پرداخت را مشتری برای انجام می دهد، جزئیات پیام روی database ذخیره می شود که پرداخت انجام شد.

در این بخش یک فرآیند سطح context را به ۳ فرآیند شکستیم ولی می دانیم که هر کدام از اینها جزئیاتی دارند که در سطوح بعدی باید به آنها بپردازیم.

وقتی از سطوح بعدی حرف می‌زنیم یعنی هر کدام از این کارها را باید بشکنیم و به اجزای کوچکتری تقسیم کنیم در این فرایند یک سری data sture اضافه شد. فرایند اصلی که اسمش order بود را و از شماره صفر بود را شکستیم و بسته فرآیند تقسیم کردیم و این و در حین کار متوجه شدیم که data sture اضافه شده اما موجودیت های ما تغییر نکردند ما به این ویژگی **balancing** میگوییم



Balancing

از هر سطح مثلاً context وقتی از سطح ۰ به ۱ یا از ۱ به ۲ و غیره می رویم این ویژگی باید حفظ شود که موجودیت جدیدی اضافه نشود.

در سطح یک شماره بندی فرایندها دو رقم شده ۱ـ۱، ۱ـ۲ و۱ـ۳ ۰

یعنی fill order را به سه فرآیند تقسیم کردیم

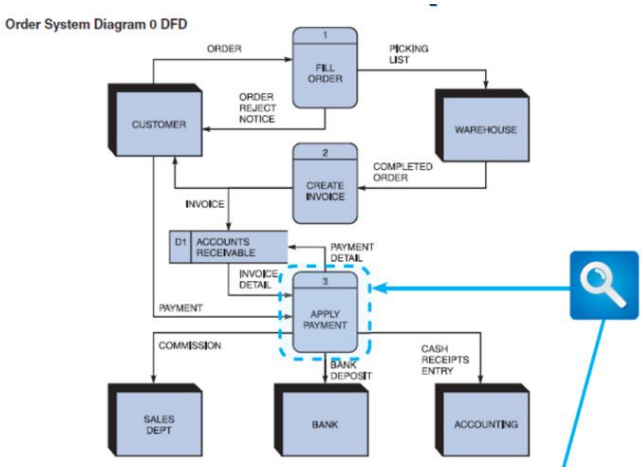
Verify order.۱_۱

Prepare reject order.۱_۲

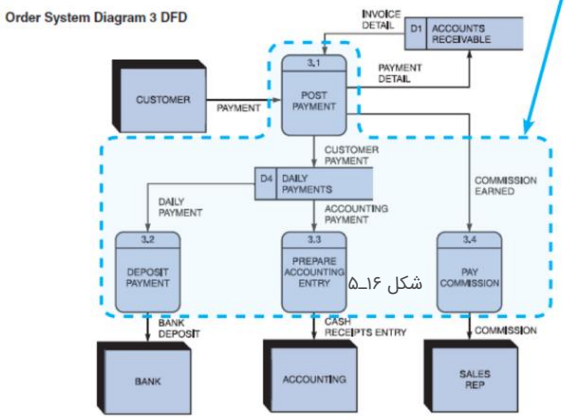
Assembler order.۱_۳

و اطلاعات را در خصوص چگونگی انجام فرایند که از مشتری گرفته بودیم سعی کردیم به این شکل نشان بدهیم این نمودار را که در سطح یک کشیدیم(شکل ۱۴ـ۵)کل سیستم نیست و بنابراین اگر می بینید قسمت bank و ... نیست، دلیل آن نیست که ما آنها را حذف کردیم ما کل سیستم را مدل نکردیم در این سطح یک شیوه این است که موجودیت ها را در سطح یک نشان بدهیم اما می توانیم بگوییم این از یه موجودیتی حاصل شده که

در این قسمت ماست سیستم را بریدیم یعنی موجودیت ها را ترسیم نکردیم فقط فرایندها و data sture‌هایی که با آنها در ارتباط بودند را در این سطح نشان دادیم عملاً کاری کردیم که فرآیند شماره ۱ را به سه فرآیند بشکنند اگر لازم باشد هر کدام از این فرآیند تا را نیز می‌توانیم بشکنیم و در سطح بعدی نیز انجام بدهیم.



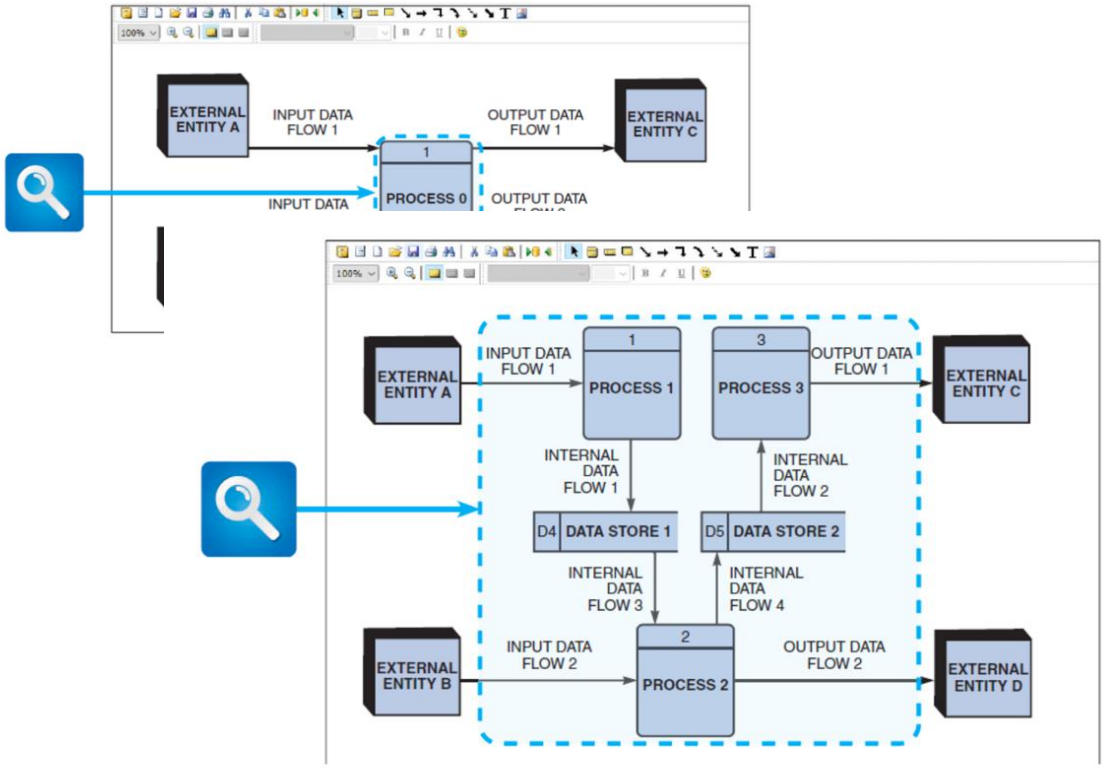
شکل ۱۶ـ۵ مثلاً دسته یکی از فرایند ها را برداشتیم جزئیات بیشتری را راجع به این فرایند توضیح دادیم، میتوانیم Entity هایش را نگه داریم و هم می‌توانیم حذف کنیم و فقط در سطح یکی که داریم نشان میدهیم خود فرآیند شماره ۳ را به۱ـ۳، ۲ـ۳، ۳ـ۳ تقسیم کرده باشیم.



شکل ۱۶ـ۵

نتیجه اخلاقی: اطلاعات را تا حد ممکن به صورت دقیق جمع‌آوری کنیم و این اطلاعات را در سطوح مختلف اضافه کنیم یعنی همه اطلاعات را در یک نمودار بکشیم ماه اول در سطح context diagram فقط کلیت سیستم را می‌گوییم باید در سطح صفر می‌آییم خود contextرا به چند فرایند اصلی تقسیم می کنیم در سطح یک هر کدام از این فرآیند هایی که در سطح صفر بودن را انتخاب می کنیم میشه کنیم و به یک تعداد فرآیندهای کوچکتر ها را دوباره مدل می‌کنیم و تا جایی این کار را تکرار می کنیم که برسیم به جایی که فرآیند ها قابل شکستن نباشند و برای ما و طراح مشخص باشد که این فرایند مثلاً باید به یک function مشخص در سیستم یا به یک دکمه سرویس یک سرویسی که همه بر روی آن اتفاق نظر دارند و می توانند آن را درک کنند و به راحتی آن را پیاده سازی کنند تبدیل شود این کاری است که باید روی نمودار DFD انجام بدهیم.

شکل ۱۷ـ۵ این است که در سطح context را میشکنیم و به یک تعداد فرآیند کوچکتر ۱،۲،۳ و یک سری data sture این بین وجود دارند که اطلاعات را از ما می‌گیرند نگهداری می‌کنند، تبدیل می کنیم.



شکل ۱۸ـ۵ نمودار DFD ما نمودار فرایند هایی که اطلاعات ما را انجام می دهند، نمایش می‌دهیم و میگوییم از چه موجودیت چه اطلاعاتی را می‌گیرند و این اطلاعات را چگونه پردازش می کند ولی می گوییم این اطلاعات را که پردازش کرد در چه محل هایی نگهداری کند که همان data sture ما هستند.

Data Dictionary

توصیفی که ما از هر فرایند هر data sture موجودیت یعنی توضیحات تکمیلی برای اینکه ما بفهمیم این فرایندی که ما اسمش را ۳ process گذاشتیم یعنی چه؟ چه کار می‌کند؟ ورودیش از کجا می‌آید؟ چه پردازشی روی آن انجام می شود؟ و خروجی اش را کجا میبرند؟

دیکشنری داده ها فقط برای فرایند نیست برای data sture،entityها data flow ها هم هست یعنی چهn علامتی که از آنها استفاده می‌کنیمdata sture, data flow,entity,process))برای این ها توضیح می گذاریم.

Element

یک عنصر داده ای که کوچکترین قطعه اطلاعاتی ست که در سیستم ما معنا دارد. مثلاً فرض کنید وقتی از حساب کاربری صحبت می‌کنیم در واقع این کوچکترین توضیحی است که می‌توانیم بدهیم که حساب کاربری در واقع همان username و password ست که کاربر در پروفایل خودش نگهداری می کند تا بتواند حق دسترسی داشته باشد. یا فرض کنید میزان درآمد یک صفت یا عنصری است که ما توضیح می‌دهیم این میزان درآمد چرا برای ما مهم است چرا باید آن را نگهداریم و چرا پردازش باید روی آنها انجام دهیم عملاً ما داریم data dictionaryرا فراهم می‌کنیم تا بتوانیم توضیحات تکمیلی را روی مدلمان قرار بدهیم ما معمولاً برای اینکه بتوانید data dictionary را تهیه کنیم از case tools ها استفاده می کنیم.

case tools(Computer Aided Software Engneering)

کمک کامپیوتری مهندس نرم‌افزار یعنی از ابزارهایی که در فرایند تحلیل، طراحی، پیاده سازی، تست و نگهداری سیستم ابزار های کامپیوتری هستند، استفاده می‌کنیم.

هرچقدر سیستم پیچیده تر باشد مستندسازی پیچیده‌تری هم لازم دارد مستندسازی بیشتری لازم دارد و عملاً مستند در مستند سازی سعی می کنیم پیچیدگی ها را ساده تر کنید و ابزارهایی که اخیراً تولید شدند مثلا گیت هاب یکی از ابزارهایی است که به انجام کارهای پروژه نرم‌افزاری اتفاق می افتد و به ما کمک می‌کند که بخشی از کار را برای ما مستندسازی کند.

ادامه data dictionary: اطلاعاتی را که راجع به DFD فراهم کردیم را باید در data dictionary ذخیره و مستند سازی کنیم اطلاعاتی در خصوص attribute و recordست، وقتی ما یک موجودیت را در database به یک table تبدیل می کنیم و درواقع در ابتدا entity بوده, مثلاً برای مشتری اسم سن تاریخ تولد آیدی و غیره هر نوع این ویژگی است که تعریف می کنیم.

مثلا وقتی بگوییم در سیستم سیستم بشیری با سن ۴۰ سال و غیره به اینها رکورد داخل database می گویند.

data dictionary در واقع قرار است مشخص کند که چه نوع اطلاعاتی درون table ها ذخیره شود توضیح دهد که اینجا نام خانوادگی قرار بگیرد از بازی ۲۰ تا ۹۰ سال می تواند قرار بگیرد تاریخ تولد به چه فرمت هایی می تواند قرار بگیرد اسم مستعار نوع طول مقدار پیش فرض مقادیر قابل پذیرش مانند پذیرش نوع فرمت تاریخ

تولد منبع امنیت چطور برآورده می شود که افرادی اجازه دارند به این اطلاعات دسترسی داشته باشند و آنها را وارد کنند همه این کارها را در data dictionary برای ریکوردر ها انجام می دهیم.

حال برای data flow ها جریان داده از یک منبعی ما را به یک مقداری می‌برد که این منبع می‌تواند یک موجودیت باشد یا یک مقصد فرآیند باشد یا مثلاً یک پروسه به یک پروژه دیگر یک پروژه به یک data sture دیگر باشد که قبلاً به این ها اشاره کردیم.

ما data sture ها را هم باید document کنید و برای آنها dictionary درست کنیم، باید بگوییم که عنوان این data sture ها چیست؟ lable چیست؟ و توضیح بدهیم که این data اصلاً برای چه نگهداری می شود؟ چه ویژگی‌هایی دارد؟ چند وقت یکبار آپدیت می شود؟اسم یا اسامی جایگزینی که دارد چیست؟ چند وقت یکبار ساخته می شود؟ یا اطلاعات درون آن insert می شود؟یا اطلاعاتی در آن بازیابی می شود؟

خود فرآیندها مهم ترین بخش در سیستم ما هستند که باید برای آنها data dictionary درست کنیم مشخص کنید که یک فرایند چه اسمی دارد توضیح بدهیم که این فرآیند کارش چیست شماره اش چیست به انواع شماره ها اشاره کردیم توصیفی از فرایند ها را ادامه ارائه بدهیم.

موجودیت ها اول ابتدایی در ابتدا گفتیم که چهار element در سیستم داریم data flow،entity،process ، data sturکه باید برای تک تک این ها data dictionary درست کنیم برای موجودیت ها باید بگوییم اسم موجودیت ها چیست؟ موجودیت ها راتوضیح بدهیم اسامی شان چیست ورودی‌ها و خروجی‌های شان چیست؟ اینها را توضیح می‌دهیم و data dictionary داد درست می کنیم عملاً سعی می کنیم تمام ریکورد هایی که قبلاً تعریف کرده‌ایم که چطور database آن قرار می‌گیرد که ویژگی‌هایی دارد مقادیر پیش فرض شان چیست و همه این ها را مستندسازی کنیم برای ریپورت ها و گزارش هایی که سیستم ارائه می‌کند نیز باید دیکشنری درست کنیم یعنی بگوییم که گزارشی که درست می شود اولاً چه ویژگی باید داشته باشد در این گزارش چه نوع اطلاعاتی باید قرار بگیرد که اطلاعاتی مهم هستند و باید باشند چه اطلاعاتی ضروری نیستند اینکه اطلاعات چگونه تهیه می شود و از کدام فرایند این گزارش‌ها را تهیه می‌کند منبع اطلاعاتی این گزارش ما چیست این ها همه یک توصیف دیتا دیکشنری است.

ما موفق شدیم که متوجه شویم که اگر از سیستم یک مدل DFD آماده کنیم یک مدل ERآماده کنیم data dictionary برایش آماده می‌کنیم.

تا به اینجا کمکی که توانستیم در تحلیل سیستم انجام بدهیم این بود که متوجه شویم دی اف دی این کمک را به ما کرده که چه فرایندهایی در سیستم وجود دارند که موجودیت هایی وجود دارند ارتباط بین موجودیت ها و فرآیندها چیست چطور ارتباط برقرار می‌کنند داده‌هایی را که رد و بدل کردند را مشخص کردیم ده ها چگونه در دیتا استور قرار می‌گیرد.

خود داده ها چه جزئیاتی دارند یعنی جزئیات data sture را با ER diagram نشان می دهیم.dictionary هم که مستندسازی تمام اجزایی که در سیستم وجود دارند است.

ما درباره اینکه یک فرآیند چگونه کار میکند چیزی نگفتیم مثلاً اگر فرایندی به نام tax calculation و محاسبه مالیات داریم نگفتیم که چگونه حساب می شود مثلاً مالیات تا یک مبلغی صفر می باشد و از یک مبلغی تا یک مبلغی در درصد و ....

برای اینکه این توضیحات را بدهیم لازم است اصطلاح مدل سازی منطقی را تکمیل کنیم یعنی با یک سری افزارهای دیگرDecision tree ، decision table، structed english منطق درون فرآیندها را توضیح دهیم.

ابزارهایی وجود دارند که بتوانیم منطق داخل فرآیندها را توضیح بدهیم.

structed English: یعنی وقتی میخواهیم توضیح بدهیم که مالیات چگونه حساب می شود یا چگونه حقوق‌ها حساب می‌شوند بیایم به شکل سودوکو توضیح بدهیم.

Decision table: لیست کنیم و می گوییم اگر یک نوع از مشتری های ترجیحی باشند قانون ۱،۲،۳ برایشان اعمال می شود اگر سفارش بیشتر از هزار دلار باشد قانون‌های فلان و فلان انجام می شود (منطق انجام یک فرایند).

Decision tree: آیا این مشتری ترجیحی ماه هست یا نه؟ اگر بله سفارش ۱۰۰۰ دلار بیشتره؟ اگر بله می خواهد آیا می خواهد از کارت اعتباری استفاده کند؟ اگر نه فلان قوانین (این منطق داخل یک فرایند).

فصل ۶

Development Strategies

در این مرحله به تحلیل و طراحی سیستم، چگونگی توجه به استراتژی های توسعه نرم افزار می پردازیم.

یکی از مهارت هایی که یک تحلیلگر باید داشته باشد این است که بشناسد چه تکنولوژی هایی برای حل مسئله وجود دارد، مثلا اگر می‌خواهیم یک نرم افزار بنویسیم باید تحلیلگر متوجه تفاوت های راه حل های مختلف را داشته باشد، اینکه solution دسکتاپ با solution تحت شبکه با solution تحت وب چه فرق هایی با یکدیگر دارند و چه امکاناتی را در اختیار کاربر قرار می‌دهد.

اگر به عنوان یک کارشناس IT در یک سازمان کار می کنیم، درک اینکه این پروژه را داخل خود سازمان با مجموعه ی برنامه نویس های خودمان بنویسیم اصطلاحی که در این فصل تحت عنوان In House Development یاد می کنیم یا توسعه درون خانه _سازمان، یا اینکه پروژه را OutSource کنیم یعنی به سازمان های دیگر بدهیم، یا پکیج های آماده بخریم یا اینکه به صورت سرویس، solution as service یا software as service خدمات بگیریم. مثل خیلی از سیستم هایی که ما تحت عنوان cloud computing از آنها یاد می کنیم.

اینکه چه فرق هایی با دارند و عملا بخش مالی شرکت را چگونه قانع کنیم که کدام یک از این بسته ها را باید تهیه کند، در ادامه خواهیم پرداخت.

اهداف این فصل:

بتوانیم تفاوت توسعه های سنتی را با توسعه های تحت وب متوجه شویم، اینکه مفهوم web۲.۰ و مفاهیمی مانند cloud computing یا محاسبات ابری، اپلیکیشن هایی که روی ابزار های موبایل کار می کنند و اینکه ما توسعه را ببریم. یعنی اپلیکیشن بنویسیم،وب سایت درست کنیم یا یک اپلیکیشن دسکتاپ باشد (منظور همان اپ موبایل بود) این ها چه فرق هایی با هم دارند؟

اینکه کار In House Development را انجام دهیم یا این کار را OutSource کنیم.

