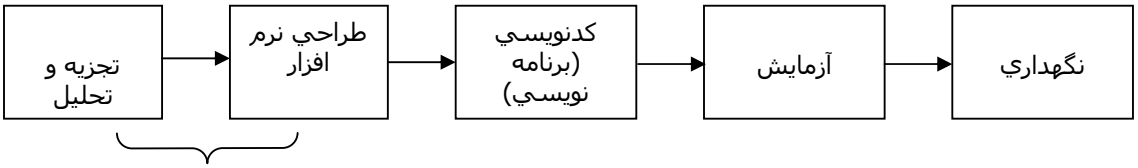
۱. روش ترتيب خطي (The Linear sequential Method)

اين روش ساده ترين و كار آمدترين روش براي توليد نرم افزار نسبتا محدود مي باشد كه در آن تحليل نيازمندي ها و تعريف صورت مسئله به سادگي انجام مي شود .



اين روش جهت مهندسي نرم افزار كه بعضي مواقع چرخه حيات كلاسيكي نيز ناميده مي شود .روش ترتيب خطي يك رويكرد تر تيبي سيستماتيك در مورد طراحي وساخت نرم افزار دارد در سطح سيستم آغاز ميگردد و در راستاي تجزيه و تحليل ، طراحي ، برنامه نويسي ، آزمايش و نگهداري ادامه مي يابد . اين مدل كه در پيروي از چرخه مهندسي متعارف مدل سازي مي شود در بر گيرنده فعاليت هاي زير است:

- **مهندسي سيستم /اطلاعات و مدلسازي :**

به علت اين كه نرم افزار پيوسته بخشي از يك سيستم بزرگتر (يا تجاري اداري) است با تعيين نيازمندي ها جهت تمام عناصر سيستم آغاز ميگردد و سپس زير مجموعه اي از اين نيازمندي ها به نرم افزار اختصاص مي يابد . اين ديد سيستم هنگامي كه نرم افزار بايد با ساير عناصر نظير سخت افزار ، افراد و پايگاه هاي داده ها در تعامل باشد حايز اهميت است . مهندسي سيستم و تجزيه و تحليل در برگيرنده جمع آوري نيازمندي ها در سطح سيستم توام با مقدار كمي تجزيه و تحليل و طراحي سطح بالا مي باشد. مهندسي اطلاعات در برگيرنده جمع آوري نيازمندي ها در سطح تجاري اداري راهبردي و در سطح زمينه تجاري اداري مي باشدد .

- **تجزيه وتحليل نيازمندي ها :**

فرآيند جمع آوري نيازمندي ها شدت مي يابد و به ويژه برروي نرم افزار متمركز ميگردد . براي درك ماهيت برنامه ها كه بايد ايجاد گردند مهندس نرم افزار «تحليلگر» بايد قلمرو اطلاعات جهت نرم افزار ووظيفه مورد نياز ، رفتار ، عملكرد و تعامل را درك نمايند . نيازمندي ها جهت سيستم و نرم افزار هر دو مستند سازي ميشوند و توسط مشتري مورد بررسي قرار ميگيرند .

- **طراحي :**

طراحي نرم افزار يك فرايند چند گامي است كه برروي چهار صفت خاصه متمايز يك برنامه به نامهاي ساختمان داده ، معماري نرم افزار ، نمايش هاي رابط و جزئيات رويه اي (الگوريتمي) تمركز مي يابد . فرآيند طراحي نيازمندي ها را به نمايش نر افزار كه مي تواند قبل از توليد كد جهت كيفيت مورد ارزيابي قرار مي گيرد تبديل مي نمايد . طراحي همانند نيازمندي ها مستند سازي مي شود و به صورت بخشي از پيكر بندي نرم افزار در مي آيد .

- **توليد كد :**

طراحي بايد به شكلي قابل خواندن جهت ماشين تبديل گردد . گام توليد كد اين وظيفه را انجام ميدهد ، چنانچه طراحي به روش تفضيلي انجام گيرد توليد كد مي تواند به صورت مكانيكي انجام گيرد .

- **آزمايش :**

پس از توليد كد آزمايش برنامه آغاز ميگردد . اين فرايند براي حصول اطمينان در مورد عدم وجود خطاي منطقي در نرم افزار و اطمينان از اينكه تمام دستور ها آزمايش شده اند و برروي هدايت آزمايش ها جهت كشف خطاها ي دستور ي تمركز مي يابد تا اطمينان حاصل شود كه ورودي تعريف شده نتايج واقعي كه با نتايج موردد نياز تطابق دارند را توليد مي نمايد .

- **نگهداري :**

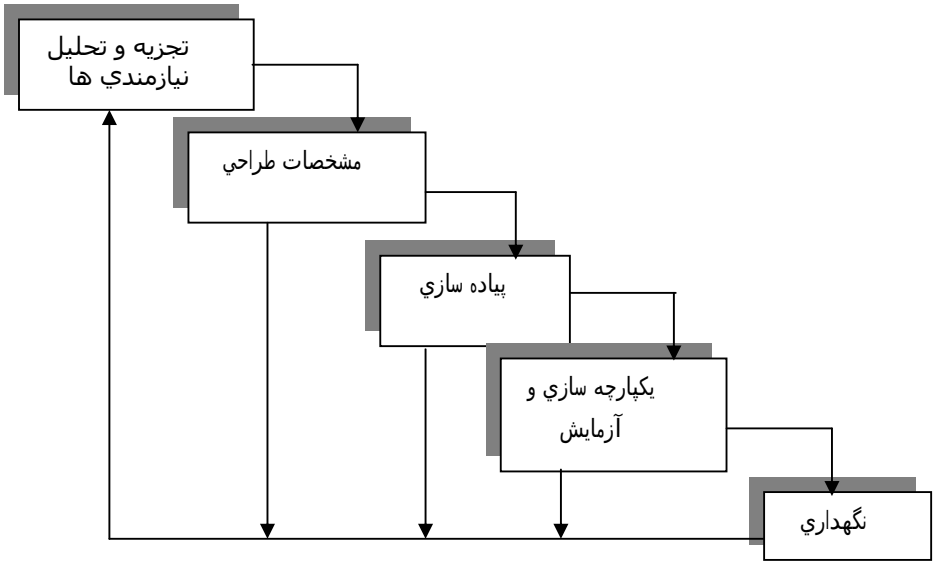
بدون ترديد نرم افزار پس از تحويل به مشتري ، تغييراتي را طي مي كند . اين تغيير ممكن است به علت برخورد مشتري با خطاهاي احتمالي باشد و يا به علت تغيير در محيط اطراف نرم افزار باشد (تغيير سيستم عامل يا دستگاه هاي جانبي) و يا بهينه سازي عملكرد نرم افزار .

