**Determinarea si analiza cerintelor :**

Cerinte –o afirmatie a ceea ce sistemul trebuie sa faca , a caracteristicilor sist care se pot modifica in timp.

Cerinte – functionale(ce trebuie sa faca un user ) si nonfucntionale(proprietati pe care trebuie sa le aiba un system(operationale, culturale,politice) ;

Avem 3 tipuri de sisteme pentru care scriem cerintele :

1.Sist. Existent (AS-is) – cerinte pentru sist care trebuie sau nu computerizat

2.sist de perspectiva(to be) –sistem ce trebuie sa se bazez pe cerintele create

3. Propunerea de system(System proposal()) –rezultat livrat la faza de analiza.

Surse de cerinte :clientii,utilizatorii sistemului,administrator,parteneri, experti din domeniu, analistii din industrie)

**Tehnici de colectie a cerintelor**

Caracteristicile care colecteaza cerinte –

1. impertinent – insistent si sa dai intrebari

2. impartialiatate –neutru

3. relaxarea constringerilor –(scaparea)

4. atentia la detalii

5.reframing –repozitionarea problemei

Tehnici de colectare :

1. Interviul : - intrebari u sfirsti deschis/inchis

2.Chestionare : p/u esantioane foarte mari, corectitudine,esantionul correct,contine intrebari deshis /inchise

3. observarea lucratorilor

4. analiza do.Business

5.Jad –join app design -grupuri de determinarea cerintelor

6. Prototipizarea.

**Interviul**

Punctele la care trebuie de atras atentia :

* Colectarea faptelor, opiniilor (opinie => fapte)
* Limbajul corpului, emotii, p/u a obtine info adaugatoare

In Plan Set de puncte care trebuie sa fie discutat (Checklist):

* Planificarea timpului
* Neutru – impartial
* Sa nu tacem
* Sa putem asculta mai multe puncte de vedere

Selectarea intervievatilor :

1) in baza info de care avem nevoie

2)obtinerea punctelor diferite de vedere(manager, user)

3)sistemul politic in companie este mai influent sau invers(grupuri de lucru)

Crearea intrebarilor :

Tipuri de intrebari : 1)deschise/ inchise – opinii, discutii / raspunsuri bine definite

Intrebari de stimulare (de ce ? Puteti sa explicate)

Organizarea intebarilor : interviu structurat/nestructurat (bazat pe conczuzii / nu tare concret)

Realizarae interviului : inregistrati toata info, intelegerea termenilor folositi,separarae faptelor de opinii,dati timp pentru a rapsunde la intrebari,sumar, concluzii.

**Chestionar**

un set de intrebari care este des transmis unui grup mare de oamenii . Au un comportament inchis mai des, poate fi electronic sau pe hirtie, selectarea corecta a esantionului, intrebari usor de analizat

**Observarea –** se realizeaza prin monitorizarea lucratorilor, validatatea info poate fi verificata, observarea perioadelor in dperosanele sunt prea incarcate sau nu (echilibrare)

**Analiza doc Business –** studierea materialelor despre system, informative poate fi obtinuta prin problemele in cadrul stistemului, la aparitia anumiro necesitati,valorile organizatiei(departamente),reguli de procesare a datelor, circumstante d eprocesare special a datelor.

**Sesiunile Jad**

grupuri structurate ,focusate pe procesul determinarii cerintelor si implica o echipa de proiect manageri si utilizatori care lucreaza impreuna pot reduce timpul de analiza pina la 50 %.

Participanti JAD – coordinator(antrenat in tehnici JAD), participant(colegi participa la discutii ), secretar(inregistreaza datele).

Sesiunile Jad – au nevoie de timp asigurat (jum de zi pe parcursul citeorva saptamani), trebuie sa fie sustinere managementului de a elibira de tascuri zilnice lucratoare, planificare esentiala.

**Prototipizarea -**

Model al unui system ,pune accent pe aspect , tehnica de colectarea cerintelor preeliminare p/u a avea versiuni initiale a solutiei(prototipuri). Clientul vizualizeaza aceste cerinte si adauga cerinte aditionale.