۱. In House Development: توسعه نرم افزار درون شرکت
۲. Outsourcing: کار را به یک تیم دیگر بسپاریم که آن را برای ما انجام دهد.
۳. offshoring در واقع یک outsource هست که فرا مرزی ست یعنی عملا کار را خارج از مرز های کشور انجام می دهیم و همینطور درباره‌ی software as a service صحبت خواهیم کرد.
۴. software as a service یعنی ما نرم افزار را به عنوان یک سرویس، یه عنوان یک خدمت ارائه بدهیم، صرفا اگر مشتری هستیم از نرم‌افزاری که توسط یک شرکت دیگر نوشته شده خدمات بگیریم و اگر توسعه

دهنده نرم افزار هستیم عملا ما سرویس ها را بفروشیم، مثلا عضو یک سایت شدیم، حق عضویت می دهیم و از آن سرویس دریافت می کنیم.

این ۴ روش هست که با آنها می توانیم توسعه ی محصول یا استفاده یا آن خدماتی که میخواهیم کامپیوتری بشود را دنبال کنیم.

اینکه یک تحلیلگر چگونه می تواند در تصمیم گیری ها کمک کند، که از این روش های مختلف کدام انتخاب شود و همچنین در خصوص اینکه فرایند نرم افزار چطور باید احصا شود صحبت خواهیم کرد. در مورد موضوع RFP و RFQ صحبت می کنیم که RFPیکی از کار های بسیار مهمی است که یک کارشناس IT یا یک تحلیلگر سیستم باید تسلط کافی داشته باشد.

فاز هایی که لازم هست تا ما بخش تحلیل SDLC را به پایان برسانیم را توضیح خواهیم داد.

مقایسه روش های سنتی توسعه نرم افزار با روش web_based

تحلیلگر سیستم باید بداند که نرم افزاری که قرار است نوشته شود قرار است با آن روش های مرسوم باشد؟ روش های مرسوم عمدتا دسکتاپ, اپلیکیشن ها و یا NETWORK اپلیکیشن ها هستند، یعنی اپلیکیشن هایی که در یک دسکتاپ کامپیوتر،روی یک کامپیوتر stand alone کار می کند و web_based اپلیکیشنی است که تحت وب کاربرد های زیادی دارد که می‌توانند به آن دسترسی داشته باشند اصطلاحا scale abilty دارد یعنی مقیاس پذیری آن بالاست، می تواند در مقیاس کوچک و در مقیاس خیلی بزرگ کار کند.

برای توسعه وب حداقل ۲ تکنولوژی را معرفی می‌کند:

۱) Microsoft-NET:

که این امکان وجود دارد که ما عملا با زبان هایی مثل Asp-NET و C# و محیط visual studio بتوانیم اپلیکیشن web-based توسعه دهیم.

۲)MERN stack:

تکنولوژی ای که mongoDB معرفی کرد برای مدیریت دیتا بیس هایش. در MERN در واقع M اشاره دارد به mongoDB و E به(.js)Express) و R به (.js)React)و N به(.js)Node)که back-end کار را با Nodeمی نویسیم.

در واقع وقتی از MERN STACK برای توسعه اپلیکیشن های تحت وب استفاده می کنیم، DataBase را با mongoDB طراحی می کنیم. از فریم ورک (.js)Node برای مدیریت front-end استفاده می کنیم و back-end را با (.js)React)انجام می دهیم که در فرانت اند استفاده می شود که یکی از ساختار های جدید

جاوااسکریپت است تحت عنوان React که قبل از آن Angular بوده و Node(.js) برای بخش back-end یا اصطلاحا برای توسعه web server ها استفاده می شود،این یک راه حل بود.

راه حل دیگر MERN STACK هست که نوع دیگری از همین ترکیب را داریم. mongo راداریم و به جای React از Angular استفاده می کنیم و یا حتی ممکن است یک variant دیگر یا یک پکیج دیگر برای توسعه برنامه تحت وب، به نام MEVN تحت عنوان (MongoDB, Express, Vue, Node)داشته باشیم.

با این توضیحات متوجه شدیم که بخش React و Angular یا Vue قابل جایگزینی هست، هر کدام از این ها یک framework هستند و امکاناتی را در front-endبرای ما فراهم می‌کنند، به شرطی که Database را به سمت mongo ببریم که راه حل بسیار خوب و جدیدی ست.

ما در بخش وب یعنی همان فرانت اند از تکنولوژی ای مانند React جاوااسکریپت یا Angular یا Vue که یک تکنولوژی جدید است و می توانیم از آن استفاده کنیم، در بخش سرئر Expressو Node.js را به کار می‌بریم و در بخش دیتا بیس هم از MongoDB استفاده می کنیم.

داکیومنتیشن خیلی خوبی دارد و عملا از روش های relational که ما در database های مرسوم استفاده می کردیم،کونگو استفاده نمی کند، باز جایگزین ها یا variant های مختلفی تخت عنوان couchdb اسم هایشان را شنیده ایم.

پس متوجه شدیم برای اینکه Web Based توسعه بدهیم یکی از تکنولوژی های مرسوم Microsoft.NET هست که عملا از databaseهایی مانند SQL برای مدیریت داده هایمان استفاده می کنیم.

ولی در MERN STACK تاکید روی تکنولوژی هایی مثل mongoDB و تکنولوژی های دیگری مثل Express و React و Node.js بود که این ها به عنوان راه حل هستند. MERN STACK جز تکنولوژی های جدید و به روزی ست که امکانات خیلی خوبی به خصوص در بخش front-end دارند (هم React و هم Angular) بسیار خوب می تواند با کاربر تعامل کند، presentation فوق العاده ای را می توانیم داشته باشیم.

ولی Microsoft.NET عملا ابزار ها و components های از پیش آماده شده ی زیادی دارد، توسعه را [مقداری ساده تر می کند، database اش سال ها استفاده شده (همان SQL server)بنابراین افراد زیادی را می توانیم پیدا کنیم که با این مجموعه ابزار ها کار می کنند و برنامه نویسی می کنند ولی در حوزه MERN STACK افراد متخصص کمتری هست.

این ها موضوعاتی هست که در انتخاب تکنولوژی می‌تواند برای ما موثر باشد.

در روش های traditional مواردی که مورد توجه هستند این است که ما باید بتوانیم موضوع سازگاری را مدیریت کنیم، معمولا سیستم را برای سیستم های local یعنی همان Desktop application ها و نهایتا سیستم های که تحت شبکه دارند کار می کنند مورد استفاده قرار می دهیم. یعنی اینکه ما یک اپلیکیشن تحت شبکه داشته باشیم که با اپلیکیشنی که در وب کار می کند، فرق می کند.

یک اپلیکیشن میتواند تحت شبکه باشد ولی لزوما Web Based نباشد.

در روش های traditional ما معمولا این ها را می‌دیدیم و سعی می کردیم که اگر از منابعی در اینترنت قرار است استفاده کنیم آن ها را به application که در شبکه کار می کند لینک کنیم.

و مسیری که برای این روش های traditional می توانستیم طی کنیم بدین صورت است که توسعه درون سازمانی داشته باشیم (in house)یا پکیج نرم افزاری را با یک سری تغییرات جزئی از شرکت های ارائه دهنده نرم افزار خریداری کنیم، یا مشورت از بیرون بگیریم و عملا آن فرآیند نرم افزاری را داخل خود سازمان دنبال کنیم.

مثال: یک شرکتی مثل صفا رایانه در ایران در حوزه توسعه ی اپلیکیشن های یکپارچه برای شهرداری مدت هاست که کار می کند ولی شهرداری ها معمولا قوانین متفاوتی دارند، بنابراین اگر صفا رایانه برای شهرداری مشهد یک پکیج را نوشت این پکیج نمی تواند دقیقا در شهر همدان هم استفاده شود،این را باید اصطلاحا یک مقدار modification یا customisation یا به روز رسانی کند یا تغییرات دهد و بومی سازی کند، یعنی بتوانیم یک سری تغییرات اعمال کنیم و این پکیج را خریداری کنیم.

در روش های سنتی در مقایسه با روش های Web Based این است که این مقیاس پذیری محدوده به همان شرایط و محدودیت هایی که ما داخل شبکه داریم یعنی نمی‌توانیم به دلخواه کاربران زیادی را اضافه کنیم، این به محدودیت ها و شرایطی که در شبکه هست بستگی دارد. وقتی ما app را به صورت traditional توسعه دادیم یعنی روی شبکه یا روی دسکتاپ کار می‌کند از توان محاسباتی و منابعی که درون دسکتاپ وجود دارد، استفاده می کنیم. بنابراین کامپیوتر ضعیف نمی تواند کار راه انداز باشد. خیلی وقت ها لازم است که ما اگر application نوشتیم در این کامپیوتر، مثلا Database را نصب کنیم، app را نصب کنیم و عملا app با Database در ارتباط است. حال تصور کنید که در شبکه ای هستیم که افراد به هم وصل هستند و در این شبکه باید هر کدام از این کامپیوتر ها توان محاسباتی کافی داشته باشند.

البته داخل همین شبکه هم یک راه حل این است: ما یک سرور مناسب را در نظر بگیریم و Database را عملا نصب کنیم و client هایمان برای دسترسی به داده ها به Database وصل شوند ولی به هر حال لازم است که توان محاسباتی خوبی را برای کامپیوتر هایی که در شبکه وجود دارند پیش بینی کرده باشیم.

اگر بخواهیم یک مقایسه دیگری هم داشته باشیم با app هایی که به روش های traditional توسعه پیدا می کنند) یعنی در شبکه و روی دسکتاپ کار می‌کنند(با app های Web Based این است که مسائل امنیتی شان

به مراتب کمتر یا ساده تر از مسائل امنیتی اپ های Web Based است.(چرا؟)چون در app های Web Based عملا برای دسترسی هر کاربر دربها را باز گذاشتیم که در اینترنت بتواند به وب سرور ما وصل شود، ولی در مورد شبکه های محلی این دسترسی را عملا از بیرون سیستم و صرفا مسائل امنیتی برمیگردد به این که کاربرهایمان یا استفاده کننده های app داخل شرکت یا سازمان یک سری قوانین و استاندارد ها را رعایت کنند که مدیریت کردن این داستان به مراتب ساده تر از این است که بخش امنیت یک app، Web Based را مدیریت کنیم.

نکاتی که باید در اپ های Web Based به آن ها توجه کنیم

۱. framework های تحت اینترنت باید این app ها توسعه پیدا کنند و نوشته شوند و تحلیل شوند.
۲. در واقع وب به عنوان یک پلت فرم دارد برای ما کار می‌کند یعنی وب یک پلت فرم که ما به آن وصل میشویم پس باید داخل سازمان همه ی کاربرهایمان امکان اتصال به اینترنت یا وب را داشته باشند.

مزیتش این است که به راحتی مقیاس پذیر هست یعنی می توانیم یک اپ Web Based که نوشتیم مقدار کاربرهایمان اگر ۱۰۰ نفر بودند اگر بخواهیم ۱۰۰۰ نفر یا ۱۰۰۰۰ نفر ارتقا دهیم،خیلی کار سختی نیست تنها کافی ست توان سرور را بالا ببریم و یه تغییرات جزئی در Database و در بخش وب سرور و back-end کار پیاده کنیم که کاربرها را بهتر مدیریت کند ولی عملا کار سختی نیست.

میتواند روی multiple hardware environments کار کند یعنی وقتی app را Web Based توسعه دادیم دیگر جای نگرانی نیست که کاربری که دارد وصل می شود سیستم عاملش چیست؟ سخت افزارش چقدر پیچیده است؟ یا توان محاسباتی خوبی دارد؟ فقط میخواهیم بتواند راحت به اینترنت وصل شود و عملا محیط برنامه ی ما را در مرورگر خودش اجرا کند.

برای مواردی مثل ارتباط با مشتری ها، پردازش سفارش و مدیریت مواد یا محتوا app های وب می توانند خیلی خوب کار کنند و مورد استفاده قرار بگیرند.

یک مقایسه دیگر هم کنیم برای اینکه اپ های تحت شبکه با Web Based متفاوت است، این است که میتوانیم سرویس هایی را ارائه دهیم که این سرویس ها کمتر به توان محاسباتی و منابع دسکتاپ وابسته باشد.

همچنین ما میتوانیم به جای اینکه اپ Web Based را به عنوان یک محصول بفروشیم، میتوانیم آن را به عنوان یک سرویس ارائه کنیم یعنی عملا اپ را به جایی نفروختیم و این اپ را به عنوان یک شرکت توسعه دادیم و روی سرور راه اندازی کردیم و تنها سرویس را داریم به مشتری ای استفاده می کند می فروشیم بنابراین این ها فرمت هایی است که در app های Web Based می توانیم تصور کنیم.

app های Web Based نیاز دارند به یک سری میان افزار ها که بتوانند با مجموعه نرم افزارهایی که روی دسکتاپ یا روی وب ما داریم بتوانند ارتباط برقرار کنند و راه حل هایی که Web Based هستند یا نرم افزارهایی

که ما ایجاد می کنیم، این ها در حوزه ی امنیت خیلی پیچیده تر هستند که ما باید به آن ها توجه کنیم و حل کنیم.

احتمالا متوجه شدید که وقتی یک نرم افزار را توسعه می دهیم و روی کامپیوتر دسکتاپ راه اندازی می کنیم خیلی کمتر به مسائل امنیتی آن توجه می کنیم تا اینکه یک اپ را به صورت Web Based ارائه بدهیم و اینکه عملا فرصتی را ایجاد می کنیم تا app های Web Based که افراد بیشتر بتوانند دسترسی داشته باشند، حداقل به لایه ی بیرونی نرم افزار یا وب سرور ما یا front-end، آن صفحاتی که می توانیم نشان دهیم، بنابراین ممکن است از آن قسمت بتوانند وارد Databaseما شوند و اطلاعات را بردارند.

ترند هایی که در نرم‌افزار و توسعه نرم افزارها دارد اتفاق می افتد یکی web ۲/۰ و دیگری cloud computing هست.

Web ۲.۰

معمولا در دسته‌بندی ها آن بخشی از وب را که صرفاً یک سری صفحات استاتیک ایجاد می کرد و آن صفحات را به کاربر ارائه می داد که نسل اول وب می گفتیم.

نسل دوم وب: صفحه استاتیک نداریم بلکه این صفحات دینامیک اند، فرم داریم و کاربران می‌توانند اطلاعات وارد کنند و این اطلاعات در دیتا بیس رفته و یک سری اطلاعات دیگر را به کاربر نشان می‌دهد، اصطلاحا این صفحه کاملا interactive و دینامیک طراحی شده به همین دلیل ما آن را web ۲/۰ میدانیم، این web ۲/۰ اجازه می‌دهد که افراد collaborate (شرکت)کنند، با سیستم interact داشته باشند، اطلاعاتشان را share کنند به روش موثرتری همچنین interactive را به مراتب web ۲/۰ بالاتر برده است.

cloud computing

یکی دیگر از ترند هایی که در حوزه نرم افزار پیشرفت پشگیری داشته است، نرم افزارها به جای اینکه روی دسکتاپ نصب شوند روی سرور نصب می شوند و عملا کاربرها می‌توانند از طریق شبکه ی اینترنت به این مجموعه نرم افزارها دسترسی داشته باشند و سرویس بگیرند مانند Google form, Google doc, Google drive.

Mobile device

اسمارت فون ها و تبلت ها در دست مردم زیاد شده و app هایی که روی آن ها نصب می شوند و کار می‌کنند،یعنی اگر یک فکری درباره‌ی استارت آپ کردیم حتما به این فکر کردیم که اگر قرار است این سرویس را به مشتری ارائه بدهیم به خصوص استارت آپ ها نگاهشان این است که اپ روی موبایل داشته باشند یا وب و یا ترکیب هر دو را داشته باشند.

ابتدا باید تصمیم بگیریم سرویس مان را می خواهیم به عنوان یک اپلیکیشن وب ارائه بدهیم یا اپ موبایل یا هر دو. اگر قرار است هر دو باشد اول می خواهیم اپ موبایل را ارائه بدهیم یا وبسایت رو یا برعکس، یا اگر app موبایل ارائه می کنیم روی چه موبایل هایی مدنظرمان است، روی چه سیستم عامل هایی قرار است کار کند.

می خواهیم دو نقش داشته باشیم: یک نقش به عنوان کارشناس IT سازمان که کمک می کند مدیر سیستم یا مجموعه تصمیم گیری کند، یکی هم به عنوان یک تحلیلگر در یک شرکت یعنی خودمان جز مجموعه ی برنامه نویس ها یا تحلیلگر های سیستم یا مدیر پروژه و... باشد، هدف این است که ما به عنوان یک شرکت نرم افزاری توضیح بدیم.

یکی از روش هایی که معمولا سیستم ها را توسعه می‌دهند این است که روش In House را استفاده کنیم.

فرض کنید شهرداری همدان می خواهد سیستم شهرسازی و نوسازی و مطالباتش و سیستم های مختلفی که در شهرداری وجود دارد و الان بخشی از آن دارد دستی انجام می شود و بخشی با کامپیوتر انجام می شود آنها را یکپارچه کند.

استراتژی اول این است که به برنامه نویس های خودش که در یک سازمانی به نام سازمان فناوری اطلاعات خودشان حضور دارند بسپارد و بگوید که شما بیایید این سیستم را تحلیل کنید، مدلهایتان را طراحی کنید، پیاده سازی کنید،کار را تست کنید و آن را راه اندازی کنید. در مناطق مختلف شهرداری، به آن ها می گوید من دارم به شما حقوق میدهم و برای حقوق این کار را باید برای ما انجام دهید، یعنی ما تصمیم می گیریم که نرم افزار را داخل سازمان(IN HOUSE DEVELOPMENT)توسعه بدهیم.

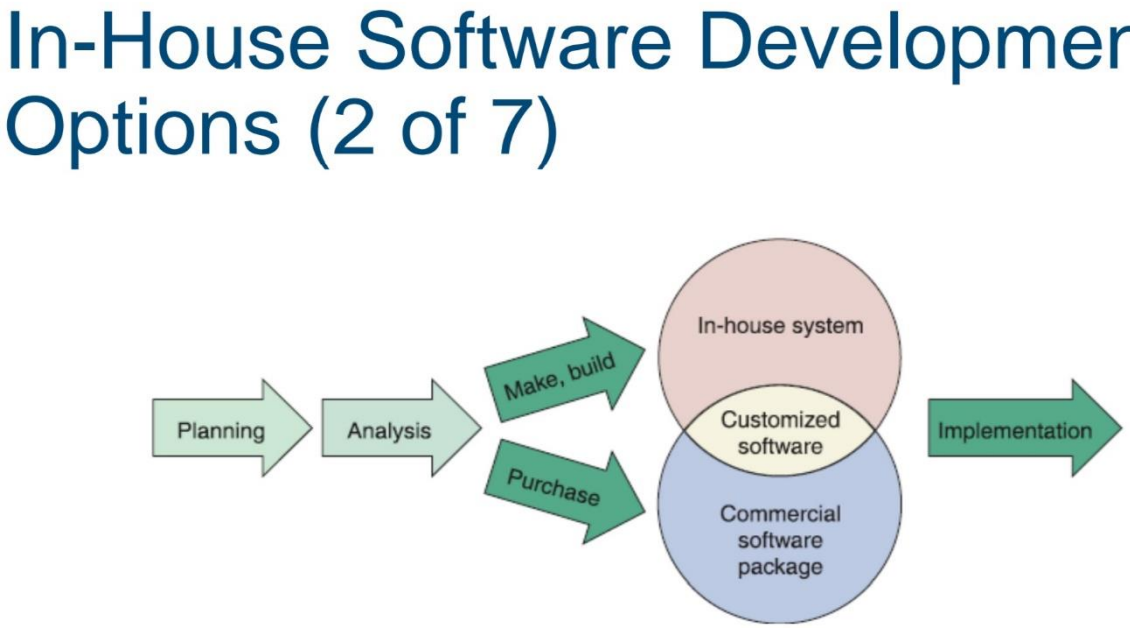
سازمان های زیادی هم هستند که برنامه نویس های مختلفی دارند مانند دیجی کالا، در دیجی کالا تعدادی برنامه نویس هستند که کار آنها توسعه است که در واقع توسعه نیاز ها یا محصولات است.