هنگام بهره گيري از از اين روش مسائل زير مشهودند :

- پروژه هاي واقعي كمتر از اين روش استفاده ميكنند .
- غالبا براي مشتري سخت است تمام نياز مندیها را صريحا بيان نمايد .
- مشتري بايد صبور باشد .
- ايجاد كنندگان غالبا تاخير غير ضرور را سبب مي شوند .

۲. روش آبشاري(Water fall Method):

اين روش براي طراحي وساخت نرم افزار به گونه اي كه در شكل ميبينيد ميباشد . اين روش فرايند طراحي وساخت نرم افزار را به گونه اي تجلي ميبخشد كه همانند كارخانه اي كه مواد خام از يك طرف آن وارد و محصول از طرف ديگر خارج ميشود ، ميباشد . اهميت نسبي مراحل مختلف بر حسب سطح كاري وزمان متفاوت است . مراحل اوليه در مدل كار مهندسي بيشتري را مي طلبد ولي زمان كمتري مصرف آنها ميشود ، در حاليكه در مراحل بعدي فشردگي كار كمتر است ، ليكن بيشتر طول ميكشد . مراحل تجزيه وتحليل نيازمندي ها و مشخصات طراحي نياز به تعامل قابل ملاحظه اي ما بين مهندسين سيستم و كاربران نهايي دارد .



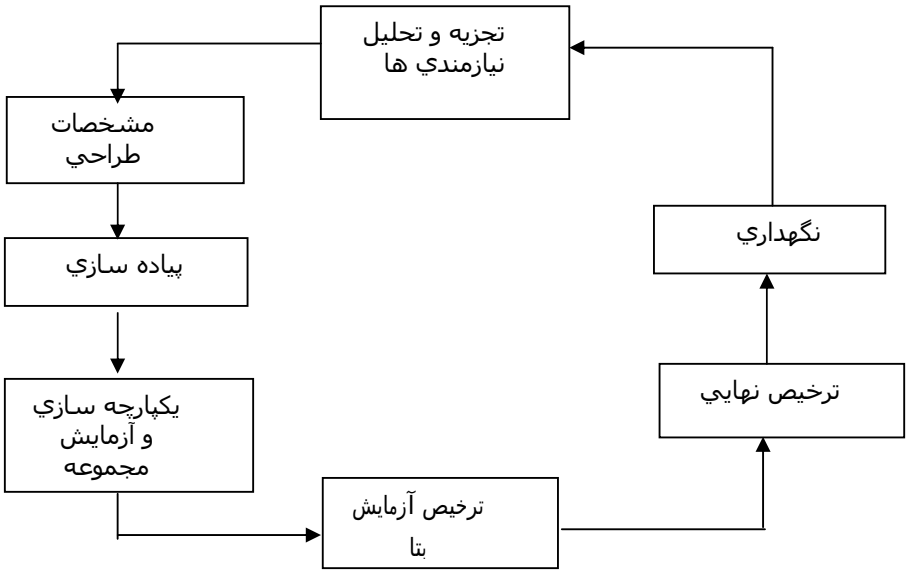
مشكلات روش آبشاري :

- **بازگشت به عقب**
- **نامشخص بودن نيازهاي مشتري**
- **انتظار طولاني مشتري**

۳. روش چرخه اي (Feedback Loops Method)

اين روش به گونه اي است كه در شكل نمايش داده شده است ، هرند در روش چرخه اي به طور كلي مراحل به يك روش خطي گام به گام اجرا ميگردند ، معمولا در مراحل تجزيه وتحليل نيازمندي ها و مشخصات طراحي حلقه هاي باز خور وجود دارند .

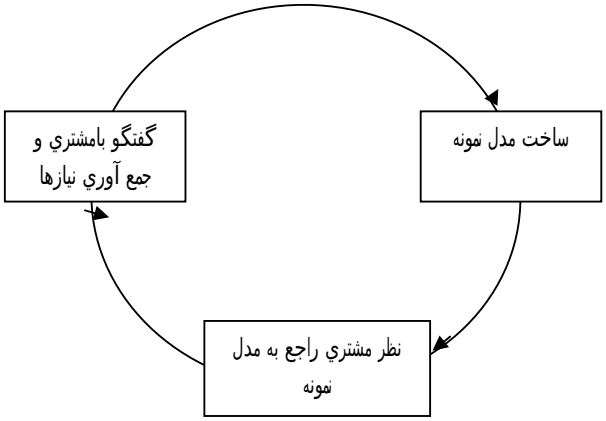
اين بويژه در طراحي و ساخت نر افزار سفارشي كه در نمونه ها طراحي كه در طول مرحله مشخصات طراحي ساخته ميشند ممكن است نيازمندب=ي هاي كاربر نهايي را آشكار سازند ، صدق ميكند .



٤. روش نمونه سازي (Prototyping method)

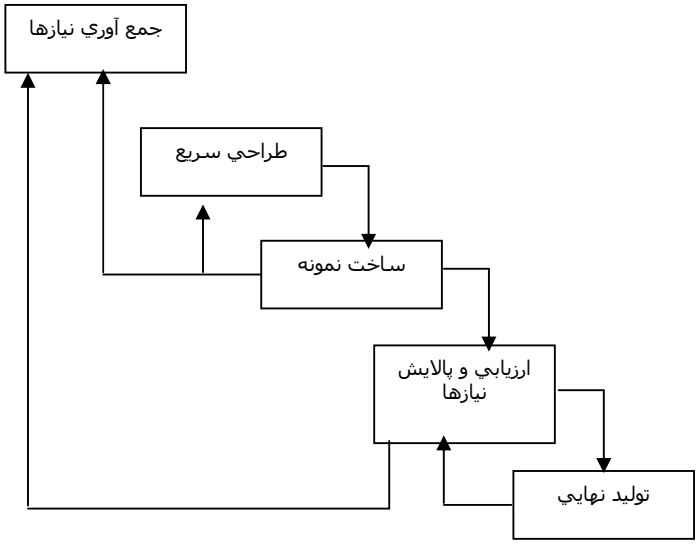
در اين روش توليد كننده يا همان طراح نرم افزار قادر است روشي از نرم افزاري را كه مي خواهد توليد كند هر چند به طور مختصر و مفيد و به شكل پيش فرض به صور مختلف حتي روي كاغذ به كاربر نشان دهد بطوري كه كاربر ارتباط ميان خود و كامپيوتر را احساس كرده و متوجه عملكرد نرم افزار شود يا اينكه قسمتي از برنامه را نوشته و به كاربر نشان دهد و بعد از اينكار به نوشتن برنامه اصلي و جزئيات بپردازد .