Proprietatile prototipizarii – process iterativi, repetitiv, a carui rezultat reprezinta o versiune mai simplificata a sistemului, cu scopul de a dezvolta specificatii concrete p/u system final .

**Utilizam prototipizarea atunci :** cerintele utilizatorului nu sunt clare, sunt implicate putini utilizatori, cind desingul este complex si necesita o forma complexa, cind avem instrumente pentru a crea rapid un prototip.

**Aspecte problematice a prototipizarii :** tendinta de a Evita descrierea formala, dificultate in adaptarea designului de auditoriu mai mare, nu este luata in consideratie partajarea datelor cu alte sisteme.

**Alte tehnici :**

Interviu 1 la 1 - clientul si anlistul discuta

Pe grup - mai multe personae lucreaza cu un grup de oameni

Brainstorming – in unele cazuri cind cerintele trebuie sa fie create, de facut ceva inovativ.

Analiza interfetelor – process in care analizam punctele de interactiune cu sisteme externe dispositive sau oamenii .

Studierea sist similar – punctul de start al unui system sunt des sisteme existente, care prin comparative pot da solutii.

Reverse Engineering **–** utilizam atunci cind avem cod si trebuie sa scrim documentatie pe el,tine mai mult de cerinte tehnice.

**Analiza Cerintelor –** un process care reduce diferenta dintre cerintele de system si designul de proiect.

Ofera proiectantului rezultatul acestui process(model de informatie de system ,de functionare)

Obiectivele analizei : identificarea necesitatilor clientilor, analiza economica si tehnica, alocarea functiilor elementelor de system ,stabilirea unui plan si constringeri.

Fazele analizei cerintelor : recunoasterea problemei, evaluarea si sinteza, modelarea, specificatia, review.

Intrebarile managerului in timpul analizei : cit de mult effort este depus, cine face analiza, de ce este dificult,cine plateste p/u asta.

Studiul de fezabilitate : fizabilitate economica (costul,beneficiu) ,tehnica(cit de realizabile este),juridical (patente care interzic realizarea produsului) , alternative(alte solutii pentru a rezolva problema))

**Specificarea sistemului :** introducere, descriere fucntionala,descriere subsistem,produse, anexe.

**Validarea cerintelor-**

correct? –daca nu se contrazic atunci nu s-a schimbat situatia

consistent –se integreaza cu restul cerintelor

complete ?– toate proprietatile dorite/necesare

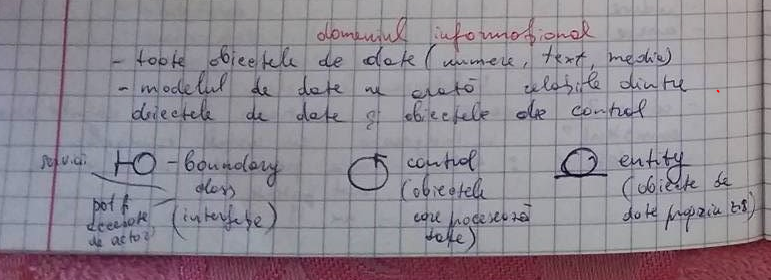
testabile ?

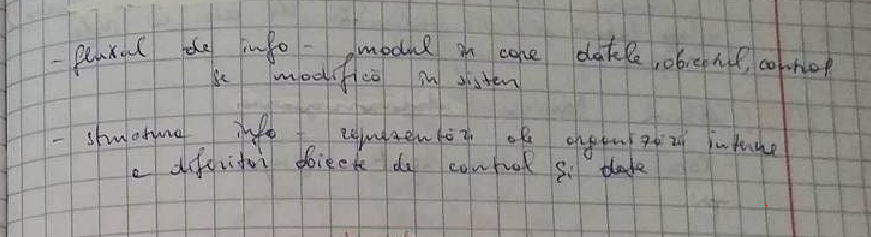
**Document de definire a cerintelelor :** contien scopul general al documentului,domeniul sistemuui si obiectivele lui ,descrierea abordarii.

**Use Case –** funcitonalitatile sistemului disponibile actorilor.

**Principii de analiza –** domeniul informational trebuie sa fie present si inteles, modele ce arat info despre system trebuie dezvoltate in cadrul unui analizei,analiza se porenste de la info esentiala si duce inspre detalii de implementare.