ما نرم افزار را از بیرون نمی خریم و خودمان توسعه می دهیم، در مورد شهرداری این توسعه عملا دارد توسط مجموعه کارکنان و برنامه نویسان شهردار اتفاق می افتد.

از طرفی مالکیت این نرم افزار با خود شهرداری است چون نیرو های خود سازمان توسعه را انجام دادند و این هم یک نکته مهم برای تصمیم گیرنده است.

یک حالت دیگر تصمیم این است که ایجاد کنیم یا بخریم که اگر تصمیم بگیریم خودمان ایجادش کنیم روش In House می شود و اگر بخواهیم بخریم را در ادامه خواهیم گفت:

شرکت هایی هستند که می توانند app های کاربردی خود را بر اساس یک سری پکیج هابی که از قبل آماده شده است تحت عنوان پیش ساخته، آنها را مقداری تغییر داده و روی سیستم خود نصب کنند، یعنی به جای اینکه کارمان را از صفر توسعه بدهیم و نرم افزار را بسازیم، نرم افزار شرکت بیرونی را می‌خریم. مثلا شرکت صفا رایانه یک ابزار را نوشته ولی ما آن را می خریم و یک سری تغییرات جزئی روی آن اعمال می کنیم و روی سیستم راه اندازی می کنیم.



توضیح شکل: در هر صورت تحلیل سیستم داخل خود سازمان انجام می شود یعنی In House، باید دقیقا بدانیم که چه چیزی می خواهیم بنابراین اگر ما کارشناس IT سازمان هستیم، حتی اگر قرار نیست خودمان مستقیما از صفر محصولات خود را توسعه بدهیم اما باید بدانیم نیاز سازمان دقیقا چه چیزی است، سازمان اولا چطور کار می کند؟ کدام بخش را میخواهد کامپیوتری کند؟ و اینکه دقیقا چه انتظاراتی را از این نرم افزار دارد؟

کار برنامه ریزی را باید داخل سازمان انجام بدهیم، کار تحلیل را انجام می دهیم، در اینجا (۱)تصمیم می گیریم که خودمان این را بسازیم که مسیر (۱) را می رویم و یا اینکه پکیج های آماده بخریم و آن ها را تغییر بدهیم.

اگر بخواهیم خودمان آن را ایجاد کنیم یعنی روش In House System ها، که ما توسعه می دهیم، مالکیت هم با خودمان است، پیاده سازی کرده و توسعه می دهیم، مستقرش کرده و نرم افزار را استقرار می دهیم.

اگر مسیر (۲)را انتخاب کنیم از پکیج های از پیش آماده شده خریداری می کنیم و لازم است که بخشی از این محصول را customize کنیم یعنی برحسب تحلیلی که انجام دادیم و نیاز هایی که داریم customize می کنیم و بعد پیاده‌سازی کرده و انتظار داریم نهایتا چیزی را که میخواهیم برای ما خروجی بدهد.

هر کدام از این مسیر ها مزایا و معایب خود را دارد ولی تجربه نشان داده در خیلی از سازمان ها برنامه نویس هایی که در خود سازمان وجود دارند معمولا با تکنولوژی های به روز آشنا نیستند، بنابراین محصولاتی که توسعه می دهند محصولات نسبتا قدیمی و اینکه ممکن است کارایی کمتری داشته باشند.

software package:

این نرم افزار را از یک فروشنده ی نرم افزار یا اپ خریداری کرده و بعد یک سری تغییرات روی آن اعمال می کنیم.

software vendors:

کسانی که فروشنده نرم افزار هستند یعنی از بیرون تهیه کرده مانند صفا رایانه.

Value added reseller:

نماینده های فروش که ارزش افزوده ایجاد می کنند یعنی بر اساس نیاز هایی که ما به آن‌ها اعلام می کنیم، آن بسته ای که خودشان یا شرکت تولید کننده ی اصلی تولید کردند و این ها نماینده هستند را، تغییراتی داده و سپس به ما می فروشند، معمولا شرکت هایی که خارج از ایران هستند و نمایندگی داخل ایران دارند مانند هواوی و... (فقط سخت افزاری نیستند و نرم افزاری هم دارند، معمولا در کشور های مختلف نمایندگی دارند که همان reseller ها هستند و value added ها هستند.)مثلا یک تیم صد نفره در ایران دارند که به پکیج آنها دسترسی دارند و با توجه به شرایط ایران، مثل مخابرات ایران، می گوید که نیاز های ما این ها هستند و افراد تغییراتی را در این پکیج اعمال کرده و نصب و راه اندازی می کنند که مدیریت و نگهداری و... را هم به عهده دارند.

Horizontal application:

برنامه های کاربردی افقی، به این معنی که ما یک نرم افزار ایجاد می کنیم که توسط سازمان های مختلفی استفاده می شود یعنی نوع کار طوری هست که اپ ما توسط خیلی سازمان ها استفاده می شود.

vertical application:

در مقابل Horizontal، عملا نرم‌افزار ما داخل یک سازمان استفاده می شود (عمق دارد)اپ های عمودی، یعنی نیاز های ویژه ای داخل یک سازمان وجود دارد، ممکن است بخشی از app ما عمومیت داشته باشد اما هر وقت بخواهیم این app را در جایی راه اندازی کنیم باید کلی تغییرات در آن بر حسب نیاز های خاصی که در سازمان خریدار وجود دارد،ایجاد کنیم.

در واقع در توسعه نرم افزار هایی که داخل سازمان اتفاق می افتد، ما انتظار داریم که آن ویژگی ها و نیازمندی های منحصر به فردی که داخل سازمان ما هست محقق شود و آن برنامه نویس های ما خوب بدانند که در سازمان ما این نیاز ها وجود دارند، که امکانش نیست که این قابلیت ها در نرم افزار های از بیرون تهیه شده این ویژگی هایی که دنبالش هستیم را خیلی از آن ها ندارند و به همین خاطر تصمیم می گیریم که با روش In House توسعه دهیم.

اینکه ما درون سازمان توسعه را انجام دهیم تغییرات را به حداقل می رساند، شرایط و محدودیت هایی که سیستم های موجود دارند را معمولاً خیلی خوب می توانیم اعمال کنیم به خاطر اینکه نیرو های کاری داخل سازمان هستند و شرایط سازمان را می دانند و اینکه از منابع و ظرفیت های داخلی استفاده می کنیم.

اما اگر پکیج های آماده را خریداری کنیم احتمالا هزینه ی توسعه برای ما کمتر است (معمولا). حتی در محاسبات اولیه در بخش planning ممکن است فکر کنیم که هزار واحد لازم است تا app را نیروهای خود سازمان توسعه بدهند ولی ۸۰۰ واحد دارند، همان app را خارج سازمان به ما می فروشند، در حالی که فقط ۲۰۰ واحد کم داریم بنابراین تصمیم میگیریم In House کار را توسعه بدهیم. اما واقعیتی که وجود دارد،معمولا در عمل این این هزار واحد ما خیلی اوقات تا دوبرابر افزایش پیدا می کند و متوجه می شویم که نیروهایمان مسلط می شوند و باید آموزش ببینند، هزینه می کنند و زمان زیادی را معمولا انجام پروژه های داخل سازمان اجرا شده به خودش اختصاص می‌دهد، بنابراین کلی حقوق باید پرداخت کنیم و برآورد اولیه ما خیلی اوقات درست بدست نمی‌آید و اگر برویم از پکیج های آماده بخریم، عملا زمان کمتری را برای راه اندازی سیستم نیاز داریم و اینکه قابلیت اعتماد ثابت شده است.

در زمان ارزیابی مثلا محصول صفا رایانه قابلیت اطمینان دارند و خودشان را ثابت کرده اند (چرا؟) چون در جاهای دیگر اجرا شده است و چند سال است استفاده می شود و میتوان تا حد زیادی به آن اعتماد کرد.

اگر پکیج آماده تهیه کنیم نیاز به افراد فنی کمتری داریم چون دیگر برنامه نویس حرفه ای نمی خواهیم که استخدام کنیم و ماهانه به آن حقوق بدهیم.

به روز رسانی ها توسط خود فرد فروشنده انجام می شود نه توسط ما.

میتوان از شرکت های دیگر اطلاعات گرفت که اگر مورد استفاده قرار گرفته چقدر راضی کننده بوده تا این پکیج را با پکیج های دیگر مقایسه کنیم.

فرض کنیم پکیج آماده را خریدیم پس باید آن را customize کنیم، قرار شد بر اساس نیاز های خودمان پکیج را customize کنیم. بنابراین هزینه ای را باید بابت پکیج اولیه یا basic پرداخت کنیم و هزینه ای را بابت customize کردن که نیازمندی های ما را دقیقا در سیستم اعمال کند و می توانیم مستقیما با آن فروشنده ی نرم افزار وارد مذاکره شویم و درباره ی هزینه و زمان و نیازمندی ها صحبت کنیم.

بحث های modification را از آن شرکت خریداری کنیم یعنی تغییرات مورد نظرمان را اعمال کند.

با این روش داریم اپ های کاربردی را توسعه می دهیم و سعی می کنیم از نرم افزارهایی استاندارد که در حوزه ی کسب و کار وجود دارد استفاده کنیم و برای کاربرهایمان خدمت بگیریم.

دو بخش دارد:

۱. app را به صورت پکیج خریدیم می توانیم بگوییم این را برای کاربرهایمان به صورت دقیق customize کند یانه. پکیج هایی که ما از قبل داشتیم، یک interface طراحی کند که کاربرهایمان از طریق این User InterfaceUI))با پکیج هایی که از قبل وجود داشتند ارتباط بگیرند.
۲. یا service desk برای آن‌ها فراهم کنیم، یک service desk که خیلی نزدیک به همین UI می باشد. یک interface را فراهم می کنیم که سرویسی را روی این دسکتاپ می گیرد و عملا به آن پکیجی که برایش خریدیم متصل است.

مدل سازی

هر مسئله ای که برای مدل سازی به ما داده می شود، ابتدا باید موضوع مسئله و جزئیات آن روشن باشد.

مسئله مدل سازی

دامنه مسئله: کتابخانه دانشگاه

قصد داریم برای کتابخانه مرکزی دانشگاه، نرم افزاری بنویسیم که پلت فرم نرم افزار برای ما مهم است. یعنی باید در جریان تعاملی که در بخش تحلیل با کارفرما داریم بپرسیم که این راه حل نرم افزاری را در چه پلتفرمی می خواهند، اجرا کنند. مثلا می گویند که آنلاین تحت وب باشد، یا در شبکه کار کند و فقط دانشجویانی که وارد کتابخانه می شوند می توانند سرچ کنند؛ یا نه، از همه دنیا بتوانند سرچ کنند که چه کتاب هایی داریم. و اگر در سیستم login کنند، می توانند کتاب را رزرو نمایند.

طبیعتا در موضوع آنلاین امکان تحویل کتاب، یا دریافت کتاب (امانت دادن) را نداریم. این موضوعی است که باید در بخش local برای کتابدار ببینیم، ولی در بخش آنلاین فقط می توانیم رزرو را انجام دهیم و قوانین مخصوص را دارد. می توانیم فرآیندها را پیدا کنیم، data sture را پیدا کنیم، اما بخشی از کار باقی می ماند؛ در آن سطح دسترسی که کاربر کتابخانه یا همان کتابدار در سیستم login می کند و می تواند کتاب را تحویل بگیرد یا تحویل بدهد. پس در ابتدا لازم است که صورت مسئله کاملا روشن شود.

فرض نمایید به ما گفته اند دامنه مسئله کتابخانه دانشگاه است و ما قرار است به روش ساختار یافته، فرآیندها را شناسایی کنیم و DFD را مدل کنیم. عملکردهایی که از سیستم انتظار هست باید برای ما توضیح داده شود، به عنوان مثال نام اعضا وجود داشته باشد، اینکه اعضا توانایی بررسی و جستجو در محتویات کتابخانه انجام دهند، ممکن است در همین زمینه بگویند برای اینکه کاربر جستجو انجام دهد، مهم نیست عضو login کرده باشد، اما برای امانت و تحویل و رزرو حتما باید login کرده باشد. پس این موارد را باید بررسی کنیم و در بخش تحلیل، اطلاعات را جمع آوری نماییم.

مسئله مدل سازی

دامنه مسئله: کتابخانه دانشگاه

زبان مدل سازی DFD

عملکردهای اصلی سیستم: امکان ثبت نام اعضا - بررسی و جسجو در محتویات کتابخانه – امانت item – تحویل item (کتاب،CD)

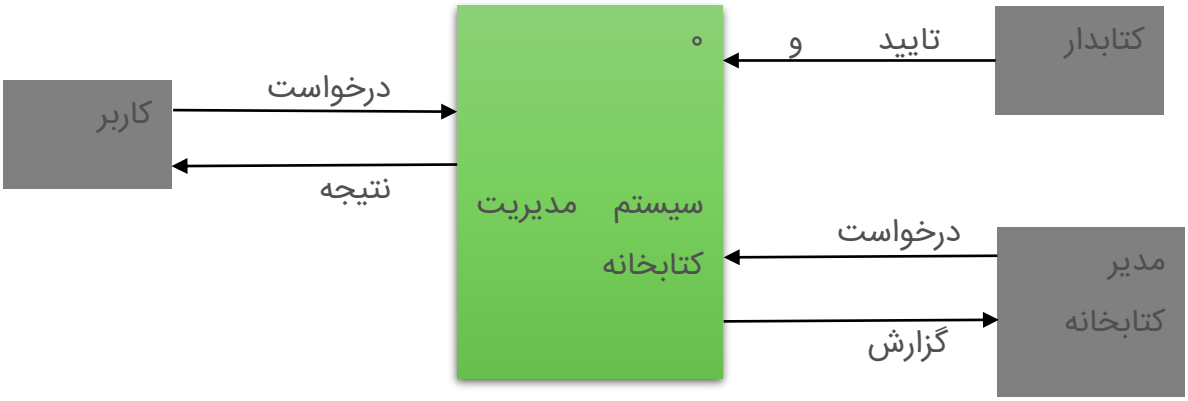
پلتفرم: آنلاین تحت وب

مدل سازی را چگونه انجام بدهیم؟ به پلتفرمی که مسئله را برای آن می نویسیم بسیار مهم است که توجه کنیم. در ابتدا یک Context Diagram رسم می کنیم و نمودار سطح ۰، سطح۱ و... را تا سطح لازم رسم می کنیم و این کار را تکرار می کنیم تا برای همه فرآیندها به مرحله ی فرآیندهای اتمیک یا فرآیندهای مرحله برگ برسیم.

عنوان سیستم: سیستم مدیریت کتابخانه

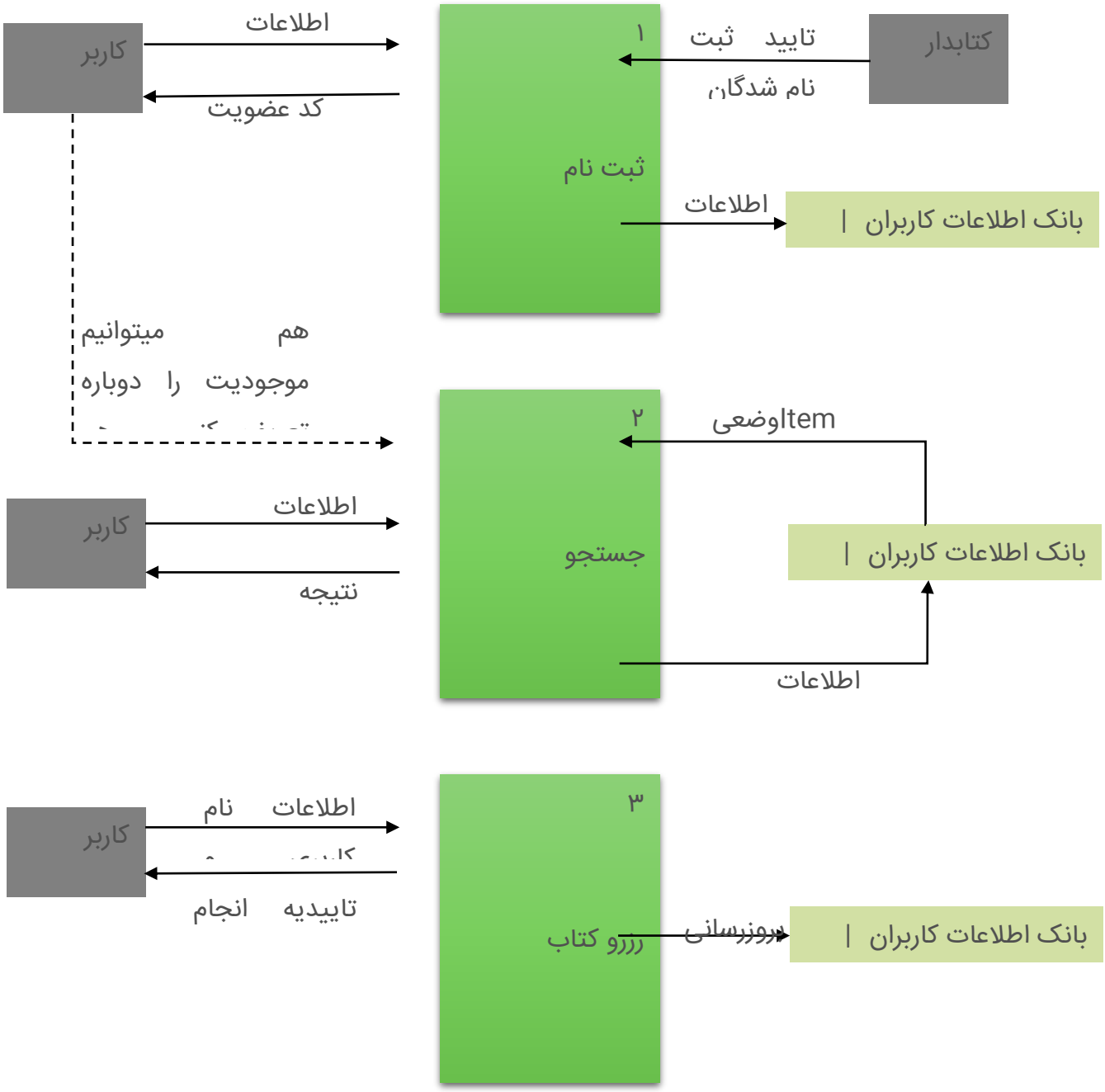
موجودیت ها: کاربر(دانشجو، عضو هیئت علمی، کارکنان دانشگاه)– کتابدار – مدیرکتابخانه(مدیر می تواند به یک سری گزارش هایی دسترسی داشته باشد که کتابدار نداشته باشد و اصلا این مدیر است که کتابدار و دانشجو را در سیستم تعریف می کند یا یک سری قابلیت ها را فعال می کند.)

در مرحله Context Diagram که با ۰ آن را نشان می دهیم سیستم را به صورت Process Model رسم می کنیم.



در نمودار سطح۰ این فرآیند را به یک سری فرآیندها می شکنیم و این ها را از صورت سوال می آوریم.