برنامه نوشته شده بايد در حدي باشد كه حداقل تصور ممكن را از برنامه در ذهن كاربر زنده كند و طراح نرم افزار توضيحات لازم را براي تشريح عملكرد كامل برنامه در آتيه به كابران ارايه كند . البته طراح بايد به كاربر ومصرف كننده بفهماند اين برنامه فقط قسمت كوچكي از كارهايي را كه در آينده و پس از تكميل پروژه انجام خواهد داد را در حال حاضر انجام ميدهد تا كاربر دچار اشتباه نشود و فكر نكند كه برنامه نهايي نيز در اين حد كارايي خواهدداشت .



« شكل ارتباط طراح با مشتري »

همانطور كه در شكل پايين مشاهده ميكنيد طراح در ابتدا بامشتري به گفتگو ميپردازد و نظرات آنرا در رابطه با نيازهايي كه توقع دارد نرم افزار مورد نظرش را در آينده برآورده سازد جويا ميشود و به جمع آوري آنها ميپردازد ، سپس به ايجاد يك برنامه پيش فرض و مدل پرداخته و آنرا در اختيار مشتري قرار ميدهد و نظر آنرا نسبت به مدل ساخته شده جويا ميشود و در صورت نياز به تكميل مدل نمونه يا ساخت نمونه اي ديگر ميپردازد .



مزايا :

- **امكان تغيير و جمع آوري نيازها**
- **در طول توليد محصول ارتباط مشتري با طراح و توليد كننده همواره برقرار است .**

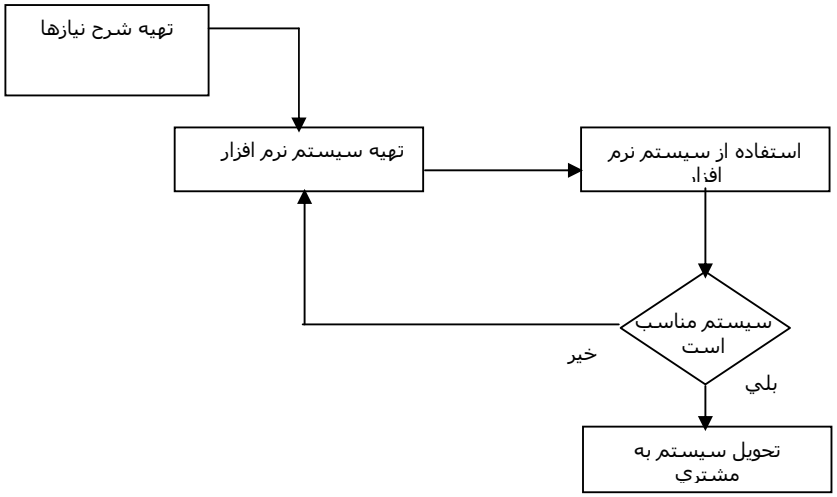
معايب :

- **چون نمونه اي از نر افزار در اختيار مشتري قرار ميگيرد و مشتري در ابتدا نميتواند نرم افزار كامل را ببيند ممكن است تصوير غلطي از نرم افزار نهايي پيدا كند .**
- **نمونه ايجاد شده بدون در نظر گرفتن مسائل كيفي و غيره ميباشد و درك ايجاد مجدد نرم افزار براي مشتري مشكل است . چون او از مسائل كيفي ومهندسي هيچ اطلاعي ندارد و معمولا تغييرات پي در پي برنامه براي مشتري خوشايند نيست .**
- **مشكل سوم به خود طراح يا توليد كننده نرم افزار مربوط ميشود . او ممكن است براي دستيابي سريع تر به مدل نمونه به مسائل مهمي توجه نكند . مثلا سيستم عامل مناسبي انتخاب نكرده و يا زبان برنامه نويسي نا مناسبي را براي نوشتن مدل انتخاب نمايد ، فقط به دليل سهولت كار ويا اينكه صرفا اين برنامه يك نمونه است و اين باعث وجود مشكلاتي در آينده مي شود .**

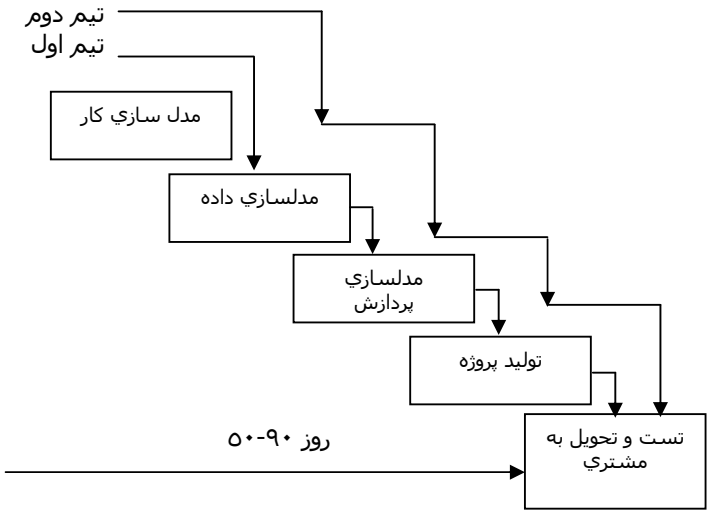
۵. روش اكتشافي(Exploratory Method)

در اين روش ابتدا يك سيستم ساده توسعه يافته به مشتري تحويل ميگردد و اين سيستم در مراحل مختلف اجرائي مرتبا با نظرات مشتري تكامل مي يابد تا به يك سيستم مناسب تبديل گردد. اين روش براي حالتي مفيد است كه نتوان به شرح احتياجات كامل دست يافت مانند سيستمهائي كه براي كاربردهاي هوش مصنوعي لازمند.يك اختلاف اساسي اين روش و روشهاي مبني بر شرح احتياجات كامل در اين است كه تضمين اعتبار و تست اين روشها بسيار مشكل است. اين روشها براي حالاتي كه سيستم مرتبط با هوش مصنوعي مورد نيازاند به سه دليل زير بسيار مفيد خواهند بود:

۱. در سيستمهاي مرتبط با هوش مصنوعي شرح احتياجات اوليه سيستم بصورت كاملي مشخص نيست و انجام هزينه هاي گزاف تهيه اسنادي كه دقيقا نمي تواند احتياجات را مشخص سازند عملا ضرورت ندارد.
۲. در اين سيستمها , تعمير و نگهداري بسيار ساده تر از سيستمهائي است كه شرح احتياجات آن دقيق و تثبيت گرديده اند.
۳. تعداد محدودي افراد ماهر و كاري براي توسعه سيستم كافي خواهند بود.