**Domeniul Inofrmational**





**Modelarea –** modelul de date (relatiile intre obiect system) ,functional(functiil system),comportament(sistemul raspunde la evenimente)

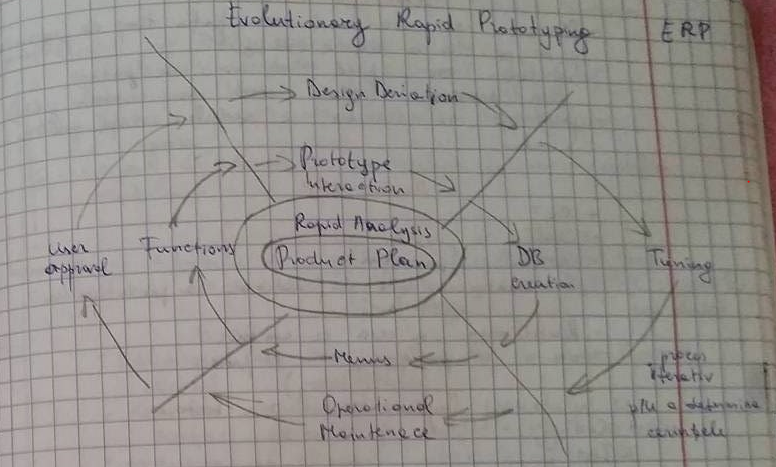
**Partitionarea -** process de dezvoltare comprotamentului functiilor si datelor, are 2 aspecte (orizontala si vertical) , cel orizontal permite sa obtinem info despre acelasi nivel de abstractizare , cel vertical este mai detaliat, ne adincim mai mult prin schimbare nivelului de abstractizare.

**Principii de specificare:** separarea functionalitatilor suplimentare, folosirea limbajului orientat pe process, trebuie sa contina componente software ale process,specificarea trebuie sa contina mediul tehnic,operational , cognitive.

**Reviews Specificatiilor** – trebuie realizata de client si de dezvoltatori de software , odata aprobata ele devin un contract dezvoltat. software , sunt greu de testat .

Tehnici de evaluare a specificatiilor - cerinta trebuie sa fie inteleasa de utilizatorii naivi , cerinta formeaza baza pentru proiectare . Testarea formeaza aspectul extern al sist, cazurile de testare sunt generate din cerinte (+use case) .

Avem 2 tipuri de prototipizari (la specificatii) – cu pierderi(renuntarea la modul) si evolutionara(reutilizarea prototipului)



**ERP – s**copul este de a servi mai putin cod inutil. Riscurile pot fi necunoasterea cerintelor clientilor,tehnologiei, interfetei.

Iata citeva din riscuri ERP-livrare premature, nu toate functionalitatile vor fi incluse in produsul final.

Sfaturi erp – focusarea pe acel detaliu care necesita cel mai putin timp de programat,nu servim cod pina cin fucntionalitaea nu este facuta.

Analiza ERP –ca? cind? Costul? Care sunt conditiile de finisare a prototipului?

Implementare si testare : cod, test, debug,go to 1(repeat).

**Each spin cycle –** la fieare iteratie fiekare modul trebuie evaluat(decizii : lasam asa cum este, rescriem,inlocuim cu alt cod existent).

**Erori tipice in analiza cerintelor** – clinetii nu inteeg ceea ce vor, cerintele se schimba pe parcursul proiectului,clientii pun termini nerezolvabili de rezolvare ,probleme de comunicare intre client si manager,echipa de dezvoltare nu intelege politia organizational a clientului.

**Cum rezolvam astfel de probleme –** trebuie sa ne asiguram ca petrecem destul timp la inceputul poriectului (scop,obiective) ,descrierea unei viziuni concrete a proiectului care contine riscurile si beneficiile oferite de system, stabilim un proes clar de primire, analiza si implementare a cererilor de schimbare, discutam termenul limita cu clientul,ne asiguram ca avem resurse disponibile pentru a intra in limita timpului, sa facem notite la fiekare intilnire, sa fim consistenti in utilizre cuvintelor,sa analizam reteaua de personae si sa stabilim fiekare ce tip de inf detine.