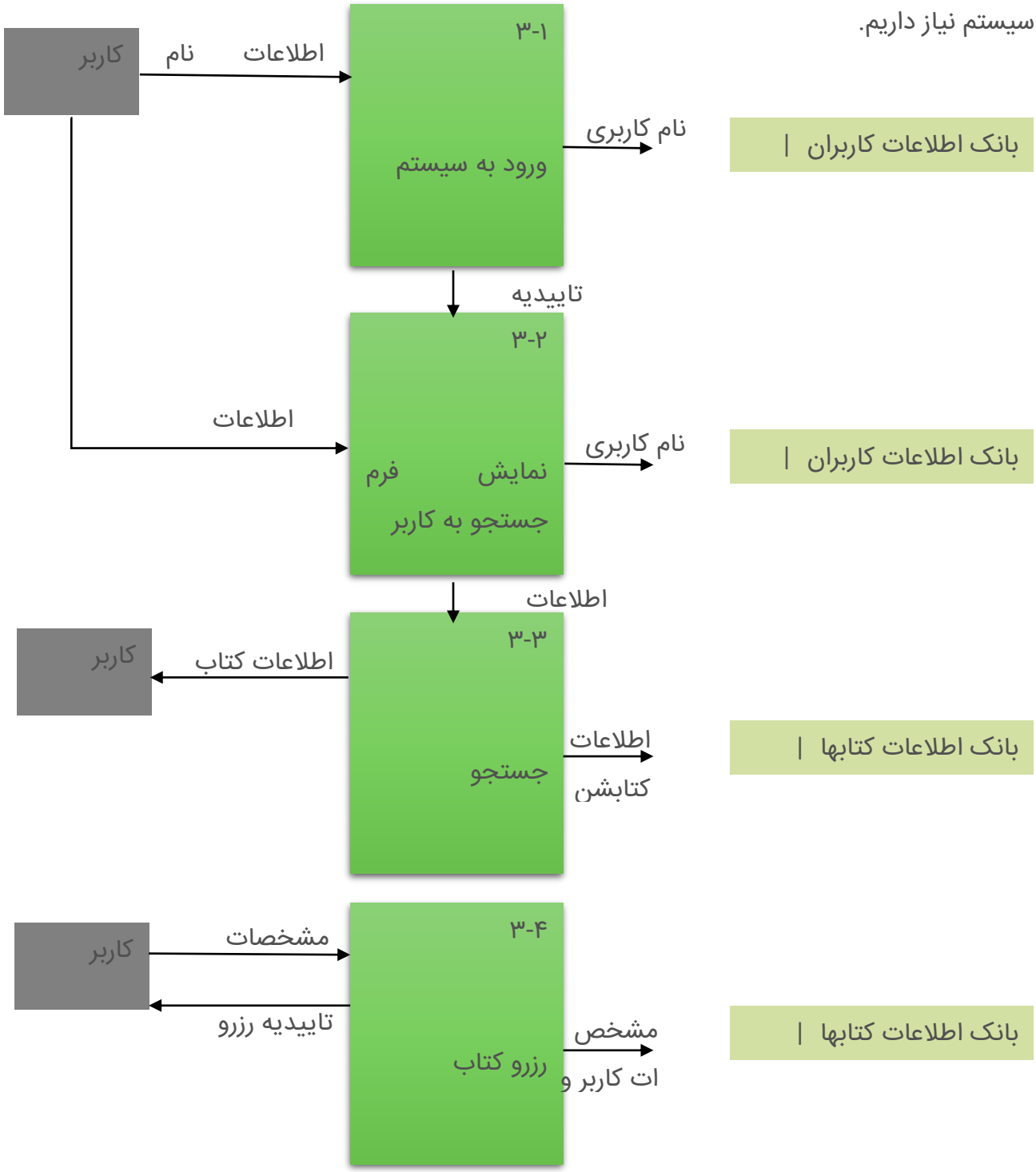
حال اینکه مثلا کتابدار چگونه تایید می کند اطلاعات را باید در تحلیل بدست بیاوریم. ممکن است بگوییم از روی لیستی از دانشجویان که بخش آموزش دانشگاه در اختیار کتابدار قرار می دهد؛ و کتابدار با بررسی لیست مطمئن می شود که اگر دانشجو عضو این دانشگاه هست، تایید می شود. اگر این گونه باشد جزء سیستم ما محسوب می شود، یعنی آموزش لیست را به کتابدار داده و کتابدار لیست را چک می کند، اما اگر قرار است سیستم ما متصل به سیستم آموزش باشد اینجا باید این فرآیند را نشان بدهیم.



در سطح۱ هرکدام از این فرآیندهای ۱و۲و۳ را به عنوان فرآیند رزرو کتاب می شکنیم. نکته ای که باید به یاد داشته باشیم در رزرو کتاب آنلاین، این است که عملا کتابدار نقشی ندارد.

به عنوان مثال در خانه نشستیم، وارد سیستم می شویم، جستجو می کنیم، اطلاعات کتابشناسی را استخراج می کنیم البته اگر کتاب در دسترس باشد، با یک کلیک کتاب x رزرو می شود.

ولی در ابتدا برای برای اینکه بتوانیم رزرو کنیم باید در سیستم login کرده باشیم، بنابراین به یک فرآیند ورود به سیستم نیاز داریم.



در این قسمت نشان می دهد کاربر در فلان تاریخ کتاب مورد نظر خود را رزرو کرده است و طبق قوانین شخص تا سه روز فرصت دارد که مراجعه کند و کتاب را تحویل بگیرد و در غیر این صورت مشخصات رزرو کتاب از حساب کاربری شخص پاک خواهد شد.

بعد از انجام مدل سازی، چند نکته باقی می ماند:

۱. **بررسی جامعیت:** جامعیت مدل را بررسی کنیم. فرض کنید مدل سازی تمام شد، باید دید کاری از قلم نیافتاده باشد. آیا تمام فرآیندهایی که مورد نیاز بود و در بخش تحلیل به آن رسیده بودیم در مدل سازی وجود دارد؟ بانک های اطلاعاتی وجود دارد؟ موجودیتها همگی هست؟
۲. **بررسی قوانین DFD**: Black Hole و Gray Hole نداشته باشیم. یا بررسی کنیم که فرآیندی از قلم نیافتاده باشد و یا موجودیتی مستقیم با data sture تعامل نداشته باشد و....
۳. **بررسی Balancing**: یعنی هر سطح را که به سطح های کوچکتر بشکنیم موجودیتی به اشتباه اضافه نشده باشد. تبادل اطلاعات و داده ها به صورت درستی صورت پذیرد. داده ی جدیدی به صورت اتفاقی خلق نشده باشد و...
۴. **قانون ۲۰-۸۰:** یعنی ۸۰ درصد مدل سازی با ۲۰ تلاش انجام می شود و برای تکمیل ۲۰ درصد باقی مانده در فرآیند مدل سازی باید ۸۰ درصد تلاش خود را به کار بگیریم. این مدل سازی را باید بارها بر حسب اطلاعاتی که از تحلیل بدست آورده ایم مرور کنیم. اگر چیزی از قلم افتاده باشد اضافه کنیم، اگر فرآیندی نیاز هست خرد بشه این کار انجام شود تا اینکه ۲۰ درصد باقی مانده انجام شود. (مثلا اگر در یک ساعت ۸۰ درصد کار رو تحلیل کردیم، چیزی حدود ۸ ساعت زمان باید بذاریم تا ۲۰ درصد باقی مانده را تکمیل کنیم)چرا؟ چون آن ۲۰ درصد باقی مانده چیزی نیست که بلافاصله به ذهن ما برسد. ما که شروع می کنیم در جریان پروژه پیش می رویم، به مرور آن موارد اشکالات خودشان را نشان می دهند و ما می توانیم آنها را پیدا کنیم و اضافه کنیم.

همچنین برای database هم به نمودار ER نیاز داریم.

باید برای تمام فرآیندها data dictionary بسازیم. یعنی بانک اطلاعاتی تعریف کنیم در این مثال ما کتاب ها یک بانک اطلاعاتی را تشکیل می دهد که اطلاعات جامع کتاب ها در آن قرار می گیرد و می توانیم به جای کتاب ها از item استفاده کنیم و تمام مواردی که ما می توانیم در کتابخانه امانت بدهیم مثل CD و مجله و..... این ها را توضیح می دهیم که چه اطلاعاتی هست و چگونه نگهداری می شود.

درباره کاربر همین تعریف را می گوییم. کاربران اعضایی هستند که طبق قوانین کتابخانه در کتابخانه عضو می شوند و مجاز هستند از خدمات کتابخانه استفاده کنند.

و در آخر هم Logical Modeling که Decision tree table و سودوکو و این موارد را باید بگوییم.

وقتی این کارها را انجام دادیم می توانیم بگوییم DFD را آماده کرده ایم.

خلاصه: پس از استخراج اطلاعات و مدل کردن سیستم باید راه حل هایی که برای انتخاب نرم افزار داریم را بشناسیم. یکی از توضیحاتی که در مهارت های تحلیل و تحلیلگری داده شد این بود که تحلیلگر باید به روز باشد و از فناوری ها اطلاع داشته باشد که به آن **مهارت های فنی** می گوییم و در این فصل راجع به این موضوع صحبت می کنیم.

منظور از عبارت development strategies، استراتژی های توسعه ی شرکت نیست و نباید با موضوع راهبردنویسی و استراتژی نویسی برای این شرکت یکسان دیده شود.

Development strategies یعنی استراتژی های توسعه ی نرم افزار. (استفاده مولف از software development strategies بهتر بود)

Outsourcing (برون سپاری)

به معنای این که هر نوع کاری که ما قرار است انجام بدهیم فقط مخصوص نرم افزار نیست. به عنوان مثال:

اگر نیاز داشته باشیم که همیشه یک نفر در محوطه دانشگاه حضور داشته باشد، به درختان آب بدهد، چمن ها را کوتاه کند و در کل باغبانی محوطه را انجام بدهد، این مقوله را Outsource می کنیم یعنی انجام این کار را به یک شرکت بیرونی واگذار می کنیم و دیگر نگران نیستیم که چگونه نیرو استخدام شود یا نیرو بیمه می شود و.... وظیفه ما فقط در تشریح سرویس مورد نیازمان است که در این مقوله جزئیات عبارتست از این که هر روز صبح چمن های یک قسمت به خصوص آبیاری شود، در طول یک هفته حداقل یک بار کوتاه شود و... که برای این مقوله قراردادی می نویسیم و بسته به قرارداد ماهیانه یا سالیانه مبلغ مورد نظر رو پرداخت می کنیم.

Outsourcing می تواند در لایه های مختلف اتفاق بیافتد. فرض کنید یک شرکت نرم افزاری-سازمان وجود دارد که بخشی از کار نرم افزاری را به جای این که In House انجام بدهد، این بخش از کار را در قسمت توسعه ی شرکت به یک شرکت بیرونی واگذار می کند. این همان رخ دادن در لایه های مختلف است، این که شرکت در لایه ی اول می تواند کار را بگیرد و تحویل بدهد.

در مدل دیگر شرکت مورد نظر کار را بین چند شرکت دیگر به عنوان شرکت های واسطه تقسیم می کند، پس از انجام کار را تحویل میگیرد، نصب می کند.

در مدل دیگر سازمان خودش کار را تقسیم کرده و به چند شرکت می سپارد و یک تیمی وجود دارد که کارهای گرفته شده را نصب می کنند

در کل outsourcing می تواند حالت های مختلفی داشته باشد.به طور کلی outsourcing به این صورت است که کار را تحت عنوان service provider به شرکت های بیرونی می دهیم، یعنی کسانی که راه حل های outsource برای ما پیشنهاد می کنند، یا این که کار را Application Service Providers یا ASP بسپاریم که کسانی هستند که در قالب پرداخت شارژ هزینه اپلیکیشن را به ما تحویل می دهند.

حالت دیگر این است که با Internet business services همکاری کنیم، کسانی که یک اپلیکیشن وب را برای ما آماده می کنند و به صورت transaction با آن ها کار می کنیم،(transaction مورد نظر همیشه مالی نیست هرچند بیشتر در این حوزه فعال است)مثل شرکت های نظیر آپ، ستاد یا به پرداخت ملت.

برای هزینه ی outsourcing سه روش را تصور می کنیم:

۱. fixed fee model: مدل هزینه ی ثابت باشد یعنی این که بر اساس یک هزینه ی ثابت سالیانه یا ماهیانه یا که در کل کار را به بیرون محول می کنیم و هزینه اش را پرداخت می کنیم. در این صورت بعد از تحویل گرفتن کار راه اندازی و به روزرسانی فعالیت موردنظر بر عهده ی خودمان است.
۲. subscription model: به معنای مدل اشتراکی (حق اشتراک)، یعنی شرکت بیرونی از ما تقاضای پرداخت ماهیانه را دارد، به عنوان مثال برای سایتی که در اختیار دارد تقاضای گرفتن ۲۰ دلار ماهیانه را از ما دارد و استفاده کردن یا نکردن ما از این سایت برایشان مهم نیست.
۳. Pay on_demand / Usage model or Transaction model: به معنای مدل داد و ستد یا مدل تراکنشی یا مدل استفاده به معنای این که به اندازه ی استفاده ی ما برایمان مبلغ در نظر می گیرند که باید آن مبلغ را بپردازیم.

به عنوان مثال اپلیکیشن arvan: از جمله شرکت هایی که برای وبسایت ها سرویس CDN ارائه می دهند. یک سری افراد هستند و معیارشان این است که یک سرویس را ارائه می کنند و ماهانه مبلغی را مطالبه می کنند. تقریبا مدلی پیدا نمی شود که CDN را ارائه کند، مبلغی را تقاضا کند و دیگر ارتباطی با ما نداشته باشد. معمولا یا روش دوم (حق اشتراکی)استفاده می شود یا مدل سوم، بدین صورت که با توجه به استفاده از CDN مبلغ مورد نظر را از ما تقاضا می کنند.

مدلهای دیگری نیز وجود دارد که **ترکیب این سه مدل** است. به عنوان مثال حق اشتراک طلایی که اگر این حق اشتراک را پرداخت کنیم (برای مثال ماهیانه صد هزار تومان)به یک سری سرویس ها دسترسی پیدا می کنیم یا حق اشتراک نقره ای (ماهیانه هشتاد هزار تومان)و می توانیم به یک سری سرویس دیگر دسترسی پیدا کنیم

قیمت گذاری به این شکل بسیار درست است، که حداقل ۳ سطح قیمت گذاری را ارائه بدهیم و مشتری حق انتخاب داشته باشد و اکنون جزئیاتی وجود دارد که می توانیم با توجه به این جزئیات استراتژی هایی قرار دهیم که به عنوان مثال همه ی کاربران به سمت اشتراک نقره ای تمایل داشته باشند. یعنی کاری کنیم که ۸۰ درصد کاربران یا اعضا پرداخت نقره ای داشه باشند؛ روش راحتی به نظر می رسد.

چرا همه ی توسعه ها را outsource انجام نمی دهیم؟

هنگامی outsource را انجام می دهیم که برای ما جذابیت هزینه ای داشته باشد (cost attractive)و همچنین reliable باشد یعنی سرویسی که برای ما فراهم می کنند قابل اطمینان باشد. چون در روش های outsourcing ممکن است داده در اختیار شرکتی که پیاده سازی را انجام می دهد قرار می گیرد بنا بر این بسته به نوع حساسیت داده هایی که در سازمان وجود دارد ممکن است روش outsourcing کلا کنار گذاشته شود یا این که فقط به بخش هایی که نیاز به دسترسی مستقیم ندارند محدود می شود. یا این که مدیریت شوند و برای مثال پروژه های دفاعی پروژه های محرمانه و حساسی هستند ولی outsource می شوند. پروژه های نظامی مدت زیادی در ایران outsource نمی شدند و در نتیجه پیشرفت نظامی هم کم بود ولی از زمانی که تصمیم گرفته شد با دانشگاه ها و شرکت های دانش بنیان همکاری شود عملا سرعت پیشرفت بالا رفته است ولی حواسشان به mission critical نیز هست یعنی اول ماموریت های کلیدی مدیریت می شود، یعنی افرادی که در وزارت دفاع هستند پروژه ها را خرد میکنند و هر کدام از پروژه ها را به یک شرکت مجزا واگذار می کنند در نتیجه هر یک از شرکت ها به تنهایی متوجه برنامه کلی که قرار است اتفاق بیافتد نمی شوند.

هنگامی که پروژه ها به وزارت دفاع تحویل داده می شوند با یکدیگر assemble یا intergrade می شوند و وزارت به محصول یا نرم افزاری که انتظارش را داشته می رسد. اگر outsourcing خارج از مرز ها اتفاق بیفتد (overseas)می تواند مسائلی مانند کنترل، فرهنگ، ارتباطات فرهنگی و مسائل امنیتی را همراه خود داشته باشد.

اگر بخواهیم کار outsource را انجام بدهیم، باید تاریخچه شرکت یا firm's که کار outsource را برای ما انجام می دهد را به دقت مطالعه کنیم؛ این که این شرکت چه تجربه ای دارد، چه کارهایی را قبلا انجام داده است، آیا در کارنامه شان خوش قولی یا بدقولی وجود داشته، کارهایشان بر اساس قوانین جلو رفته یا سعی کرده قوانین را دور بزند، مسائل مالی و... را بررسی می کنیم.

بحث امنیت شغل کارکنان شرکت نیز مهم است؛ این که وقتی داریم کارها را به بیرون outsource می کنیم یعنی عملا داریم افرادی که این کار را در سیستم ما انجام می دادند را حذف می کنیم. باید به این سوال پاسخ بدهیم که از دست رفتن این شغل ها در سیستم برایمان مهم است یا خیر.

Offshoring

یک مدل Global Outsourcing هست. یک نمونه ی مهم از مایکروسافت، یکی از مهمترین شرکت هایی که کارش را به صورت offshore انجام می دهد، به این معنی که ما پروژه را در بخش توسعه ی پروژه های IT، پشتیبانی و حتی بخش های عملیاتی به یک کشور دیگر واگذار کنیم.

مایکروسافت در سیاتل آمریکا است ولی بخش زیادی از ساپورت و حتی بخش اندکی از توسعه اش در کشورهایی نظیر هند و اسرائیل و... اتفاق می افتد. یعنی اگر ما یک کاربری باشیم که از فیلادلفیا با مایکروسافت

تماس می گیریم و مشکل سیستم که مثلا میتونه به مشکل خوردن نرم افزار یا سیستم باشه رو بهشون اطلاع بدیم، متوجه می شویم که کسی که پشت خط هست از هند به سیستم ما ریموت می شود و یک سری اصلاحات را بر روی سیستم ما انجام می دهد یعنی کار را outsource کرده و نه تنها outsource بلکه offshore کرده یعنی به یک کشور دیگر سپرده است.

معمولا کشورهایی گران هستند و نیروی کار در آن جا گران است ولی outsource می رود به سمت کشورهایی که نیروی کار در آنجا ارزان می باشد. نمی توان به طور مستقیم گفت که شرکتی مانند اپل عمده ی تولید محصولاتش را outsource کرده، چون نیروهای کار متعلق به شرکت اپل هستند ولی در چین مستقر هستند و کار می کنند. بنابراین هزینه های کم و این گونه مسائل باعث می شود که شرکت ها کار را offshore کنند ولی این کار همراه با ریسک است، به این صورت که به جای این که روی اقتصاد کشور خودشان تاثیر بگذارند عملا بر روی اقتصاد کشور دیگری تاثیر می گذارند که شامل پرداخت مالیات و حقوق به کشور دیگر می شود.