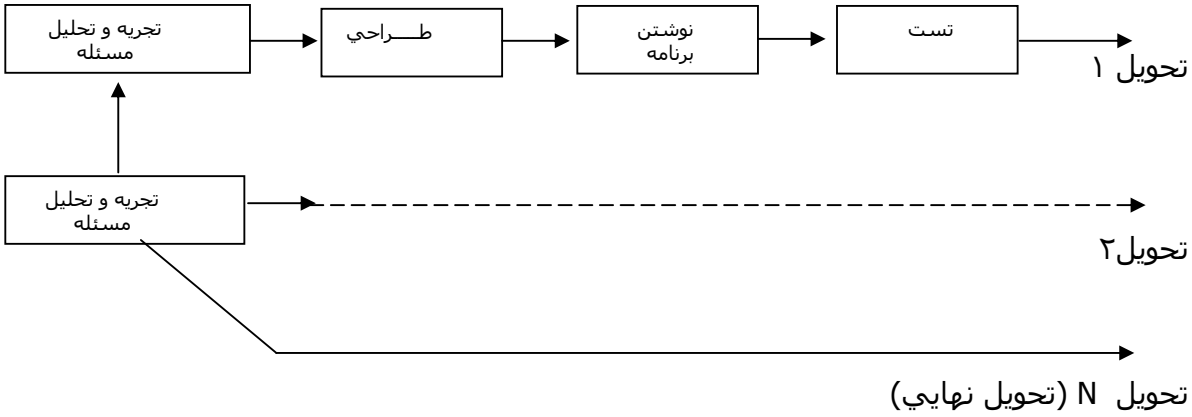


٦. روش تقسيم كار يا توليد سريع (Rapid Aoolication Development)
همانطوريكه از نام اين روش مشخص است نرم افزار به قسمت هاي مختلف تقسيم شده و همواره سعي مي شود كه نرم افزار مورد نظر سريعتر توليد شود . معمولا افرادي از يك تيم روي قسمتهتاي تفكيك شده كار ميكنند و در پايان نتيجه كار را با يكديگر تركيب مي نمايند تا محصول نهايي شكل گيرد ، البته نكته قابل توجه اين است كه نرم افزار مورد نظر بايد خاصيت تفكيك پذيري داشته باشد تا بتوان اين مدل را پياده سازي كرد . اين روش معمولا براي پروژه هاي نسبتا بزرگ كاربرد دارد و افراد يك تيم بايد نهايت همكاري و دقت را داشته باشند تا بتوانند در موعد مقرر توليد نهايي را تحويل مشتري نمايند چون اگر يك قسمت از اين پروژه انجام نشده باشد تحويل پروژه ممكن نيست ؛ بنابراين از صحبت هاي گفته شده ميتوان دريافت كه مديريت كردن اين پروژه ها كار آساني نمي باشد چون با دو مشكل افراد و زمان محدود رو به رو هستيم اما بزرگترين مزيت آن هم همان كوتاه بودن دوره توليد و تحويل به مي باشد .



۷. روش افزايشي (Incremental Method)

در اين روش پروژه به تدريج كامل مي شود يعني هر مرحله اي كه ميگذرد پروژه كامل تر شده و در نسخه هاي بعدي اين تكامل ادامه مي يابد تا به هدف اصلي برسيم ؛ بنابراين مي بينيد كه مثل روش نمونه سازي ، نمونه هاي ايجاد شده بلا استفاده نبوده و دور ريخته نمي شود . اين روش معمولا مواقعي استفاده مي شود كه امكانات كافي در اختيار نداشته باشيم و مجبور باشيم از حداقل امكانات در مراحل مختلف براي ايجاد نتيجه مطلوب استفاده كنيم .

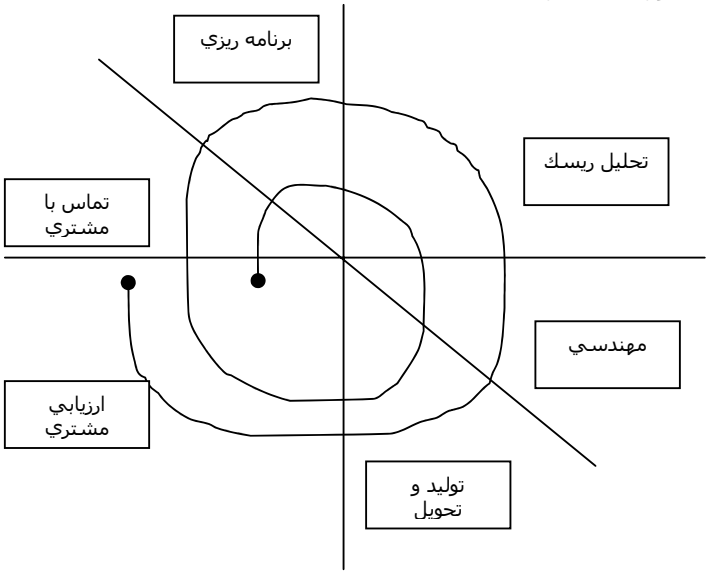


۸. روش حلزوني (Spiral Method)

در اين روش اعمالي متفاوت با آن جه كه تا بحال ديديم انجام مي شود . يعني اين روش ، روشي است كه از تركيب روش هاي خطي و نمونه سازي استفاده كرده ، بدين شكل كه روش خطي است اما منطق استفاده شده همان منطقي است كه در روش نمونه سازي استفاده ميشد . در اين روش براي توليد نرم افزار مراحلي وجود دارد كه عبارنتد از :

۱. تماس با مشتري و جمع آوري اطلاعات مورد نياز .
۲. برنامه ريزي و تشكيل تيم هاي كاري و كنترل دقيق امور جاري.
۳. پيش بيني خطرات و اتفاقات احتمالي كه ممكن است رخ دهد كه به آن تحليل ريسك نيز گفته ميشود .
٤. انجام عمليات مهندسي .
٥. توليد پروژه كه شامل كد نويسي ، تست ، عمليات كنترل كيفيت و تحويل به مشتري است .
٦. نظر خواهي از مشتري يا همان ارزشيابي مشتري .

اين روش ، روشي مطمئن و قابل اعتماد است زيرا در هر مرحله يا در هر دور تحليل ريسك ، كيفيت نرم افزار به مراتب بالاتررفته و مدير پروژه بر امور جاري كنترل دقيق دارد و همه مسائل و هزينه ها با توافق طرفين صورت ميگيرد .