**Etapa de proiectare a arhitecturii** :

def –structura unui system care cuprinde elemente software ,proprietati extern vizibile ale acestuia , si relatiile dintre ele.

**Cum de implementat?**

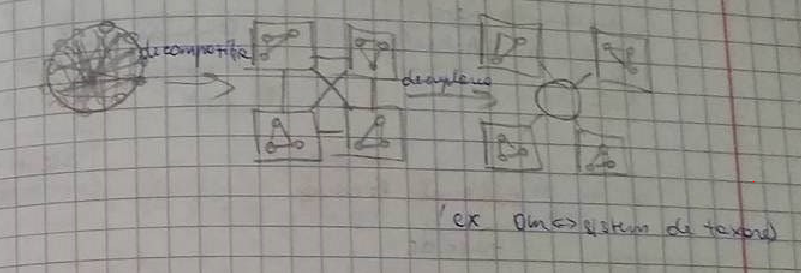
arhitectura soft contine setul de decizii importante ce tin de organizrea unui system software incluzind selectarea elementelor structural si a interfetelor acestora, specificam colaborarea dintre aceste elemente.

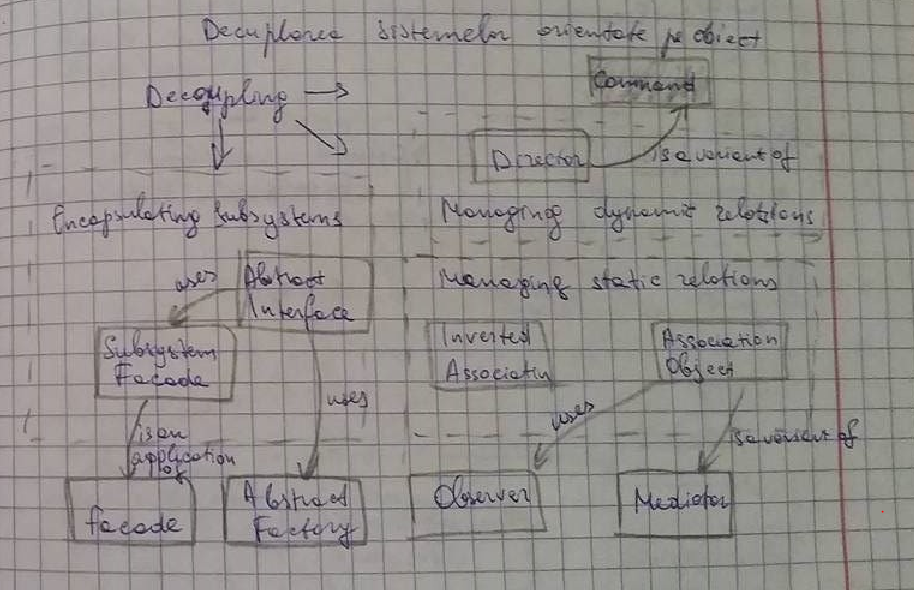
**Importanta arhitecturii** : studierea si intelegerea mai bine a infrastructurii proectului dat, utilizarea instrumentelor modern ne va usura sarcina de lucru.