کنترل پروژه ساده نیست و مسائل ایمنی و امنیتی در آن مسئله جدی است. همچنین تفاوت فرهنگی نیز وجود دارد و به هر حال ابزارهایی برای ارتباطات موثر لازم است که این روزها این موضوع با نام Effective Communication با وجود ابزار هایی مانند skype و slack و anydesk و... تا حد زیادی حل شده است.

Software as a Service

یک مدل دیگر از راه حل هایی که استفاده می کنیم برای نرم افزار هاست. یعنی نرم افزار به عنوان یک سرویس چه کار می کند؟ یعنی کار را outsource می کنیم؟(کا را به بیرون می دهیم که انجام دهند)پس پکیج آماده نمی خریم، در خانه انجام نمی دهیم و offshore نمی کنیم، بلکه از شرکت هایی که قبلا روی این مسئله کار کردن و به مسائل و موضوعاتی که اکنون دغدغه ما هست فکر کردند و راه حل ارائه داده اند و نرم افزار نوشتند سرویس میگیریم و بابت سرویسی که می گیریم پول پرداخت می کنیم. این کار طبیعتا مستلزم این است که نیاز های ما به شکل یک بسته ی جامع در بیرون وجود داشته باشد. برای همه ی نیاز های نرم افزاری این روش لزوما روش درستی نیست. طبیعتا شرکت ها و استارت آپ ها، کارهایی رو به عنوان Software ad a Service پیاده می کنند تا خیالشان از بابت این که userهایشان در دنیای بیرون زیاد است راحت باشد یعنی بدانند که حداقل بیشتر از ۱۰ هزار کاربر وجود دارد که استفاده خواهند کرد و بعدا به ازای هر کاربر مبلغی (دلار)پرداخت می کنند. برای مثال به ازای ۱۰ هزار کاربر ۱۰ دلار پرداخت می کنند یعنی ۱۰^۵ دلار در ماه دریافت می کنیم. اگر مطلع باشیم که چنین بازاری وجود دارد به سمت آن گرایش پیدا می کنیم.

نقش تحلیلگر

اکنون ما یک سری استراتژی را نام بردیم. اکنون چه کسی تصمیم می گیرد که کدام یک برای سازمان ما مفید است؟ مثلا می خواهیم برای شهرداری نرم افزار بنویسیم. کدام را انجام بدهیم؟ In House؟ Outsource؟ و...در این جا تحلیلگر سیستم باید کمک کند.

نکته: فکر نکنید که موضوع Outsourcing فقط بین یک سازمان و یک شرکت وجود دارد. خیلی اوقات بین شرکت به شرکت وجود دارد. یعنی یک بیزینس بزرگی که خودش توسعه دهنده نرم افزار هست برای مثال یک پروژه ای را از وزارت نفت گرفته است و این پروژه را بین ۴۰ شرکت کوچک کامپیوتری تقسیم می کند و در واقع خودش نیز Outsource می کند. یا حتی ممکن است یک بخش را Outsource کند، یک بخش را خودش بنویسد و یک بخش را هم از جایی خریداری کند یا حتی به عنوان سرویس در سیستم قرار دهد. بنابراین ترکیب تمام موارد گفته شده محتمل است.

ما می توانیم کارشناس IT سازمان باشیم یا این که تحلیلگر سازمان باشیم و این تصمیم را بگیریم بنابر این نقش آنالیز تفاوت چندانی ندارد. کاری که باید انجام بدهد این است که بین راه حل های ۱ تا ۶ گفته شده باید آنالیز کند و تحلیل کند و بررسی کند که هزینه و فایده ی هر کدام از راه ها چیست. آیا محدودیت هایی وجود دارد که نتوانیم یک مدل را انتخاب کنیم؟ ما سود و فایده ی هر کدام را بررسی می کنیم و نهایتا برای مدیر عامل شرکت، رئیس سازمان و تیم تصمیم گیرنده کار را ارائه بدهیم و نهایتا تصمیم بگیریم که کدام یک از موضوعات را انجام بدهیم.

در واقع نقش تحلیلگر سیستم این است که برحسب نیاز های فعلی و نیاز های آینده کار ارزیابی را انجام بدهد.alternative به همین معناست، یعنی راه حل ۱ و۲ را داشته باشیم و بدانیم که راه حل ۱ بدیل راه حل ۲ هست یعنی می تواند جایگزین شود و ما انتخاب می کنیم که کدام یک بهتر است(Evaluation and Seletion). این امر کار ساده ای نیست.

ما باید بسته به بزرگی پروژه یک تیم تشکیل بدهیم. برای مثال در مقیاس شهرداری ۷ الی ۸ نفر از دیدگاه های مختلف جمع شده اند، ترکیبی از افراد دانشگاهی، افراد اجرایی داخل خود سازمان، شرکت ها را بررسی کردند، کارشان را مشاهده کردند و تصمیم گرفتند کار به صورت In House انجام شود. اشکال کار کجاست؟ تصمیم گرفتیم به دلیل این که اصلا شدنی نیست کنار بگذاریم، feasible نیست یا هزینه ی زیادی برایمان دارد و تصمیم گرفتیم Outsource انجام شود، چطور پکیج هایی خریداری شود و همه ی این مطالب رو محاسبه می کنیم.

Analyzing Cost and Benefits

در روش های تحلیل سود و فایده و هزینه یکی از نکاتی که باید به آن توجه کرد روش financial analysis tools هست. یعنی ابزار هایی برای ما وجود دارد تا بتوانیم تحلیل مالی انجام بدهیم.حداقل ۳ روش برای توجه کردن توصیه می شود.

روش۱: payback analysis یا تحلیل باز پرداخت

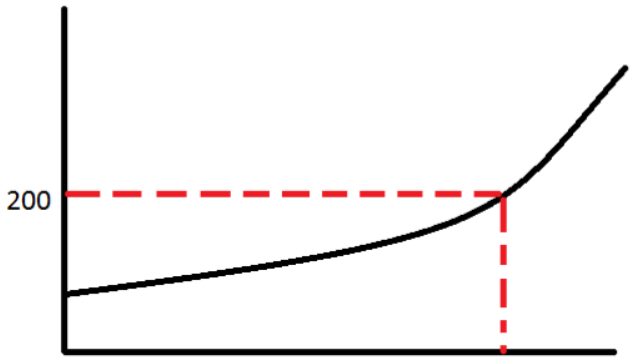
روش۲: return on investment (ROI)یا تحلیل بازگشت سرمایه

روش۳: net present value (NPV)یا ارزش خالص سود

Payback analyses

باید بگوییم سیستم اطلاعاتی که می خواهیم توسعه بدهیم چقدر هزینه ی توسعه نیاز دارد و چه جاهایی می توانیم هزینه را کاهش بدهیم و سود را افزایش بدهیم. مانند این است که بگوییم برای رستوران می خواهیم سفارش آنلاین داشته باشیم در غیر این صورت باید یک نیروی کار می گرفتیم که این نیروی کار هزینه در بردارد و اکنون با کنار گذاشتن این نیروی کار هزینه را پایین آورده و مزایا را بالا می بریم.

(ROI) Return on investment



روش متداولی است. بر روی این مسئله تمرکز می کند که ما چقدر هزینه می کنیم، چقدر سرمایه گذاری می کنیم، در چه مدتی؟ مثلا در ۳ ماه اول به عنوان مثال ۲۰۰ میلیون سرمایه گذاری می کنیم تا نرم افزار مورد نظر توسعه پیدا کند. چقدر طول میکشد تا این ۲۰۰ میلیون بازگردد؟ برای مثال سودی که می گیریم به این شکل است:

به نقطه ی ۲۰۰ میلیونی می رسیم که هزینه کرده بودیم و الان به همین مبلغ رسیدیم و به آن نقطه ی سر به سر یا break even point گفته می شود که این نقطه بسیار مهم است. احتمالا در تعداد زیادی از بیزینس ها می توانیم نقطه ی سر به سر را پیدا کنیم. اگر فردی ۲۰۰ میلیون هزینه کند و قرار باشد ۲ سال این مبلغ بماند و سپس آرام آرام سود کند، بعد از ۲ سال به نقطه ای می رسیم که که ۲۰۰ میلیون اولیه مان برگشته و بعد از آن سود خواهد داشت.(نمودار لزوما به صورت شکل بالا

نیست.)ولی ممکن است نقطه ی سر به سر را ۲ ماهه اعلام کنیم که شرایط خیلی خوبی است. هر چقدر نقطه ی سر به سر در مدت کوتاه تری اتفاق بیفتد کسب و کار مورد نظر امیدوار کننده تر خواهد بود.

(NPV) Net present value

به معنای ارزش فعلی خالص، در واقع حاصل تفاضل سود به دست آمده از کل هزینه ها.

از موارد دیگری که باید در کارهایمان باشد و به آن توجه کنیم لیست کردن روش ها یا استراتژی های مختلفی است که برای توسعه وجود دارند و هزینه و مزایای هر کدام از این موارد را حساب کنیم. برای هزینه معمولا دو مدل هزینه داریم، هزینه ی ملموس و هزینه ی غیر ملموس(این کار یک نوع تقسیم بندی می باشد)

هزینه ی ملموس: همان مقدار پولی که خرج می کنیم.

هزینه ی غیر ملموس: اعتمادی که باید از نیروی کار داخلی جلب کنیم یا بحث رقیب تراشی.

باز هم می توان به هزینه ی یکباره و هزینه ی تکراری نیز تقسیم شود. یعنی یک بخشی از پول را باید یکباره هزینه کنیم، مثلا تجهیزات بخریم تا شرکت راه بیفتد و بلافاصله پس از آن هزینه های تکراری نیز داریم، یعنی هزینه هایی که ماهیانه باید صرف کنیم مانند حق اشتراک نرم افزار، پول آب و برق و.... به عنوان یک تحلیلگر باید به همه ی موارد فوق توجه کنیم.

باید به این فکر کنیم که استراتژی ای که می خواهیم انتخاب کنیم با توجه به رشد آینده ای که سازمان خواهد داشت، آیا استتراتژی مناسب هست یا خیر؟ scability آن چقدر است؟ چقدر راه حلی که انتخاب می کنیم قابلیت توسعه و بزرگ شدن دارد؟ این که هزینه های ساپورت و پشتیبانی از نرم افزار و سخت افزار چقدر است؟ باید این موارد را بررسی و تحلیل کنیم.

برای مثال گزینه های لایسنس نرم افزار را بررسی کنیم، این که بتوانیم هزینه های مالی هر کدام از alternative هایی که داریم را انتخاب کنیم. (موارد ۱ تا ۶)هر کدام هزینه هایشان چقدر است؟ از نظر مالی چقدر برای ما تمام می شود؟ برای مثال یک نرم افزار را در ابتدا باید یک میلیارد تومان بخریم یا ماهیانه ۵۰ میلیون به عنوان حق اشتراک بدهیم؟ نهایتا نتایج را مقایسه و مطالعه می کنیم.

The Software Acquisition Process

به معنای فرآیند به دست آوردن نرم افزار می باشد. در این جا بحث توسعه نیست و بحث خرید است.

Step1: قرار شد تحلیلگر راه حل های مختلف را لیست کند، هزینه های هر کدام، فایده هایشان و مشکلاتشان را مشخص کند. لازم است که نیازمندی های سیستم خودمان را بیشتر بشناسیم، مطلبی که بعدا تحت عنوان

RFP عنوان می کنیم و مستلزم این است که ویژگی هایی در نرم افزار به دنبال آن ها هستیم را به خوبی استخراج کنیم و آن ها را یاد بگیریم؛ مشخص کنیم که کارمان قرار است تحت شبکه یا web base باشد.

برآوردی از رشد آینده ی شرکت یا محصول یا چیزی که قرار است به دست بیاوریم را داشته باشیم.

سخت افزار و نرم افزار و نیروی پسنلی یا محدودیت هایی که در حوزه ی پرسنلی وجود دارد را مشخص کنیم.

یک RFP یا RFQ اعلام کنیم و آن را در اختیار شرکت هایی که احتمالا ما آن ها را از قبل می شناسیم تحت عنوان IT Service List قرار بدهیم. این ها شرکت هایی هستند که در حوزه ای که ما به دنبال آن هستیم برای مثال کار کرده اند و تجربه دارند، بنابراین به آن ها تحویل می دهیم و از آن ها می خواهیم که پروپوزال برای ما ارسال کنند.

Step3: علاوه بر این که آن ها می توانند توسعه بدهند باید به این فکر کنیم که کدام شرکت ها پکیج را به صورت آماده دارند و میتوانند به ما بفروشند، یا استفاده از روش های Outsourcing.گزینه هایی که ما می توانیم کار ها را به شرکت های بیرونی بسپاریم کدام اند و چرا؟ چون قرار است تصمیم بگیریم که ما داخل سازمان، بیرون سازمان، خارج از کشور توسعه بدهیم یا به دنبال خریدن پکیج یا سرویس برویم. روش های به دست آوردن عارت اند از:

۱. سرچ کردن در اینترنت و پیدا کردن شرکت ها
۲. با شرکت ها یا افراد مطلع در این زمینه مشورت کنیم.
۳. در نرم افزار های آنلاین سرچ کنیم و درباره ی زمینه ی کاریشان مطالعه کنیم و بفهمیم که سازمان هایی که از نرم افزار خاصی استفاده کردند رضایت داشتند یا خیر؟

Step3: به طور کلی لازمه ی ما روش های مختلفی است که ما راجع به آن ها صحبت کردیم. Alternative ها را بررسی کنیم و ارزیابی کنیم. مستلزم این است که جست و جو کنیم، ارتباط بگیریم، و برای مثال متوجه بشویم که برای یک پکیج به خصوص چند کاربر وجود دارد؟ آیا این اپلیکیشن به خصوص تست مورد نظر را پشت سر گذاشته است؟ آیا می توان یک نسخه ی دمو از آن داشته باشیم و آن را ارزیابی کنیم؟ یک سری معیار مشخص کنیم که پکیج مورد نظر را بر اساس آن ها ارزیابی کنیم و عملا همه ی این مطالعات را باید در اختیار مجموعه ی تصمیم گیرنده به عنوان تحلیلگر قرار بدهیم و نهایتا تصمیم بگیریم.

راه حل های مختلف، هزینه ها، فایده ها، محدودیت ها، و موانع را بررسی کردیم، این که هر کدام چقدر هزینه برای خرید لازم دارند و این که اگر In House پیش برویم چقدر برای ما هزینه دارد و این که آیا این اپلیکیشن در بازار وجود دارد و تعداد کاربرانش چقدر است و در کل ارزیابی کنیم که مفید بوده یا نه؟ موارد گفته شده کارهایی است که ما باید انجام بدهیم.

Step4: باید تحلیل هزینه و فایده انجام بدهیم که چه سودی برای ما دارد، این که اگر بخواهیم مالکیت نرم افزار را داشته باشیم(total cost of ownership)، هزینه ی مالکیت، این که عملا مالکیت نرم افزار رو برعهده داشته باشیم وحق کپی رایت در اختیار ما باشد چقدر می شود.

Step5: باید یک توصیه نامه از گزارشمان آماده کنیم که که توصیه می شود با توجه به هزینه و فایده و مزایا و معایب کار را Outsource کنیم به یک شرکت مورد نظر یا از بین شرکت های موجود یکی را انتخاب کنیم.

Completion of System Analysis Tasks

تکمیل وظایف تحلیلگر سیستم: ما باید نیازمندی ها را مستند سازی کنیم. چیزی که تحت عنوان RFP آماده می کنیم بخشی از این مستند سازی است. نیازمندی ها را برای سیستم جدید مشخص می کنیم. Alternative هایی که بررسی کردیم را ذکر می کنیم، توصیه های خود را به مجموعه ی مدیریت می گوییم، سعی می کنیم قرارداد های مشابه که در این زمینه وجود دارند را آماده کرده و جزئیات آن ها را توضیح بدهیم و سعی کنیم مستند سازی را خیلی خوب سازماندهی کنیم و آن را قابل فهم، مختصر و مفید گردآوری کرده و نهایتا کار را به مجموعه ی تصمیم گیرنده ارائه می دهیم.

برای ارائه دادن باید سعی کنیم مرور مختصری داشته باشیم درباره ی موضوع بحث و یک خلاصه ای از alternative ها و روش های مختلف را بتوانیم توضیح بدهیم و شرح بدهیم که چرا یک alternative به خصوص را انتخاب کنیم. سوالاتی از قبیل چرا، هزینه، فایده، امنیت و... را پاسخ بدهیم و زمانی را برای بحث و نظر دادن دیگران قرار بدهیم. ممکن است یک توصیه ما را قبول نداشته باشند و ما برای این توصیه دلیل بیاوریم و توضیح بدهیم و درآخر بتوانیم تصمیم نهایی را بگیریم.

بعد از این که این کار را انجام دادیم عملا داریم شرایطی را فراهم می کنیم که کار تحلیل سیستم داخل سازمان می باشد.

ما به عنوان کارفرما وظیفه داریم که اگر پروپوزالی رسید و تصویب شد یا طراحی پروپوزال که راجع به آن صحبت کردیم که شامل logical and physical design است.

فصل ۷

نکاتی درباره Start Up

کسب و کارهایی که در حال حاضر کار میکنند تحت عنوان شرکت یا کمپانی، بسیاری از موضوعات را میشناسیم. یک تجربه ای در حوزه شرکت داری و کمپانی شکل گرفته است در ذهن مردم و این ذهنیت را دارند به استارتاپ ها منتقل می کنند و جالب است بدانید که این ذهنیت در حوزه استارتاپ کار نمی کند.

وقتی کسی بخواهد استارتاپ راه اندازی کند یکی از توصیه هایی که به آنها میکنند این است که برنامه کسب و کار یا Busines Plane برای خودتان بریزید، اما آیا برنامه داشتن برای راه اندازی کسب و کار کافی است؟ اینکه ما برنامه داشته باشیم و کسب و کار را راه اندازی کنیم صد البته که لازم است، اما برای شروع کافی نیست. یک Plane چیزی نیست که بتوانیم با آن استارتاپ راه اندازی کنیم. این plan برای شرکتها و بیزنس های بزرگ احتمالا میتواند جوابگو باشد ولی برای استارتاپ ها خیر.

تاریخچه

تقریبا نخستین شرکتی که به مفهموم کمپانی که امروزه میشناسیم راه اندازی شده یک شرکت هلندی بود که با عنوان هند شرقی در سال ۱۶۰۲ راه اندازی شد و تجارتی بین هند و هلند برقرار کرد، در همان زمانی که مستعمره ی اروپا بوده اند.

در حال حاضر۳۰۰ سال تجربه شرکت داری منقل شده به نسلی که می خواهند استارتاپ راه اندازی کنند. در سال ۱۹۰۸ گروهی از تحصیل کنندگان هاروارد دوره ای را تحت عنوان MBA راه اندازی کردند که بر بحث مدیریت کسب و کارها متمرکز است و سعی کردند مطالبی را آموزش بدهند که ابزارهای مورد نیاز برای رشد شرکت چیست؟ حسابداری را آموزش دادند، استراتژی، رفتار سازمانی، منابع انسانی، مدیریت و ابزارهایی برای مدیریت شرکت و کارخانه نیازداریم.

اما واقعیت این است که این مهارت ها برای راه اندازس استارتاپ مناسب نیست. یکی از تصورات غلط این است که ما فکر می کنیم که استارتاپ ها همان نسخه کوچک شرکت ها هستن! یعنی در گام های اولیه و نوپا بودن را به مفهوم نو پا بودن شرکت تلقی می کنیم، یعنی می گوییم شرکت بخش حسابداری دارد، بازاریابی، بازرگانی، فنی، و... دارد. باید همه بخش ها را داشته باشیم ولی در هر کدام از این بخش ها اگر شرکت بزرگه مثلا ۱۰ نفر نیرو دارد، ما یک نفر نیرو داشته باشیم، اصطلاحا Smaller Version شرکت های بزرگ است؛ ولی واقعا اینطور نیست و این یک تصور غلط است.