۹. روش جمع آوري اجزا (Component Assembly Method)

در اين روش كه به روش مونتاژ مؤلفه ها نيز مشهور است اجزاي نرم افزار به شكل مؤلفه هايي در نظر گرفته ميشود كه هريك از قابليت ساخت ، تست و استفاده مناسبي برخوردار است . در اين روش ميتوانيد هرروشي را استفاده نماييد .

۱۰. روش توسعه موازي (The Concurrent Development Method)

در اين روش فعاليت هايي كه جهت تكميل پروژه و رسيدن به نتيجه نهايي انجام ميگيرد به شكل موازي ميباشد ؛ بنابراين باتوجه به موازي بودن فعاليت ها الزاما زمان انجام آنها يكي خواهد بود . پس زمان تحويل يا بهتر است بگوئيم دوره توليد نرم افزار از لحاظ زماني كاهش مي يابد ، اما ممكن است پارامتر هاي ديگر با كاهش مواجه نشوند و حتي شايد افزايش يابند .

۱۱. روش صوري (Formal Methods)

اين رو ش شامل فعاليت هايي است كه سعي دارد پروژه را در غالب روابطي رسمي و صوري بگنجاند . بعنوان مثال طراحي مدل رياضي است و از قالب روابط رياضي استفادهد مكند . بنابراين از مزاياي اين روش روشن و واضح بودن طرح و همينطور سازگاري و جامع بودن آن است و بيشتر در سيستم هايي كه از حساسيت بالايي برخوردارند استفاده ميشود اما داراي معايبي نيز ميباشد ، چون معمولا قالب انتخاب شده وقت گير و داراي هزينه ميباشد لذا كاربردهاي آن محدود ميباشد و همواره نياز به آموزش كلي احساس ميشود و در نهايت باز هم به دليل قالب در نظر گرفته شده ايجاد ارتباط با مشتري به مشكل بر ميخورد اما روي هم رفته مدل جامعي به شمار ميرود .

۱۲. روش تكنيك هاي نسل چهارم (Fourth Generation Techniques)

همانطوري كه ما از علم مهندسي نرم افزار براي استفاده بهتر از علوم ديگر بهره مند ميشويم در اين روش نيز براي استفاده بهتر از نرم افزار از خود علم مهندسي نرم افزار كمك ميگيرد . در اين روش توليد كننده نرم افزار بايد بعض از خصوصيات نرم افزاري را كه قرار است توليد شود در سطح بالا مشخص ساخته و سپس به كمك همان ابزار هاي موجود كدي بر مبناي مشخصات تعريف شده توسط توليد كننده بطور خود كار ايجاد شود . عمده ترين اشكال اين روش همان حالت خاص بودن آن ميباشد ، چون معمولا تكنيك هاي گفته شده خيلي جنبه عام ندارد . از آنجايي كه در اين مدل آناليز محيطي تقريبا از بين ميرود بنابراين چگونگي گفتگو بين مشتري و توليد كننده بسيار مهم است ، چون معمولا در اين مدل مشتري ممكن است در رابطه با خواسته هاي خود كه در ابتدا بيان مي دارد اطمينان كافي نداشته و نسبت به برخي از واقعيت هاي موجود در محيط سيستم با ديده ترديد بنگرد . يك ابزار 4GT بايد داراي حداقل امكانات زير باشد :

۱. امكان گفتگو و پرس و جو با مشتري
۲. ابزار و امكاني براي توليد كد
۳. قابليت گرافيكي خوب و ابزاري براي تعريف صحنه نمايش

هنوز سيستمهايي كه به كمك اين مدل طراحي مي شوند از قابليت نگهداري خوبي برخوردار نيست و از لحاظ سهولت نيز آسانتر از زبانهاي برنامه نويسي نميباشد . هزينه ايجاد اين سيستم ها نيز زياد پا يين نيست . در هر حال از اين مدل فعلا بيشتر در سيستم هاي تجاري مخصوصا پايگاه هاي داده اي استفاده ميگردد و كمتر در كارهاي مهندسي و سيستمي كاربرد دارد .

۱۳.روش مارپيچي برنده برنده (WINWIN)

در روش حلزوني يكي از اعمال چارچوبي ، به برقراري ارتباط با مشتري مربوط مشد . هدف اين عمل ، روشن كردن خواسته ها از سوي مشتري است . در حالت ايده آل ، سازنده صرفا از مشتري مي پرسد كه چه چيزي مورد نياز است و مشتري جزئيات لازم براي پيشرفت كار را فراهم مي آورد . متاسفانه چنين چيزي به ندرت رخ ميدهد . در حالت واقعي ، مشتري و سازنده وارد يك فرايند گفتگو و بحث ميشوند و در اين فرايند ممكن است از مشتري خواسته شود تا ميان عملكرد ، كارايي وويژگي هاي ديگر سيستم يا محصول ، در مقابل هزينه و زمان بازيابي ، موازنه برقرار كند .

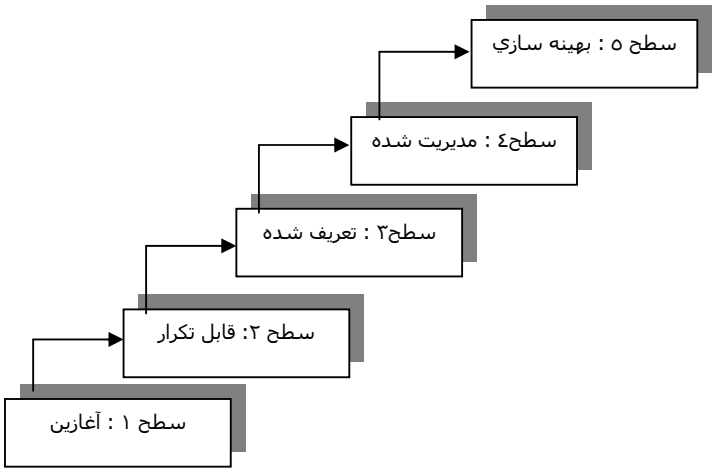
در بهترين مباحثات سعي مي شود تا يك نتيجه «برد برد » حاصل شود . يعني مشتري با دستيابي به محصول يا سيستمي كه واجد اكثر نيازهاي اوست برنده ميشود و سازنده با كار كردن در چارچوب مهلت و بودجه اي واقعي و قابل حصول ، برنده ميشود .
روش مارپيچي WINWIN بوهم ، مجموعه اي از اعمال مباحثاتي را در آغاز هر دور جديد از مارپيچ تعريف ميكند . به جاي يك عمل ارتباط با مشتري ، اعمال زير انجام مي شود :