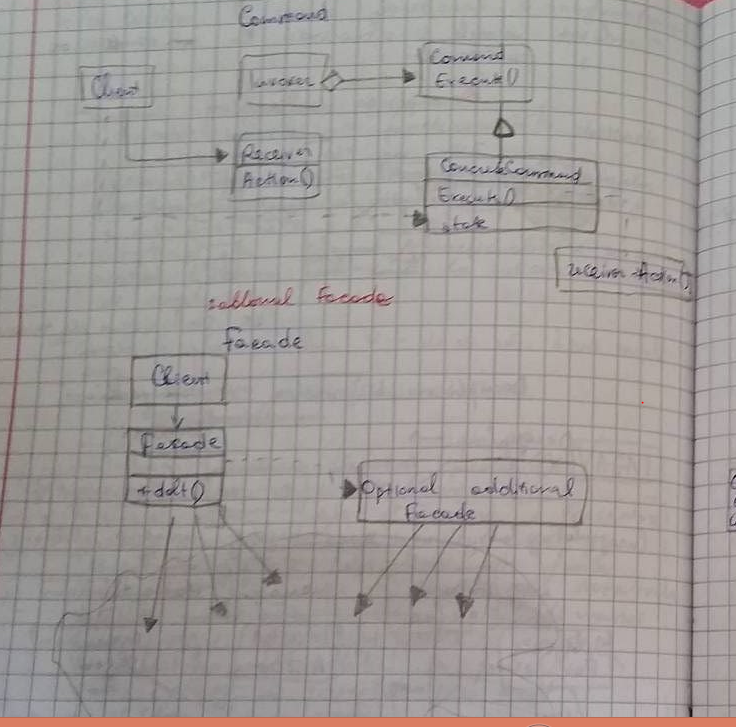
**Scopurile arhitecturii :** legatura dintre cerintele business si cele tehnice prin intelegearea cazurilor de util si gasirea meotdelor de implementare a acestra in software, identifcarea cerintelor care afecteaza structura aplicatiei,realizarea tutuor cerinteleor use case.

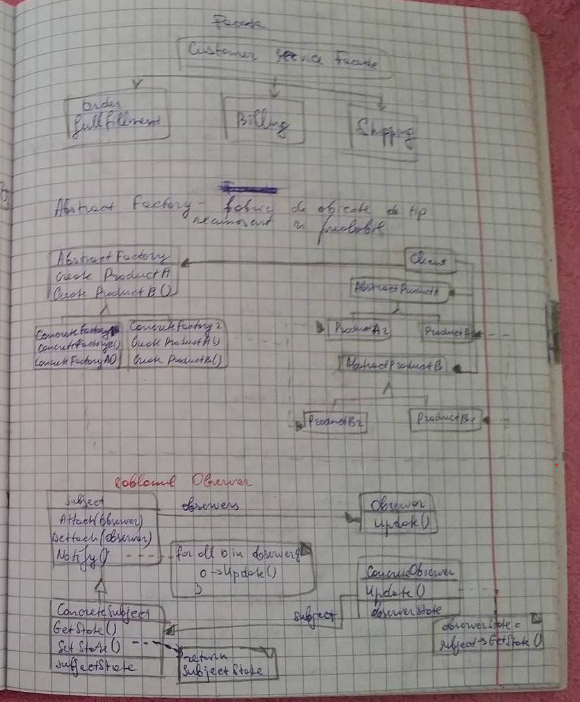
**Principii cheie in arhitectura :** separarea problemelor (divizara aplicatiei in fucntionalitati distincte,mnimizarea punctelro de interactiuni,extinderea functionalitatilor, minimizati proeictarea premature ,proiectam doar ceea ce este necesar, utilizam elemente uml pentru analiza si reducerea riscurilor .