این ابزارا و این منابع در دوره MBA اوایل قرن ۲۰ راه اندازی شد. این ابزارها وجود داشتن و معرفی شدن اما این ابزارها و سرفصل های در نظر گرفته شده برای عصر آشوبناک و پر تلاطم امروزی کافیه؟

قرار است به این برسیم که این تصور که استارتاپ نسخه کوچک شرکت بزرگ است، غلط است.

یکی از کارهایی که به طور مستمر در کارهای استارتاپ باید انجام بدهیم موضوع سرچ است، در حالی که در شرکت های بزرگ مهم ترین فعالیت اجرا است. و یک تصور غلط دیگه این است که فکر می کنیم با یک برنامه می توانیم کسب و کار راه اندازی کنیم، به همین دلیل هم میگویند اول BP بنویسیم، در صورتی که در باره استارتاپ ها این موضوع ممکن است صادق نباشد.

برنامه ها در یک محیط عدم قطعیت کسب و کارهای نوپا ماندگار نیست، یعنی اگر می خواهیم یک برنامه(BP)را تنظیم کنیم و بر اساس اطلاعاتی که فکر می کنیم از بازار به دست آوردیم، این اطلاعات ایستا نیستند؛ مثل این است که یک سیب را می خواهم روی آن هدف گیری کنم. اگر اول سیب ثابت باشد احتمالا محاسبات رو انجام میدهیم که تیر را با چه شدتی و زاویه ای پرتاب کنم که احتمالا به هدف میخورد و این احتمالا میتواند انجام بشود. اما وقتی که آن سیب دائما در حال حرکت است، من نمیتوانم با همین محاسبات ساده این اتفاقات را رقم بزنم، باید خودم را دائما با این شرایط داینامیک و پویا تغییر بدهم. پس Plan ها در عصری که همه چیز مدام در حال تغییر است بنابراین استارتاپ ها نمی توانند با آن برنامه هایی که قبلا برای شرکت های بزرگ ریخته شده است جلو بروند و باید قبل از برنامه، برنامه ریزی انجام بدهیم Planning Comes Before The Plane باید Planning را یاد بگیریم، صد البته که ما برنامه می خواهیم اما این برنامه را با یه سری ویژگی های متفاوت است که آن را از قبل میدانستیم.

ممکن است سوال پیش آید که برنامه برای شروع استارتاپ خوب نیست پس چه برنامه ای خوب است؟ باید از بیزنس مدل کانواس استفاده کنیم. یعنی بیایم یک مدل کسب و کار تنظیم کنیم که این BMC در واقع مخفف Business Model Canvas است که چندمزیت دارد:

BMC فکر ما را سازماندهی می کند و باعث می شود به صورت تقسیم شده فکر کنیم.

برای تکمیل کردن آن باید از ساختمان یا دفترمان بیرون برویم در صورتی که بسیاری از افراد ممکن است بنشینند و BMC خود را تکمیل کنند و بر اساس برنامه ای که ریخته اند جلو بروند. اما در آخر داستان شکست میخورند! خیلی از استارتاپ ها به دلیل فنی شکست نخورده اند بلکه به خاطر بی توجهی به نیازهای مشتری شکست خورده اند.

BMC شامل ۹ بخش است:

- شرکای کلیدی
- فعالیت های کلیدی
- گزاره ارزش(اینکه محصول و اون خدمتی که داریم عرضه میکنیم چه ارزشی برای مشتری ایجاد میکند)
- منابع
- ارتباط با مشتری
- چند نوع مشتری داریم، چند نوع مخاطب داریم و چه تایپی هستن
- کانالهای ارتباطی با مشتری(فیزیکی محصول رو ارائه میدیم یا از طریق سایت، اپ و...)
- جریان بازدهی و مدل درامد
- هزینه های لازم

استارتاپ ها یک بخش مهمی به نام سرچ و یک بخش به نام اجرا دارند. نکاتی که درباره شرکت ها در تمام این سالها تجربه شده اصلا تجربه ی بی ارزشی نیست. و این هایی که موفق می شوند بخش دستاوردشان را مدیون این تلاشهایی هستند که قبلا شکل گرفته اما برای استارتاپ ها از متد دیگری باید استفاده کنیم، از BMC استفاده کنیم وقتی این را تکمیل کردیم و فرآیند سرچ مان را به یک بلوغی رساندیم آن وقت به مرحله اجرا می رویم، در مرحله اجرا جایی هست که برنامه خود را اجرا می کنیم و عملیاتی می کنیم و یک سری پیش بینی مالی داشته باشیم، برنامه ریزی می کنیم، بازاریابی می کنیم و...

فرآیند اجرا

قبلا برداشت این بود که یک مفهوم و ایده را مطرح می کنیم و سعی می کنیم ایده را توسعه بدهیم، بعد آن ایده را تست و ارزیابی کنیم و نهایتا اجرا کنیم. بعد متوجه شدند که این مدل خطی فکر کردن مردود است و عملا نمی توانیم بگوییم ایده را توسعه می دهیم. حالا رسیدیم به جایی که می توانیم اجراش کنیم و تمام، واقعیت این است که در خصوص فرآیندهای تکراری گفتیم که تفکر خطی و اینکه فکر کنیم تحلیل تموم می شود و بعد به ترتیب سراغ طراحی و پس از اتمام سراغ پیاده سازی و تست و نگهداری سیستم می رویم در عمل سیستم تا این اندازه خطی نیست، یعنی سیستم که تمام بشود، شناخت ما از سیستم به سادگی تمام نمی شود، به نحوی قانون ۸۰-۲۰.

یک بعد دیگر هم اضافه کنیم تا بیاییم در بعد دیگر که کانسپت و گسترش و تست لانچ و بحث های بازاریابی و فروش، توسعه کسب و کار و توسعه محصول را هم در یک بعد دیگری ببینیم و به این ترتیب سعی کنیم توسعه

ی محصول را در دو بعد X و Y گسترش بدهیم. با این اجرا، جواب این است که این مدل نمیتواند پاسخگو باشد، تجربه نشان داده که برای استارتاپ ها پاسخگو نیست.

فرآیند آبشاری

نیازمندی ها را شناسایی می کنیم در فاز تحلیل و طراحی انجام میدهیم پیاده سازی می کنیم، تست می کنیم و بعد نگهداری می کنیم، یک روش غلط است چون ما متوجه شدیم که نیازمندی ها را می توانیم کاملا شناسایی کنیم، بعد طراحی، پیاده سازی و... را انجام می دهیم.

میدانیم که بیشتر فرایندهای نوپا از نبود مشتری شکست خوردند تا توسعه محصول، یعنی اگر نگاه کنیم میبینیم فلان استارتاپ یک ورژن از محصول خودش را ارائه داده، هرچند که ویژگی هایش کامل نبوده باشد. بنابراین مشکل فنی کمتر گریبان گیر استارتاپ ها بوده است. اما وقتی محصول عرضه شده کسی حاضر نبوده پولی بده تا از این سرویس و محصول استفاده کند. پس اشکال به این نیازی که از محصول احصاء کردیم درست نبوده، برمی گردد. اصطلاحا بخش required engineering همان تحلیل اشکال دار است. چون به عنوان مثال من با خودم را جای مشتری گذاشتم و برداشت کردم و جلو رفتم و بخش فنی رو توسعه دادم که معمولا ضعف افرادیه که خیلی فنی کار می کنند مسلط هم هستن اما رویکرد ارتباط با مشتری رو خیلی بهش دقت نمی کنند.

هدف این فصل

فرایندی رو دنبال کنیم که از روش هایی برای مدیریت در یک مشتری استفاده کنیم عده‌ای می گویند برای اینکه مدیریت ریسک را کنترل کنیم یا ریسک را مدیریت کنیم؛ با مشتری دائماً تماس بگیریم اما واقعاً روش رسمی برای مدیریت یک مشتری چیست؟ اینکه بفهمیم مشتری های ما چه چیزی واقعا نیاز دارند. آیا روش تلفنی کفایت میکنه؟ ممکن پشت خط با کسی صحبت کنیم و جوابهای سربالا بده و بگوید که اگر به اینصورت است من واقعا استفاده می کنم ولی وقتی که پای عمل میرسد می بینیم که این روشی نبود که بتوانیم بهش اطمینان کنیم

مدل پیشنهادی

توسعه مشتری: گام سرچ یک فرآیند تکراری هست که بخش customer, discovery, validation شامل می شود و شناسایی مشتری و کشف مشتری است. یعنی بریم بیرون ببینیم چه کسانی میتوانند مشتری بالقوه محصول ما باشند. اینجا از معیارها به صورت رفت و برگشت استفاده می کنیم و سعی می کنیم اطلاعاتی رو که در مراحل اولیه جمع می کنیم اصلاح و تکمیل کنیم و عملاً مدل BMC بخش‌هایی مثل کانال relationships,

customer, segmentation را از این قسمت ها تکمیل می کنیم. یک بخش هم بخش اجرای جایی که ما customer creation می گوییم است و در نهایت company building رو داریم یعنی به سمت این که مشتری رو ایجاد کنیم پیش می رویم نیاز را احصاء کردیم و خدمت برای آن ارائه کردیم. حال سراغ ساخت کمپانی می رویم یا شرکتمان را به گونه ای ساختارش را می چینیم که جایی که عملاً از استارت آپ خارج میشویم استارتاپ همین اول شروع شود مفهوم آن است. قرار نیست یک بیزنس مادام العمر استارتاپ بمونه قرارست به شرکت تبدیل شود.

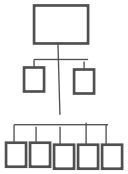
روش های چابک، روش های خوبی برای توسعه محصول هستند مدل پیشنهادی برای مدیریت مشتری یا مدیریت ریسک بود در این قسمت مدل پیشنهادی برای توسعه محصول، مدل‌های چابک است:

در واقع مدل های چابک مدل های هستند که سعی می‌کنند روایت های کاربران را دریافت کنند و برنامه‌ریزی کنند برای آن نیازهایی که کاربران اعلام می‌کنند و در یک فرایند رفت و برگشتی یعنی فرایندی که تکراری ست. منظور توسعه محصول است که اتفاق می افتد در این فرایند محصول را توسعه می‌دهیم و به مشتری میدهیم که آزمون پذیرش را از مشتری بگیرند و دوباره سراغ گام های بعدی بروند یعنی ما نهایتاً در این فرآیند نسخه اولیه از محصول را ارائه می دهیم که خود این فرآیند یک فرایند تکراری هست مدلش مثل اسکرام یا xp یا extreme programming.

استراتژی ما سرچ و اجرا شد و process ما قرار شد از روش های توسعه محصول agile استفاده کنیم در یک روش های غیر آبشاری بلکه به صورت تکراری چون روش آبشاری مهم ترین ویژگی اش sequential بودن است اما در روش های تکراری کاملا در مقابل روش های sequential قرار می گیرند.

ساختار شرکت ها معمولا به این شکل است

مدیرعامل در راس است و هیئت مدیره و مشاوره در بخش حسابداری و بازرگانی و فنی و فروش و... این تصویری است که از ساختار یک شرکت داریم.



اما در مورد استارتاپ هایی که از سازماندهی کسب و کار بر اساس کارکردشان، مثلاً حسابداری رسیدگی به حسابداری، نقش فنی برای توسعه محصول از فروش و وظیفه اش این است که محصولات را به فروش برساند و.... این یک تصویر functional است که برای استارتاپ ها جواب نمیدهد. برخلاف تصوری که از شرکتهای بزرگ داریم که مدیر عامل و موسسین و در اتاق های شان مدیریت می کنند در مورد استارتاپ ها اون کسایی که founder استارتاپ هستند به این ها نیاز داریم که حداقل ۲۰ درصد از زمان خودشان را خارج از شرکت صرف کنند برای اینکه بتوانند نیازهای مشتری را درک کنند.

نکته مهم این است که ویژگی های شغلی مثل بازاریابی فروش توی شرکت های بزرگ این مشاغل در کسب و کارهای نوپا متفاوت است یعنی کسی که دارد بازاریابی می‌کند و کسی که فروش را انجام می دهد در استارتاپ ها ممکن است یک نفر باشند.

آموزش کارآفرینی

مدتها فکر می کردیم برای کارآفرینی موضوعاتی مانند نوشتن BP چگونه برای سرمایه گذاران را ارائه کنیم اینکه چگونه پژوهشها را انجام بدهیم آموزش کارآفرینی کنیم ولی همه ی این موضوعات در اجرا خلاصه می‌شوند یعنی در فاز دوم هستند نمیگوییم به درد نمی خورند اما در فاز اول ما کارهای دیگری باید کنیم.

ما در بازی اول باید سرچ کنیم اینکه BP بنویسیم ارائه با سرمایه گذار داشته باشیم و تحقیق کنیم و محصول مان را توسعه بدهیم و نیازهای جدیدش را شناسایی می کنیم اینها همه به فاز اجرا برمیگردد اما برای راه اندازی استارتاپ آن چیزی که خیلی خوب باید بدانیم جستجو از فرآیند جستجو طی می شود و کشف می‌کنیم که مشتری های ما چه کسانی می توانند باشند می‌رویم و فرآیندهایی که باید BP بنویسیم.

در فاز سرچ باید راجع به موضوعات: فرضیه بیزنس مدل، اینکه چگونه مشتری را توسعه میدهیم کشف کنیم. درباره توسعه اجایل اینکه چگونه تیم تشکیل بدهیم موسسین چه وظایفی دارند بیزنس مدل و تیم و... را چگونه راه اندازی کنیم.

سوال: خروجی استارت آپ چه خواهد بود؟ آیا می تواند تبدیل به یک شرکت بشود؟ می‌تواند به صورت مستمری استودیو کند برای اینکه بتواند بیزنس مدل خود را ایجاد کند و گسترش بدهد؟ حتی ممکن است خیلی کند رشد کند و در نهایت حتی شکست بخورد. می تواند کلی پول و سرمایه از دست بدهد و ورشکست شود.

مطلوب این است که به شرکت تبدیل شویم اما یک خروجی کدام یک از این حالت ها است؟ همه این گزینه ها می توانند باشند. یعنی افراد کارشان را راه اندازی استارت آپ است یعنی یک ایده می دهند و دنبالش سرچ می کنند و به مرحله می رسانند و نمونه اولیه تولید می‌کنند و بعد استارتاپ را می‌فروشند به جایی که می خواهد کمپانی شود.

یک شرکت یا کمپانی یک سازمان کسب و کاری است که دنبال تولید محصول یا خدمت است یا محصول را تولید می‌کند یا سرویس را ایجاد می‌کند و دنبال این است که یک تعادل بین بازدهی و سود برقرار کند یعنی بتواند از کسب و کار خودش درآمد ایجاد کند که بازدهی و سود داشته باشد.

Start up چیست؟

شاید یک تصور این باشد که جایی که یک تعداد آدم باحال و کول دور هم جمع شده اند تا یک کسب و کاری را توسعه دهند ولی در واقعیت این طور نیست چون آدم های باحال می توانند دور هم جمع شوند ولی منجر به یک کسب و کار موفق نمی شود. یک استارتاپ یک سازمان موقت است طراحی می شود تا بتواند یک مدل بیزینسی که قابل تکرار و مقیاس پذیر باشد را جستجو کند.

Temporary

بر اساس آن چیزی که در دنیا دارد اتفاق می‌افتد این تغییر درست است که یک عده کارشان این است که یک استارتاپ را راه اندازی کنند و آن را به مرحله‌ای برسانند که بتوان به شرکت‌های بزرگ فروخت، آن را می‌فروشند و می‌روند یک استارتاپ دیگر را راه می‌اندازند و... یعنی استارتاپ قرار نیست مادام العمر استارتاپ باقی بماند قرار است تبدیل به شرکت شود.

طراحی می شود تا جستجو کند یعنی قرار است ما فعالیت جستجو(search)را به خوبی در مرحله استارتاپ انجام بدهیم تا بتوانیم یک بیزینس مدل تولید کنیم و اگر بیزینس مدل را ادامه بدهیم تبدیل به شرکت می شود این بیزنس باید دو ویژگی داشته باشد:

۱. Repeatable: جاهای مختلف بتوانیم کار را تکرار کنیم
۲. Scable: اگر مشتری های ۱۰۰ نفر است این مدل بیزینس جواب بدهد اگر شد ۱۰۰۰ تا جواب بدهد و....

ممکن است بگوییم ما اگر یک وبسایت راه انداختیم مثل اسنپ یک سری سرویس ارائه میدهد اگر تعداد مشتری ها زیاد شود لازم است که سرویس هایمان را ارتقا بدهیم ولی مدل بیزینس را تغییر نمی‌دهد تجهیزات و تکنیک ها را ممکن است عوض کنیم یا به روز رسانی کنیم.

Business Model

این canvas نوع های دیگری هم داریم که مفاهیم آن یکی ست اما با دسته بندی های یک مقدار متفاوت شاید ترتیب در این بیزینس مدل معنی نداشته باشد ولی اولین گزاره ای که ما به آن می پردازیم گزاره ی ارزشهاست.

گزاره ارزش Value Proposition

در این بخش به این مساله می پردازیم که محصول ما یا خدمت ما چه ارزشی دارد برای مشتری ایجاد می کند و چه مشکلی از مشتری را حل می کند و چه سودی به مشتری می رساند مشتریان ما چه کسانی هستند که این سود را میبرند و عملاً ارتباط ها با customer relationships و customer segments هست.

مسئله چیز دیگری ست و نیاز چیز دیگر؛ درباره ایده و از این مسایل نیست مشتری اهمیتی نمی دهد ما از چه تکنولوژی داریم استفاده می‌کنیم تا نیازش حل شود.

مثال: موضوعی مثل این که من یک ابزاری را درست کنم یا سرویسی را ارائه بدهم که مشتری می تواند کار پردازش لغت و تایپ و نامه نگاری و... را انجام بدهد این یک مسئله است و یک ابزاری داشته باشند که بتواند کارهای حسابرسی را انجام بدهد و عملاً دارم مسئله ای را حل می کنم اما موضوعاتی مثل ارتباطات و Entertainment که موضوعات کلی تری هستند را به عنوان نیاز میدانیم.

بخش بندی مشتریانcustomer segments

سعی می کنیم ببینیم چند دسته مشتری داریم در خیلی از بیزینس ها ما بیشتر از یک نوع مشتری داریم خیلی مهم است که متوجه شویم مشتری های ما چه کسانی هستند؟ چه تعداد احتمالاً محصولات ما را می خرند؟ کجا زندگی می کنند؟ یا شرایط اجتماعی شان چیست؟ درآمد اقتصادی شان چه میزان است؟ چه تیپ آدمهایی هستند؟ چگونه فکر می کنند؟به این اتفاقات در استارتاپ‌ها کمتر پرداخته می شود اما مهم است.

در حوزه فنی به مشکل جدی بر نمی خورند اما در بخش فروش مشکل دارند چون مشتری را از اول خوب نشناختند مشتری ما نیست که محصول ما را بخرد در واقع ما هستیم که برای مشتری وجود داریم و باید بتوانیم کاری کنیم که آن ها تمایل پیدا کنند که محصول یا سرویس ما را بخرند.

به عنوان مثال مطالعه کردند و گفتند که مشتری ها معمولاً بین ۲۰ تا ۳۰ سال هستند مثلاً یک جوان ۲۴ ساله آقا در شبکه های اجتماعی بسیار فعال است و تک درآمد است این ها را دسته بندی می کنیم مثل کاری که در آبجکت کلاس ها انجام می دهیم اول اشیا را می نویسیم و بعد در مفهومی به اسم کلاس دسته بندی می کنیم.