١. شناسايي واگذارنده كليدي سيستم يا زير سيستم .
٢. تعيين « شرايط برد » واگذارنده .
٣. بحث و گفتگو درباره شرايط برد واگذارنده ، براي مصالحه و توافق آنها در يك مجموعه شرايط بردبرد ، براي همه موارد مربوط (از جمله تيم پروژه نرم افزار)

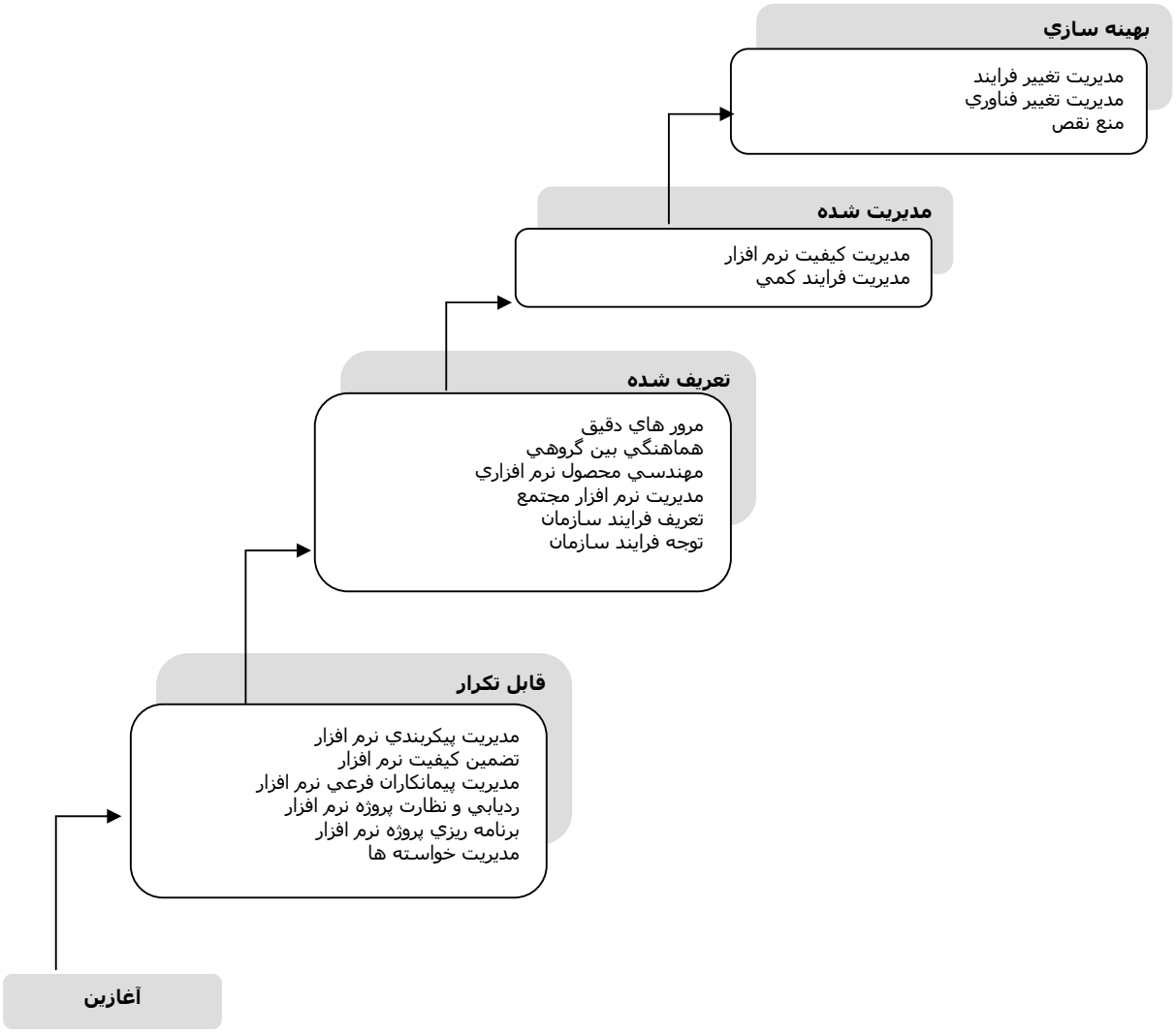
با به پايان بردن موفقيت آميز اين مراحل اوليه ، يك نتيجه بردبرد حاصل ميشود كه ملاك كليدي براي پيشروي به سمت تعريف نرم افزار و سيستم ميشود .
علاوه بر تاكيداتي كه بر زود هنگام بودن مباحثات ميشود ، مدل WINWIN سه مرحله فرايند دارد كه نقاط لنگرگاه نام دارند . اين نقاط به تعيين زمان تكميل يك چرخه حول مارپيچ كمك كرده نقاط عطفي براي اتخاذ تصميم ، قبل از ادامه پروژه هستند .
در اصل نقاط لنگرگاه سه ديد متفاوت از پيشرفت پروژه را در راستاي طي كردن مارپيچ بدست ميدهند . نخستين نقطه لنگرگاهي ، كه اهداف چرخه حيات نام دارد ، براي فعاليت عمده در مهندسي نرم افزار ، مجموعه اي از اهداف را تعيين ميكند . براي مثال ، بعنوان بخشي از اهداف چرخه حيات ، يك مجموعه اهداف با تعريف خواسته هاي محصول / سيستم سطح بالا همراه ميشود . دومين نقطه لنگرگاهي ، كه معماري چرخه حيات ، نام دارد ، اهدافي را تعيين ميكند كه بايد به موازات تعريف معماري سيستم و نرم افزار برآورده شوند . براي مثال ، به عنوان بخشي از معماري چرخه حيات ، تيم پروژة نرم افزاري بايد تشريح كند كه موجوديت مؤلفه هاي آماده و قابل استفاده مجدد را ارزيابي كرده ، تاثير آنها را بر تصميم گيري هاي معماري مد نظر قرار داده است . قابليت عملياتي اوليه ، سومين نقطه لنگرگاهي است و مجموعه اي از اهداف است كه عبارتند از آماده سازي نرم افزار جهت نصب / توزيع ، آماده سازي سايت پيش از نصب ، وكمك به تمام كساني كه نرم افزار را استفاده يا پشتيباني مي كنند .