کانال های ارتباطیchanels

از چه روشی محصول یا سرویس را به مشتری ارائه می کنیم یعنی همان گزاره ارزشی که اشاره کردیم را با چه روشی به دست مشتری می رسانیم قبل از اینکه اینترنت و شبکه شکل بگیرد قبل از دهه ۹۰ فقط یک روش برای توزیع محصول وجود داشت که آن هم فیزیکی باید محصول را درب خانه مشتری یا یک جای مشخص می‌کردیم و می‌رساندیم، مشتری می آمد آن جا اما از دهه ۹۰ به بعد روش مختلفی مثل اینترنت موبایل و cloud وب بود

که باید بفهمیم که دقیقاً در مورد بیزینس مدل یا آن سرویسی که ما داریم کار می‌کنیم به چه روش است حتی می شود فیزیکی باشد و هم در نت ارائه بدهیم و هم به صورت اپلیکیشن موبایل باشد و.... یعنی ترکیبی باشد.

ارتباط با مشتری customer relationships

ارزش را به چه مشتری هایی و از چه کانالهایی میخواهیم برسانیم ولی ما ارتباط خودمان را با مشتری ها برای این که مشتری ها را جذب کنیم نگه داریم و ارتقا بدیم باید به سه سوال جواب دهیم:

چگونه مشتری بگیریم؟

چگونه مشتری ها را حفظ کنیم؟

چگونه مشتری ها را رشد بدهیم؟ یعنی بیشتر بشوند.

در مورد همه این بخش ها تنها یک مفروض داریم مثلاً در شرکت بیزینس مدل را تکمیل می کنیم فرضیه داریم اما واقعیت این است که وقتی بیرون می رویم و شروع به جستجو می کنیم نمونه های مشابه را می‌بینیم و رقبا را می‌بینیم و با مشتری ها شروع به حرف زدن می کنیم با مشتری تیپ ۱ و تیپ ۲ روش های مختلف ارزیابی محصول را که بررسی می‌کنیم سوالات مضمون دقیق تر می شوند یعنی پاسخ به سوالاتی که در همین بخش که پیشرفتیم را می توانیم دقیق‌تر پاسخ دهیم.

جریان بازدهیRevenue Streams

کسب و کارها چگونه ثروت ایجاد می کنند سوال را اینگونه جواب می دهیم که منابع درآمدی ما و روش درآمدی و مدل درآمدی چیست آیا حق اشتراک می گیریم؟ آیا می‌فروشیم؟ آیا پرداخت درون برنامه ای داریم؟ مثلا آیا وقتی کسی میخواهد برنامه آپ را نصب کند اول باید پول پرداخت کند؟ بعد نصب کند، ما می خواهیم یک ارزشی را به مشتری بدهیم و مشتری میخواهد پول بپردازد آن ارزش را چگونه میخواهیم؟ دریافت پول از مشتری مدل های مختلفی دارد که می توان ترکیبی هم در نظر گرفت.

به سرویس هایی که رایگان هستند مثل روزنامه الکترونیکی بدهیم که از راه تبلیغات کسب درآمد دارند، اینکه خبری را از یک نفر خاصی که با او در ارتباط هستند و صحبت می کنند و به او می گویند در مورد کارخانه و محصول شما توضیح می دهیم و ماهیت خبری می دهیم ولی عملاً کار را تبلیغ می کنیم و یه مبلغی دریافت می کنیم حال می توان خلاقیت های مختلفی را به کار ببریم تا بتوانیم درآمد ایجاد کنیم.

منابع کلیدیKey Resource

دارایی ها و منابعی که لازم داریم تا این کسب و کار را راه اندازی کنیم قرار به این سوال پاسخ بدهیم که اعتبار می خواهیم؟ تجهیزات فیزیکی و کارگاه و سوله می‌خواهیم؟ می خواهیم پتنت بخریم یا برای خودمان را حفظ کنیم؟

منابع انسانی: برنامه نویس و بازاریاب و...

هر آنچه را که می خواهیم لیست می کنیم تا این کسب و کار شکل بگیرد و بعد فکر می‌کنیم که هر کدام از این اعتبار های مان را چگونه جذب کنیم. ممکن است بگوییم ۵ نفر هستیم که هرکدام x مقدار داریم یا تسهیلات را از بانک می گیریم یا از سرمایه‌گذار ها می گیریم یا تسهیلات دولتی و...

می توانیم بگوییم ما ورژن اولیه را با پول خودمان بالا می آوریم اما وقتی که شروع به فروختن می کنیم از مشتری ها پول می‌گیریم بنابراین کسب و کار می چرخد و همیشه توصیه شده است. ساده ترین راه این است که سراغ سرمایه گذار برویم و تسهیلات بانکی بگیریم اما این بدترین تصمیم است سعی کنید آخرین گزینه جذب سرمایه گذار و تسهیلات ها باشد لزوماً خوب نیستند در اینجا به منابع مالی و فیزیکی فکر می‌کنیم و بعضی منابع Intellectual هستند مثل پتنت ها فکر و ایده افراد و منابع انسانی و نرم افزاری که باید بخریم تا راه بیندازیم.

همکار کلیدی Partners

اینکه چه افراد و سازمانهایی شریک کلیدی یا تامین کننده ما بحساب می آیند بعضی و حتما کسب و کار ما به یک بیزینس دیگر گره میخورد اصطلاحاتی مثل B۲B, B۲C, C۲B را که شنیدیم.

:Business to Customer

یعنی ما یک بیزینسی هستیم که مستقیماً محصول یا سرویس را به مشتری‌ها یا end user ارائه میدهیم.

:Business to Business

یک بیزنسی داریم که سرویس یا محصول را به یک بیزینس دیگر می فروشیم که آن بیزینس به یک تعداد مشتری سرویس می دهد بسته به نوع کاری که انجام می دهیم است. در اینجا پارتنر ها یا سازمان هایی که با آنها طرف قرار دادیم یا داریم محصول را به آن ها می فروشیم یا از آنها منابع اولیه تهیه می کنیم. مثلاً اگر ما یک استارتاپ راه انداختیم که نیاز به تجهیزات سخت افزاری دارد آیا سخت افزار را خودمان تولید می‌کنیم یا از یک شرکت واسط می‌خریم؟

لازم است سریعا متوجه شویم که تامین کننده ها چه کسانی هستند و چه امکاناتی دارند و به چه قیمتی می فروشند این که فقط برای خودمان فرضیه ببافیم و مدل کانواس رو پر کنیم کار اشتباهی است.

فعالیت های کلیدیKey Activities

برای اینکه بیزینس مدل ما کار کند چه کارهای مهمی رو باید انجام دهیم تا این محصول درست شود.

اینجا بحث‌های مدیریتی پروژه است یعنی اینکه من مثلاً سرمایه گذار بگیرم برنامه نویس استخدام کنیم بخشی از کارهای مدیریتی ممکن باشد اینکه توسعه محصول بدهیم بعد بازاریابی و انجام بدهیم وبسایت درست کنیم و ذکر کنیم که با فلان شرکت قرارداد ببندیم اینجا برنامه‌ریزی می‌کنیم نه به مفهوم اینکه چه کاری را چه موقع و توسط چه کسی شروع کنند بلکه میگوییم چه کارهایی را باید انجام بدهیم تا کسب و کار شروع شده را پیش ببریم سایت میخواهیم، اپ میخواهیم جزء key activity می گذاریم و قراردادها و تفاهم نامه ها با تامین کننده ها و....

مثلاً یک استارتاپ از مواد پسماند معدنی گلدان های شیک و مرتب می ساخت به سراغ کارخانه ها رفته بود صحبت کرده بود و قبول کردند که تا در کارگاه استارتاپ ببرند و پولی دریافت نکردند و رفتند، تامین کننده ماست. حتی ممکن است رقابت ایجاد شود و بگویند خودتان باید ماشین بگیرید ببرید یا حتی بفروشند.

هزینه ها costs

هزینه های اجرای طرح خیلی متنوع ملموس و ناملموس یکباره و در گردش است مثلاً حقوق کارکنان هر ماه می باشد ولی تجهیزات یکباره ملموس مثل خرید کامپیوتر و حقوق و این گونه چیزها اعتبار از دست دادن و گرفتن ناملموس درآمد ناملموس و ملموس روی این مسائل فکر می‌کنیم و تقسیم بندی می کنیم

هزینه ها و پول اولیه مشخص می کنیم؛ آیا هزینه ها با هم همخوانی دارد؟ آیا پولی که قرار است n نفر آدم استارتاپی را راه بیاندازند، کفاف هزینه ها را تا چه زمانی می دهد؟ آیا تا مدتی که نسخه اولیه را ببینیم ساپورت می کند؟ یا جذب سرمایه لازم است؟

فصل ۸

(UI) User Interface Design

سال‌های اخیر به ویژه با گسترش وب سایت و اپلیکیشن های موبایل دو اصطلاح ui و ux مقبولیت و موضوعیت بیشتری پیدا کردند و همچنین اهمیت طراحی یک اپلیکیشن و ظاهر اپلیکیشن یا وب سایت یا نرم افزاری که کاربر از آن استفاده میکند، یک طراحی مبتنی بر نیاز کاربر و تجربه کاربری باشه و مورد توجه قرار گرفته است. به جهت اینکه کاربرهای غیرحرفه‌ای گسترش یافتند و تقریبا افراد زیادی از این اپلیکیشن‌ها استفاده می‌کنند.

به طور کلی اهدافی که مدنظر این است که با مفهوم UI آشنا شوید و تعاملی که با کامپیوتر برقرار می شود و اصطلاح User Friendly بودن (کاربرپسند بودن)و نکاتی که باید در طراحی UI بهش توجه کنید.

نکاتی که باید در طراحی report ها یا گزارش‌هایی که ما به عنوان خروجی از سیستم استخراج می کنیم، نکته بسیار مهمی است که باید به آن توجه کرد و همچنین در خصوص ترندهایی که در حوزه UI وجود دارد مثل صفحات وب سایت Responsive باشند به مفهوم این که با توجه به اینکه اندازه device ها متفاوت است وب سایت واکنشی با توجه به سایز صفحه نمایش نشان دهد و به اصطلاح Responsive باشد.

ما وب سایت و اپلیکیشن را در یک موبایل باز می کنیم موبایل لمسی ۵ اینچی در یک تبلت ۱۰ اینچی در یک اسکرین مثلاً ۱۵ اینچ ۲۱ اینچ و بزرگتر بنابراین با توجه به اینکه اندازه های اسکرین های ما متفاوت هستند این قابلیت را دیده باشیم که در همه این دیوایس ها به شکل مناسبی به کاربر نمایش داده شود. تعامل یا UI واسط کاربری که کاربر با این کار میکنه فرآیندی رو ما باید در نظر بگیریم در طراحی نرم افزار در فاز طراحی بعد از اینکه اطلاعات را جمع آوری کردیم و فرآیندهای داخل سیستم را با استفاده از DFD و موجودیت ها را با استفاده از ER و مفاهیمی مثل data dictionary، مدل فیزیکی و منطقی را انجام دادیم یک بخش بسیار مهم برمیگردد به اینکه ما UI رو چطور طراحی کنیم. UI تمرکزش روی قابلیت استفاده از نرم افزار هست. بنابراین نکاتی مثل User Satisfaction اینکه کاربر باید وقتی با سیستم ما کار می کند راضی باشد و سیستم ما بتواند از عملکردهای اصلی سیستم به خوبی پشتیبانی کند و در عین حال بسیار کارآمد و بهره‌ور باشد. به این مراحلی که طی می‌کنیم تا این ظاهر یا صفحات مختلف نرم افزار و خروجی‌های این رو ورودی ها را طراحی کنیم UI می گوییم و چون مبتنی بر گرافیک هست GUI می گوییم.

apple جزء اولین شرکت‌های بود که بر اساس ایده‌هایی که در شرکت زیراکس مطرح شد توانست سیستم عامل را از حالت command که سیستم هایی مانند سیستم عامل داس اگر کار کرده باشید می بایست دستورالعمل آن را از پیش حفظ باشید فرمان به کامپیوتر بدهید و کامپیوتر اجرا کند این نوع واسطه کاربری و تعامل با کامپیوتر در مرحله گرافیکی پیاده سازی کرد یعنی موس را در سیستم آیکون نمایش داد و دستورالعمل آنها را به شکل منو یا مجموعه ای از گزینه‌ها در اختیار کاربر قرار داد. عملاً تعامل با کامپیوتر بسیار ساده تر شد.که مدت

های طولانی که کاربران کامپیوتر بخش اصلی تعامل را از طریق همین تعاملات گرافیکی انجام می دهند که این تعامل شامل ورودی اطلاعات نمایش خروجی و احتمالاً پرینت کردن خروجی بنابراین به صور مختلف باید به آن(UI)توجه کنیم. در واقع هدف این است که ما بتونیم تعامل کاربر به کامپیوتر و زیرساخت ها را فراهم کنیم این تعامل میتواند به شکل‌های مختلفی باشد؛ از طریق command و دستورالعمل ها، کلیک کردن و انتخاب، خروجی که روی صفحه نمایش نشان داده می شود یا روی کاغذ های چاپی ما می‌بینیم باشد. به طور کلی این تعامل و هر چیزی که به شکل ورودی خروجی اتفاق میافتد راتعامل با کامپیوتر می گوییم.

نکاتی که باید در طراحی واسط کاربری موفق شد توجه کرد:

طراحان UI باید عملکردهای اصلی این کسب و کار رو به خوبی بشناسند و آن‌ها را در ui بخش‌های مختلف نمایش دهند پس از اینکه کسی که قصد طراحی را دارد متوجه شود عملکرد سیستم چگونه است این مسئله بسیار موضوع مهمی ست.

این اطلاعات را بر اساس تحلیلی که در سیستم اتفاق افتاده و به دست می‌آید که از قابلیت های ترافیکی حداکثر استفاده شود تا بهره‌وری سیستم بالاتر باشه و عملکرد سیستم بهینه باشد بنابراین فرایند طراحی UI یک فرایند یادگیرنده است. سعی کنید در هنگام طراحی UI خود را به عنوان کاربری که احتمالا شناخت کافی از سیستم ندارد قرار بدهید و بارها تاکید شده که کاربر فقط EndUser نیست ممکن است مدیر سیستم باشد، یا مدیر ارشد سیستم باشد، یا ادمین سیستم باشد بنابراین از زاویه های مختلف و انواع مختلف کاربران شما باید سعی کنید UI را طراحی کنید و همانطور که گفته شد سعی کنید بسیار هدایت کننده باشد به این مفهوم که ممکن است کاربر از ابتدا شناختی از سیستم نداشته باشه ولی منطق سیستم را با دیدن واسط کاربری به سادگی بدست بیاورد، نه اینکه بسیار پیچیده باشد تا سعی کند متوجه شود منطق پشت سیستم چیست و چگونه می تواند کار کند. حتما سعی کنید از مدلها و پروتوتایپ ها یعنی نمونه های اولیه استفاده کنید همان مدل پروتوتایپی که برای بخش UI بسیار کاربردی ست؛ به این مفهوم که شما واسط کاربری اولیه را بر اساس شناختی که از سیستم به دست آوردید این را به کاربر نشان می‌دهد کاربر به شما بازخوردی می دهد که مثلا این قسمت به نظر ما اشکال دارد این کاربری که می گوییم نماینده یا تعدادی از کاربرانی هستند که قرار است در آینده از سیستم استفاده کنند و بازخورد می‌گیریم و بر اساس بازخورد مشتری و اصولی که شما باید رعایت کنید سعی کنید واسط کاربری رو اصلاح کنید.

باید تمرکز روی قابلیت استفاده از سیستم باشد یعنی اینقدر درگیر ظاهر و ظاهر زیبای سیستم نشده اید که از عمل کرد اصلی سیستم غافل شود سعی کنید در کارهای متفاوت را در سیستم به روش‌های متفاوت اجرا کنید و اجازه بدهید کاربر از چندین راه بتواند کار را انجام دهد.

مثال: این یک صفحه از یک نرم افزار وب سایتی که قرار است کار ثبت نام دانشجو انجام دهد سرویس‌هایی که دانشجویان به شکل گزینه‌هایی در قالب دکمه به کاربر نمایش داده می شود و کاربر می‌تواند بین این ها انتخاب کند کلیک کند و سعی کرده اید این کار با رنگ های مختلف انجام بدهید البته در خصوص این موارد باید حتما دقت کنید ببینید که چه کار می کنید یعنی این لزوما درست نیست که همه گزینه ها یا نمایش گزینه ها و رنگ های مختلف و فونت های مختلف انجام بدهیم یک تعادلی باید باشد که در نمونه ها بررسی کنیم.

اینکه حتماً محصول شما محصول نهایی شما باید به کاربر بازخورد دهد مثلا نگهداری کار طولانی را انجام می‌دهد از تکنیک‌های مثل Progress bar یا میله ی پیشرفت کار درصد پیشرفت کار استفاده کند اگر کار انجام شد پیام اینکه کار انجام شده و اگر کار با خطر مواجه شد نمایش خطا را ببیند. موضوع بازخورد بسیار مهم است. این تصویری که شما در فیلم می بینید وقتی با ابزار OS Studio ضبط می کنم علامت قرمز یک فیدبک است یعنی برنامه در حال ضبط است اگر این نباشه به همین سادگی من متوجه نمیشوم آیا واقعاً کار ضبط درست انجام می شود یا نه؟ این یعنی شما State ابزار و UI مون که عوض می شود را نمایش بدهید در واسط کاربری نمایش بدهیم که اتفاق می‌افتاد شما باید مستندش کنید و عملاً این اطلاعات قابل نمایش به کاربر هم باشد که شما هفتگانه که مرور می‌کنید، در طراحی کاربر شما باید با توجه به اینکه کسب و کار را خیلی خوب فهمید و همچنین عملکرد اصلی رو به یاد داشته باشید و به طور کلی توجه کنید؛ سعی کنید از امکانات گرافیکی حداکثر استفاده را بکنید، خود را در وضعیت های مختلف کاربر قرار دهید تا از تکنیک نمونه سازی و مدل سازی استفاده کنید تا بتوانید نظر کاربر را بگیرید و UI را به شکل یک مرحله تکاملی اصلاح و تکمیل کنید تمرکز را بر قابلیت استفاده از سیستم باشد. بازخورد به سیستم بدهید که کاربر را تشویق می‌کند هرجایی که لازم ا ست نظرش را راجع به سیستم بگوید و با آن تعامل داشته باشد. سعی کنید هر چیزی را مستندسازی کنید. Story board یا روایتی که کاربر از انجام کار داره و طراحی صفحات و نکاتی که در طراحی واسط کاربری کاربران مختلف استفاده کردن و به شما بازخورد دادند این ها را برای برنامه نویسی نهایی عملاً مستند کنید برای اینکه بتوانید UI طراحی کنید نسخه واحدی وجود ندارد. یک تعداد خطوط راهنما هست و باید رعایت شود و به این معنی نیست که هیچ کس حق ندارد از این خطوط راهنما یا از این چارچوبی که ترسیم می شود عبور کند شکستن ساختار به شرط اینکه یک سری نوآوری‌ها و ابتکارهایی باشد که در مجموع به نفع کاربر باشد میتواند اتفاق بیافتد و از این خطوط راهنما راحت شوید ولی دانستن این اصول و چارچوب طراحی واسط کاربری داشته باشید حتماً تمرکزتان روی اهداف سیستم یعنی عملکرد سیستم اصلی باشد و بنابراین شما باید اهداف طراحی سیستم باید برسید و بنابراین باید سعی کنید که فهم درستی از سیستم داشته باشید سعی کنید سیستم را روی کاری که قرار است انجام دهد به روش های مختلفی مثل دستورالعمل اینکه یک سری گزینه یکی از نمونه‌های مختلفی از آیکون ها استفاده می کنید به این روش توجه کنید اجازه بدهید که کاربر به سادگی خطا ها رو ببیند یعنی اشکالات داخل سیستم را ببیند و متوجه شود که اینجا جایی که مثلا داشت اطلاعات را وارد می‌کرد نادرست بوده و آنها را بتواند ویرایش کند یعنی به این مفهوم نباشدکه اگر کاربری در قسمت مشخص کد ملی خود را زد خطا میدهد

و این خطا نمیرود، ویرایش می کنیم که با کلیک روی x صفحه را ببندد با این کار کاربر به سادگی اشکالی که در سیستم ایجاد شده را برطرف می کند.

در UI خیلی خوب سعی کنید شما اطلاعات رو به کار ببرید یعی از لیبل و برچسب های مناسب استفاده کنید دکمه ها و آیکن ها همه مفاهیمی باشه که یعنی آیکون نمایش شکل باشند که قابل درک برای کاربر که ما هم همینطور و هر جا لازم از لیبل استفاده کنید یا برچسب ها و کلمات مناسب استفاده کنید که کاربر بداند الان در چه مرحله‌ای است و چه کار می کند همچنین از تصاویری که کاربر ها با آن آشنا هستند استفاده کنیم.

مثلا برای علامت save که سالهاست علامت آن شکل فلاپی است(از رده خارج شده ولی)با کلیک روی این علامت برنامه سیو می شود یا open یک پوشه ای که مثلاً باز شده این به مفهوم اینکه شما خیلی های دیگه جا افتاده و تصاویر مورد فهم برای همه کاربران است سعی کنید از command و لیستی از گزینه ها استفاده کنیم یعنی کاربر بتواند هم command رو در صورت نیاز دستورالعمل وارد کند هم یک مجموعه از گزینه ها را انتخاب کند کاربر بتواند به سادگی بین گزینه‌های مختلف جابجا شود و یک ساختار خوبی به شما برای کاربر آماده کرده باشید باید اصول را رعایت کنید تا produvtivity یا بهره وری کاربر بالا برود.

برای اینکه این اتفاق بیافتد لازم است شما کارها را دسته بندی کنید، کارهایی که هم خانواده هستند در یک منطقه قرار دهید و خیلی از منوها به عنوان مثال وقتی میگفت اینا این گزینه هم خانواده است وقتی علامت سه نقطه را جلوی یک دستوری می‌گذاریم که وقتی روی آن کلیک کنید پنجره باز می شود که اینجا باید بقیه این کار را دنبال کنید اگر تیک خورده باشد فعال است و....

کارهایی که توی سیستم قراره نمایش داده شود باید سازماندهی شوند و این organize به این مفهوم است که شما یک دسته بندی را در این عمل را انجام می دهید و سعی کنید کاربر راحت‌تر بتواند با سیستم تعامل کند اگر منوها، گزینه های زیادی وجود داشته باشد باید پشت این مرتب‌سازی ها این سوالات مطرح باشد آیا به ترتیب حروف الفبا مرتب شدند؟ آیا بر حسب موضوع و میزان اهمیت مسئله هستند؟

سعی کنید از shortcut ها یا اصطلاحاً میانبرهایی استفاده کنید تا کاربر بتواند کارش را از طریق میانبر انجام دهد. برخی کاربران علاقه دارن از موس استفاده کنند برخی از shortcutها بنابراین سلیقه هر دو را در نظر بگیرید یا اگر موس همراهش نبود کارش را بتواند با پیش‌بینی هایی که کردید کارش را انجام دهد هیچ اشکالی ندارد یک کار رو با سه شیوه‌ی مختلف شما می‌توانید پیش بینی کنید که کاربر انجام دهد.

مثال: برای پیگیری سفارش مشتری بوده منوی اصلی یک سری از گزینه ها را در بخش کاستومر میگذارد، یک سری در بخش order و یک سری در بخشproduct. پس قبل از طراحی در بخش پرداخت هزینه طراحی UI باید روی کاغذ فکر کنید به اینکه گزینه ها را چطور دسته بندی کنید و به عنوان منوها چی باشه برچسب‌ها چه باشد و...

مثلا اگر شما امکان پذیره اینکه از طریق صوتی فرمان رستوران بگیرید اطلاعات در مورد همه سیستم ها نیاز نیست ولی یادتان باشد که احتمالاً کار برای شما ممکن است یک تعداد اندک کاربران استثنایی وجود داشته باشند.

یعنی به این مفهوم که بیش از اندازه ضعیف باشه و بنابراین گزینه های ریز ممکن خیلی مناسب نباشد که این امکان را فراهم کرده باشید که از طریق صوتی با سیستم شما تعامل برقرار کند و شما به صورت صوتی پاسخ ها را در اختیار کاربران که احتمالاً نابینا و کم‌بینا است قرار بدهید.

به همین جهت سیستم برای گروهی طراحی می شود و این افراد به چقدر مهارت دارند که مخاطب این طرح اظهار داشت رسید موضوع مهمی سعی کنید سیستم شما انعطاف پذیر باشه در ارائه نمایش و هم در گرفتن ورودی از روش‌های مختلف بتوان این کار رو انجام بدهد حتماً برای کاربر راهنما آماده می کنید و این حال یکی لزوماً یک دفترچه راهنما نیست بلکه در بخش‌های مختلف شما ممکن است یکی نت یا راهنمای کوچک رو صرفاً برای پر کردن یک باکس در اختیار کاربر قرار بدهید.

حتما گزینه هایی را بگذارید که اگر کاربر دچار خطایی در سیستم شد مثلا اطلاعات را حذف کرد بتواند حداقل در چند مرحله برگرداند گزینه undo گزینه ای است که شما هر گاه استفاده کردید اشتباهی کنترل‌ها و مسیر دست‌نخورده را دیدید تصور کنید وجود نداشت چه اتفاقی می افتاد؟

تمام زحماتی که کشیدند از بین می‌رفت. اگر لازم است سعی کنید با استفاده از این لینک هایی که در صفحه ایجاد می کنید کاربر را به صفحات دیگری راهنمایی کنید یا مرتبط با موضوع هدایت کنید و جایگذاری این لینک ها، هایلایت کردن کلمات همه باید از یک منطقی برخوردار باشند.

یعنی نمی شود تعداد زیادی گزینه در صفحه قرار دهید و هایلایت کنید و همه را در یک الویت قرار دهید یا بخاطر زیبایی رنگ های مختلف انتخاب شده یا منطقی پشتش هست به عنوان نمونه این گزینه را میبینید که یک صفحه برای کاربر آماده شده که بخش های مختلف را راهنمایی می کند اگر به عنوان مثال شما روی رجیستر کنید و به یک صفحه ای که راهنمایی می کند.

راجع به اینکه چگونه ثبت نام کارت دانشجویی انجام شود، در راستای راهنما گفته می شود و به همین ترتیب در صفحه ای که به عنوان خروجی میبیند به کاربراجازه داده شود که چه کار می کند و اینکه آیا کارش نهایتا با موفقیت انجام شده یا نه.

شما تصور کنید من یک آپلود رو می خواهم انجام بدهم روی گزینه بارگذاری کلیک می کنم ولی هیچ پیامی به من داده نمی شود و من مطمئنم نیستم بارگذاری فایل درست انجام شد یا خیر همین نکات ریز بسیار مهم است. هر جا لازم بود و فکر کردید ابهامی هست متنی مناسبی را برای دکمه ها آیکون ها و تصاویر انتخاب کنید و سعی کنیدقابل فهم باشد، پیام ابهام نداشت و اگر قرار است پیام خطایی رو به کاربر بدهید به عنوان نمونه کد

ملی را وارد کرده نوعش را اگر براتون امکان پذیره مشخص کنید که آیا تعداد اعداد کم است؟ فارسی نوشته شده؟ فرمتش اشتباه است؟ و....

سعی کنید layout های جذاب استفاده کند یعنی کاربر وقتی در سیستم شما کار می کند نه تنها احساس خستگی نکند بلکه علاقه مند شود کار با سیستم را ادامه دهد. به جهت جذابیت که شما در سیستم پیش‌بینی کردید لازم است پیش بینی کنید که کاربر به خصوص در ابزارهای هوشمند که امروزه هستند مثل تبلت و موبایل آنها از کیبورد مجازی استفاده کنید یا اگر سیستم بانکی طراحی کردید به جهت امنیت از کیبوردهای درون سیستمی استفاده شود که ارزیابی keylogger نتواند حرکت کیبورد را یا دکمه هایی که کاربر آن وارد می‌کند را تشخیص دهد از عناوین و پیام ها و دستورالعمل های مناسب در وقت مناسب استفاده کنید سعی کنید یک سازگاری را رعایت کنید.

در همه موارد واژه‌شناسی استفاده کردید مثلاً برای این واژه یا علامت رو برای اخطار در صفحات دیگر استفاده کنیم بیش از اندازه تنوع و مثلث قرمز با علامت تعجب زرد دایره و مثلث متوازی الاضلاع کنم یا در مورد کلمات هم نادرست بودن این از علائمی که تقریباً برای شما منطق دارد که چرا در اینجا این علامت مثلث با علامت تعجب استفاده کردم چرا علامت دایره و.... منطق و کاربر به سادگی درک می‌کند پیش بینی کنیم. احتمالا خیلی از وبسایت هایی که مراجعه کردید فرمی را خواستید پر کنید که با علامت تب بین شان جابجا می شود این را پیش بینی کنید حتماً و این طرح باید به ترتیب باشد.

استاندارد بین المللی که تقریباً همه جا چراغ راهنما قرمز به مفهوم اینکه ایست، زرد به مفهومی که احتیاط کنید و سبز مفهوم اینکه مجاز به حرکت هستید، است. از همین علائم سه گانه استاندارد بین المللی میتوانید در خیلی از مکان ها برای ورود اطلاعات و ادامه مسیر UI استفاده کنیم.

همانطور که گفتم سعی کنید از جایگزین کیبورد برای موساستفاده کنید.

از دستورالعمل‌های متداول و commandها اگر امکان پذیرهست استفاده کنید

واسط کاربری را نیازی نیست همه اطلاعات را در یک محیط فراهم کنید، در صفحات مختلف می توانید این کار را انجام دهید و سعی کنید این نمایش پنجره خیلی جذاب و جالب باشد که کاربر احساس خوبی داشته باشد در کار با سیستم شما از کلمات فنی خیلی پیچیده و اینکه لزوماً کاربر متوجه نشود استفاده نکنید بگویید نرم افزار حسابداری را به اصطلاح حسابداری و اداری موجود استفاده می‌کنیم اشکالی ندارد ولی در راهنمایی این خطا را دارید به شما اعلام می‌کنید از اصطلاحات عجیب غریب یا پیچیده استفاده نکنید.

صفحه ورود به سیستم که اطلاعات کافی را میخواهد بدهیم، سازماندهی مناسب اتفاق افتاده باشد. دسته بندی مناسبی باشد تنوع رنگی تا حدی خوب باشد ولی از حدی بالاتر خسته کننده می شود فقط تا جایی که ممکنه

کاربر مدت‌های طولانی و سیستم شما کار کند برنامه رنگ بندی ها و کنتراست بین رنگ متن و رنگ پس زمینه وجود دارد نکات را رعایت می‌کنیم.

هر کدام از این استانداردها را انتخاب می‌کنیم مثل button radio یا چک باکس به مفهوم انتخاب کنیم یعنی گزینه مختلف کاربرد فقط و فقط مجاز است یک گزینه را انتخاب کنند ولی وقتی چک باکس انتخاب می شود نمیتواند چندین گزینه را انتخاب کند.

مثلا از * برای موارد ضروری در فر ها استفاده شود ولی اگر برای همه ی قسمت ها ستاره گذاشتیم.

مثلا برای استانها گزینه بگذاریم چون گزینه ها محدود است.

گزارش باید عنوان مشخص باشد/ تاریخ چاپ/توسط چه کسی ودر چه بازه ی زمانی/عنوان مختلف برای بخش های مختلف جدول داشته باشد.

سایت را به صورت ساختاری یا فرایند محور کار ارائه دهیم.

Structural: ساختاری که در دانشگاه هست را روی وبسایت بگذاریم این برای کاربران غیر دانشگاهی یا کسانی که اطلاعات کافی راجع به دانشگاه ندارند کار را برای پیدا کردن اطلاعات مشکل می کند.

فرایند محور: کارها یا خدماتی که کاربران بیشتر با آن سر و کار دارند را صفحه اول نمایش بدهیم.

منظور از sequence check همین است که بتوانیم با tab جابجا شویم و توالی انجام کار را بررسی کنیم. Existance check یعنی دیتا هایی که لازم به وارد شدن است را حتما چک کنیم؛ نوع داده ها درست وارد می شوند؟ در بازه مناسبی وارد می شود؟ اعتبار سنجی و قوانینی که باید رعایت بشه در UI را توجه کنیم و اگر لازم است کاربر را درست راهنمایی کنیم

حجم ورود اطلاعت رو کم کنید مثل استانها که اشاره کردیم.

فصل ۹

Entity: موجودیت می تواند هر آن چیزی باشد که به ذهن ما خطور می کند.

دو حالت مختلف داریم :

۱. فیزیکی
۲. غیر فیزیکی(مجازی)

به عنوان مثال موجودیت دانشجو، ولی درس وجود خارجی ندارد؛ یعنی موجودیتی به نام فرد داریم ولی ممکن است موجودیتی به نام درس (را به طور واقعی) نداشته باشیم.

Table , File: مجموعه ای از دیتاهایی که با یکدیگر ارتباط دارند و ما ذخیره می کنیم. همانند اینکه فردی با هویت دانشجو یکسری اطلاعات دارد و این رکورد هایی که وجود دارد با یکدیگر ارتباط دارند.

...	شماره دانشجویی	نام دانشجو
	۹۷۰۱۰۲۰	رضا محمدی
	۹۶۰۲۰۳۰	علی رضایی

Field: یعنی به هر کدام از موارد ما یک فیلد میدهیم مثلاً فیلد ۹۶۰۲۰۳۰ که یک حالت خاص است در واقع توضیحی به ما می‌دهد.

Record: به هر کدام از مجموعه های یک رکورد می گوییم و به صورت مجموعه نشان می دهیم در واقع یک نمونه را به ما نشان می دهند.

{< "..." , "۹۷۰۱۰۲۰" , "رضا محمدی" >}

Key filed

کلید :یک سری ویژگی های خاص دارد چند حالت مختلف دارد. کد ملی

c	b	a

Primary key کلید اصلی: یک یا چند فیلد است مثلاً در جدول یک ویژگی خاص مثل کدملی داریم که یکتا می باشد و برای افراد مختلف می تواند باعث شناسایی آنها شود عملاً با داشتن کد ملی افراد می توانیم به آنها دسترسی داشته باشیم. در جدول به فراخور جایگاه های مختلف ممکن است این کلید متفاوت باشد مثلا در دانشگاه شماره دانشجویی در بعضی از موارد ترکیب چند مورد می باشد.

دانشجو

...	شماره دانشجویی	کد ملی

Candidate key کلید کاندید: می تواند کلید اصلی باشد مثلاً در جدولی در دانشگاه کد ملی و شماره دانشجویی را داریم هر دو مورد یکتا هستند بر اساس حجم و در نظر گرفتن معیار ها کلید را انتخاب می کنیم که کدام یک از این ها کلید اصلی باشد مثلاً پردازش و جستجو روی شماره دانشجویی راحت تر می باشد پس کلید اصلی می گذاریم .

درس

...	کد درس	اسم درس
	۱۲	تحلیل

Foreign key کلید خارجی: وظیفه برقرار کردن ارتباط بین جداول مختلف را دارد مثلاً ۲ جدول داریم وقتی بخواهیم بین جدول دانشجو و ترس ارتباط ایجاد کنیم شماره های دانشجویی را که کلید هسته اصلی هست را همراه با کد درس کنار یکدیگر قرار می دهیم.

...	۱۲	۹۶۰۲۰۳۰

Secondary key کلید ثانویه: برای دسترسی و بر برگرداندن یکسری رکورد می باشد.

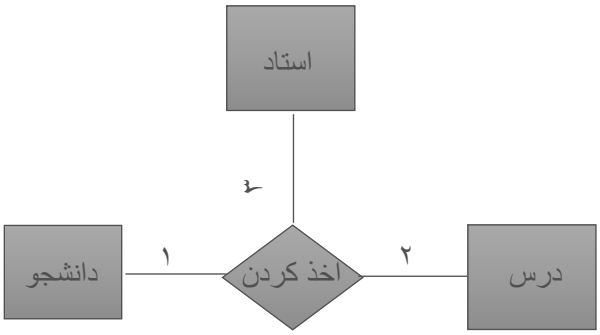
قاعده جامعیت اجرائیRefrential integrity

این قانون بر کلید خارجی نظارت می کند یعنی اگر ما در سیستم یک کلید خارجی اطلاعاتش را وارد می‌کنیم حتما در جدول رفرنس داده شده وجود داشته باشد به عنوان مثال ۹۶۰ ۲۰۳۰ درس ۱۲ را اخذ کرده این شماره دانشجویی عملاً کلید خارجی بوده نکته ای که وجود دارد این است که حتماً کد درس ۱۲ جزء این موارد داخل ستون جدول باشد.

ER

موجودیت رابطه

مثال

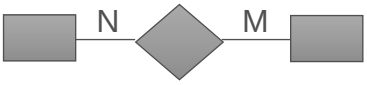


رابطه می تواند حالت های مختلفی داشته باشد مثلاً اگر روابط یک و دو باشد رابطه دوگانه داریم یا می‌توانیم رابطه بازگشتی داشته باشیم.

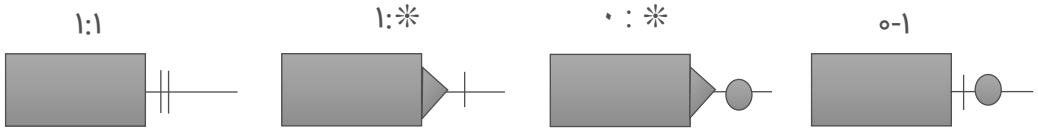
رابطه یک به یک(۱:۱): فقط یکی از این مجموعه می‌تواند با مداحی یکی از موارد مجموعه دیگر در ارتباط باشد مثلاً مدیر و دفتر مدیر رهبری را مدیریت می کند میدانیم که یک اداره یک مدیر کل دارد فکر کنید هر مدیر فقط می تواند مدیره یک اداره باشد حال یک اداره چند مدیر می تواند داشته باشد؟ یکی پس رابطه یک به یک است.

رابطه یک به چند (1:M): رابطه مادر و فرزند یک مادر می‌تواند چند فرزند داشته باشد ولی فرزند نمی تواند چند مادر داشته باشد.

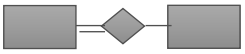
رابطه چند به چند (M:N): مثلاً یک دانشجو می‌تواند چند درصد داشته باشد و یک درس می‌تواند توسط چند دانشجو اخذ شود.



Cardinality



مفهومی به نام اجباری یا اختیاری داریم. یعنی یک موجودیت می تواند در یک رابطه حتماً باشد اختیاری است می‌تواند در رابطه نباشد اجباری را با دو خط نشان میدهیم و اختیاری را با یک خط نشان می‌دهیم.



Attribute: ماهم موجودیتی داشته باشیم یکسری ویژگی ها دارد. ویژگی را با این علامت @ نشان می‌دهیم و یا اگر صفت کلیدی باشد زیر آن خط می کشیم.