١٤. روش CMM(Capability Maturity Model)

ماوريت موسسه مهندسي نرم افزار (SEI) در دانشگاه كارنجي – ملون ، انتقال فناوري نرم افزار است . براي بهبود قابليت هاي صنعت نرم افزار در آمريكا ، بخصوص قابليت هاي سازمان هايي ايجاد شد كه بودجه نظامي دريافت ميكنند تا پروژه هاي دفاعي بزرگ را ايجاد نمايند . در اواسط دهه ١٩٨٠ ، SEI روش هاي برآورد قابليت هاي پيمانكاران را مورد مطالعه قرار داد .
نتيجه اين برآورد قابليت ، مدل CMM بود . اين موسسه نفوذ زيادي در جامعه مهندسي نرم افزار، بخصوص بهبود فرآيند داشته است . مدل SEI فرآيندهاي نرم افزار را به پنج سطح تقسيم ميكند . اين پنج سطح به صورت زير تعريف ميشوند :



۱. سطح آغازين ، در اين سطح ، سازمان رويه ها يمديريت يا برنامه هاي اثر بخشي براي پروژه ندارد . اگر رويه هاي رسمي براي كنترل پروژه وجود داشه باشند ، راهكارهاي سازماني وجود ندارند تا تضمين كنند كه آنها بطور سازگار استفاده شده اند . سازمان ممكن است نرم افزار را با موفقيت ايجاد كند ولي ويژگيهاي نرم افزار(كفيت و...) و فرايند نرم افزار (هزينه ، زمان و...) قابل پيش بيني نيست .
۲. سطح قابل تكرار، در اين سطح سازمان داراي مديريت رسمي ، تضمين كيفيت ورويه هاي كنترل پيكربندي است . به اين علت قابل تكرار ناميده ميشود كه سازمان ميتواند پروژه هاي يكسان را تكرار كند . اما ، مدل فرايند رسمي وجود ندارد . موفقيت هاي پروژه به مديراني بستگي دارد كه تيم را تشويق ميكنند و به فرهنگ سازماني بستگي دارد كه بعنوان توصيف فرايند شهودي عمل ميكند.
۳. سطح تعريف شده ، دراين سطح، سازمان پروژه اش را تعريف كرده است و مبنايي براي بهبود كيفي فرايند در اختيار دارد . رويه هاي رسمي تضمين ميكنند كه فرايند تعريف شده در پروژه نرم افزاري رعايت ميشوند .
٤. سطح مديريت شده ، سازمان سطح ٤ ، داراي فرايند تعريف شده و برنامه رسمي براي جمع آوري داده هاي كمي است .معيارهاي محصول و فرايند جمع آوري شده در فعاليت بهبود فرايند به كار گرفته ميشوند .
٥. سطح بهينه سازي ، در اين سطح ، سازمان معتقداست كه بهبود فرايند را ادامه دهد . هزينه و برنامه بهبود فرايند مشخص ميشود و بعنوان بخشي از فرايند سازمان است .



درك سه سطح اول مدل نسبتا ساده است . نواحي فرايند كليدي شامل شيوه هاي كاري است كه فعلا در صنعت استفاده ميشود . بعضي از سازمانها به سطوح بالاتر اين مدل رسيدند ، اما استاندارد ها و شيوه كاري كه در آن سطح به كار گرفته ميشدند ، به خوبي درك نشدند . در بعضي از موارد به دلايلي شرايط سازماني ممكن است شيوه هاي كاري اين روش رعايت نشود .
مشكلات موجود در سطوح بالاتر ، مفيد بودن مدل را نقض نميكند . اغلب سازمانها در سطوح پايين تر هستند . و اما اين مدل سه مشكل عمده دارد كه ممكن است نتواند قابليت سازمان را براي توليد نرم افزار كيفي پيشگويي كند . اين مشكلات عبارتند از :

۱. مدل به مديريت پروژه تاكيد دارد نه به توسعه محصول ، فناوري هاي مورد استفاده در سازمان مثل نمونه سازي ، روش هاي رسمي يا ساخت يافته ، ابزارهاي تحليل ايستا و... را در نظر نميگيرند.
۲. تحليل و رفع ريسك را به عنوان فناوري كليدي در نظر نميگيرند ،
۳. دامنه كاربرد مدل تعريف نشده است .

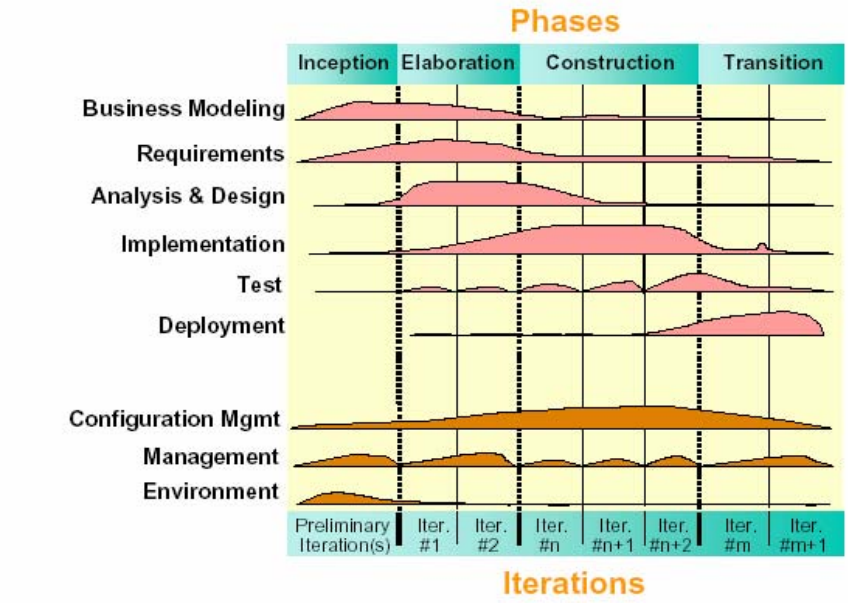
۱۵. روش RUP (Rational Unified Process)

با پيشرفت تكنولوژي كامپيوتر نياز هر چه بيشتر به گسترش علم نرم افزاري نيز احساس مي شد كه با پيدايش متدولوژي هاي همانندِSSADM و روش آبشاري و روش آبشاري آغاز شد . در ابتدا ،اين روشها مناسب بود و جوابگوي نيازهاي آن زمان بودند ولي با افزايش داده ها و پيدايش مفاهيمي همچون شبكه ، Web و ... ديگر كارايي لازم را جهت پياده سازي و هدايت پروژه هاي نرم افزاري نداشتند . پس مفاهيم برنامه نويسي شي ء گرا پا به عرصه وجود گذاشت و در سال ۱۹۹۱ بطور جدي مورد مطالعه و بحث قرار گرفت . استفاده از اين روشها و متدهاي برنامه نويسي قدرت و انعطاف بسياري را به برنامه ها داد و شركت هاي نرم افزاري توانستند با كاهش هزينه ها و بهينه سازي كد هاي خود ، نرم افزارهاي قويتري را به بازار عرضه كنند ولي اين روش جديد نيز نياز به مديريت و يكپارچگي داشت . پس روشها و متدولوژي هاي جديدي مطرح شد كه شامل Booch - OMT - OSE و ... مي باشد . در سال ۲۰۰۰ شركت Rational روشي را تحت عنوان RUP (Rational Unfied Process) مطرح ساخت كه بعد از روش MSF شركت ماكروسافت به دنياي نرم افزار عرضه شد و امروزه از طرفداران بسياري برخوردار است .

پالاينده يكپارچه Rational در اصل يك متدولوژي است كه در جهت كنترل و انجام پروژه هاي نرم افزاري در نظر گرفته شده است . در اصل اين چارچوبي در جهت انجام صحيح و موفق پروژه هاي نرم افزاري مي باشد كه كليه مراحل انجام يك پروژه كه با معماري و آناليز سازمان شروع شده و به تست نرم افزار و ارائه Gold Release ختم مي شود را در بر مي گيرد .
به سه علت RUP را يكپارچه مي نامند :

۱. اين متدولوژي از يكپارچه سازي سه متدولوژي معروف ديگر بوجود آمده است كه شامل Booch - OMT - OSE مي باشد .
۲. از UML درجهت كارهاي خود استفاده مي كند . در واقع مي توان گفت UML خود ثمره RUP مي باشد و اين بسيار خوب است كه متدولوژيي با خودش گسترش يابد .
۳. با مفاهيمياز قبيل Class ، Object و ... مفاهيم سده و ثابتي هستند ولي قبلا متدولوژي ها علامتهاي خاصي داشتند كه اكنون همه آنها يكسان شده اند .

در داخل RUP يك چارچوب توليد نرم افزار است كه ما آنرا براي سازمان و پروژه خود Customize مي كنيم و مي توان گفت كه در واقع يك Process Framework است . شكل زير ساختار اصلي RUP رامشخص مي كند . اگر در بعد از زمان به آن نگاه كنيم شامل ٤ فاز مي باشد و اگر در هر لحظه به آن نگاه كنيم شامل ۹ ، Framework خواهد بود .



خصوصيات RUP چيست ؟

١. مبتني بر همان معماري است (معماري ارتباط با طراحي دارد و مي توان گفت نوعي طراحي است كه به اجزاء اصلي مي پردازد ولي طراحي به جزئيات نيز وارد مي شود) هچنين مي توان گفت معماري يكسري اجزاء و ارتباط بين آنها است كه سيستم را مي سازد و ما را به سمت CBD(Component Base Development) راهنمايي مي كند .

٢. Usecase Driven يكي از مشكلات OOA اين بود كه مي گفتند با هر روشي تبديل و كار كنند و بعد بتوان آنرا به شي گرا تبديل كرد . يعني مثلا پروژه SSADM طراحي كرده و بعدا به شي گرا تبديل نمود ولي عقيده اشتباه بود و حتما تحليل شي گرا بايد صورت بگيرد . خصوصيت خوب شي گرا كه در ديگر روشها نمي باشد اين است كه Notation اي كه استفاده مي شود در همه مراحل يكي است يعني مفاهيمي از قبيل شي كلاس ، روابط كلاسها و ... در تمامي مراحل يكي است .

اهميتي كه Usecase Driven دارد اين است كه با زبان بيشتري نوشته مي شود :

١. مشتري مي تواند آنرا بفهمد و بسيار مناسب براي تشخيص نيازمندي هاي سيستم مي باشد . در بخش تحليل و طراحي از روي Usecase ها تحليل و طراحي انجام مي دهيم و مسائلي مانند مديريت پروژه نيز تحت تاثير Usecase ها هستند كه ما آنها را دسته بندي كرده و مديريت مي كنيم . همچنين System Manual و Help Manual ها هم تحت تاثير Usecase ها ايجاد مي شوند .

٢. Incremental مي باشد . يعني پروژه بصورت چهار مرحله حلقه اي جلو مي رود ولي در هر مرحله چرخش يك دسته از Usecase ها كامل و آماده استفاده مي شود و كليه اين كارها در ٩ Workflow كه در شكل يك مشخص شده بود و قابل مشاهده است .

چرا قبلا معماري نداشتيم ؟

١. تعريف دقيقي از آن وجود نداشت .

٢. ابزار مناسبي نبود .

٣. فرايندي نداشتيم كه به ما حكم كند و ما را وادار به استفاده كند .

تعيين معماري و نيازهاي سازمان جزء مهمترين بخشهاي هر متدولوژي مهندسي نرم افزار مي باشد كه هدف و جهت حركت پروژه ها از مشخص مي كند .

١٦. روش ساخت يافته تجزيه وتحليل و طراحي سيستم ها (SSADM)

SSADM يك رويكرد سيستماتيك جهت تجزيه وتحليل كاربرد هاي فناوري اطلاعات ميباشد . اين روش در اوايل دهه ١٩٨٠ ميلادي تحت كنترل دولت انگلستان به صورت استاندارد درآمده است . هدف آن كمك به تيم پروژه فناوري اطلاعات است تا با دقت نيازمندي هاي يك سيستم فناوري اطلاعات را كه از راهبرد فناوري يك سازمان حمايت ميكند را مورد تجزيه وتحليل قراردهند و يك سيستم فناوري اطلاعات را بگونه اي مقرون به صرفه نيازمندي ها را تامين مينمايد را طراحي نمايند .

اين روش تعداد زيادي از فنون آزمايش شده را بگونه اي گرد هم مي آورد كه با يكديگر سازگار باشند و يك چارچوب خوب مستند سازي شده را فراهم ميسازد . اصل اساسي آن اينست كه سيستم به كاربران تعلق دارد و بدين لحاظ شركت آنها را در فرايند طراحي و ساخت حائز اهميت ميباشد. آنها در طول حقيقت يابي مورد مشورت قرار ميگيرند و محصولات مراحل مختلف SSADM با كاربران مورد بحث قرار ميگيرند تا كامل بودن ، صحت و درك آنها را مورد تائيد قرار دهند . بعضي از محصولات تنها با كمك فعال كاربران (مثلا تعاريف وظيفه ها)بدست مي آيند . مديريت كاربران در طول طراحي وساخت سيستم اطلاعاتي تصميمات عمده را اتخاذ مينمايد وتنها زماني كه كاربران محصولات مرحله جاري را پذيرفته اند مرحله اي ميتواند به مرحله ديگر پيش رود .