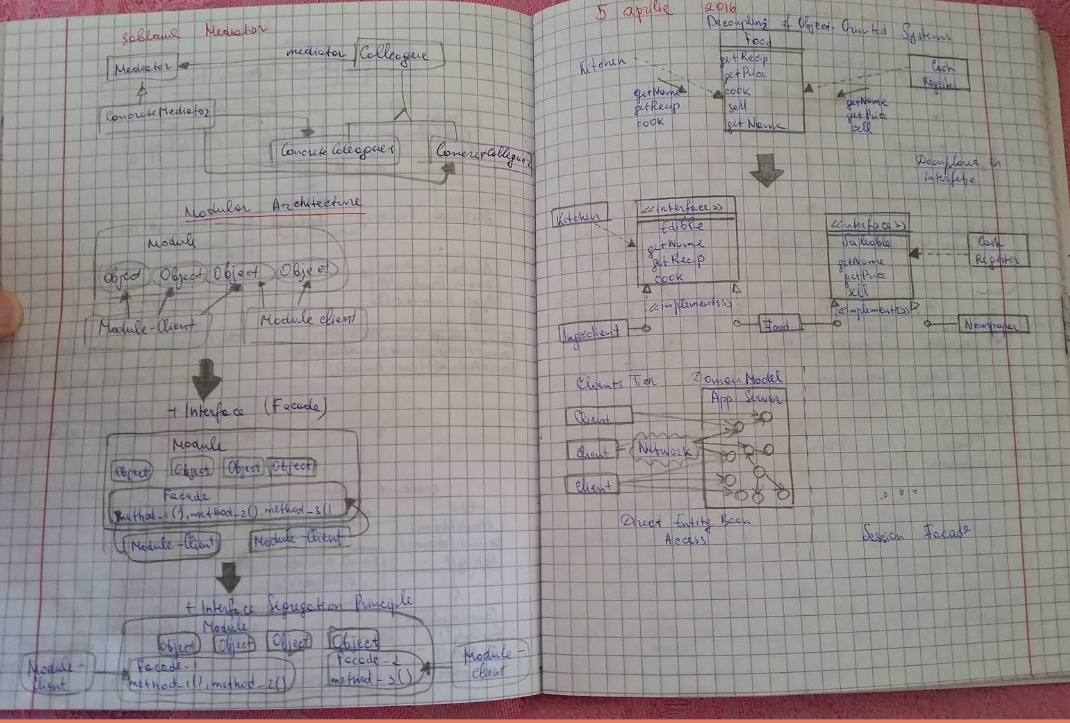
**Decompozitia si decuplarea -**



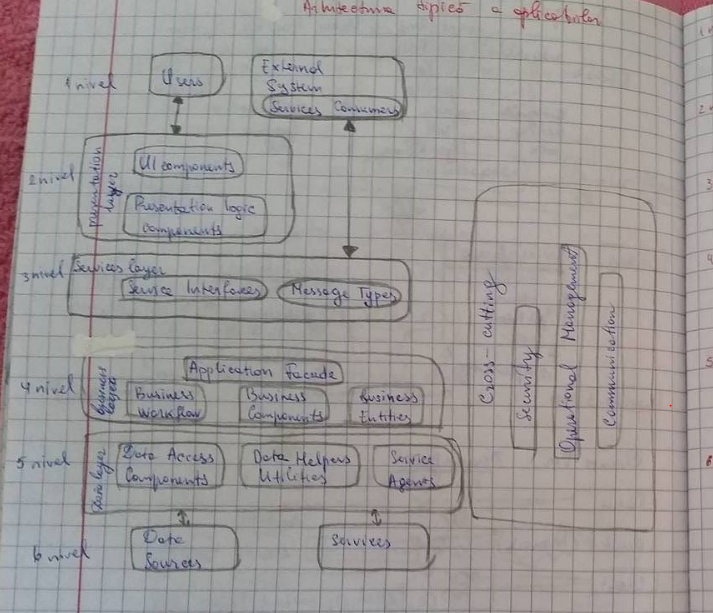








**Arhitectura tipica a aplicatiilor**



1 nivel User ,sisteme externe

2nivelul de prezentare (componente uml, si logica a prezentarii) – are ca scop separarea interfetei utilizatorului de logica sistemului, ne permite sa schimbat logica de prelucrarea a datelor si modul de vizualizare a datelor.

3nivel de servicii (interfetele serviciilor,tipuri de mesaje) – stbilirea interfetei de comunicare cu client

4nivel business (cascada aplicatiei,fluxul business,componente business,entitati business) –are ca scop de a implementa modelul de prelucarare a datelor .

5nivel de date ( nivelul de acces de date,utilitati pentru lucrurile cu datele,agenti de servicii) – implementeaza modul de acces la date si ne ofera independent de formatul acestora si modul de stocare.

6 nivel componente (surse de date,servicii) -

Cross Cutting (principii commune) –securitate,management operational,comunicare. – managementul modificarilor logistice in interiorul aplicatii, si comunicare prin protocoalele necesare.

Principiile arhitecturii : cunostintelor minimale(o component obiect nu trebuie sa cunaosca detalii ale altor componente obiect) , nu te repeta(o fucntionalitate trebuie specificata doar intrun modul.)

**Practici de proiectare :** preferati compozitia in loc de mostenire , deoarece mostenirea mareste dependent claselor una de alta, stability un stil de cod de conventii si nume,pentru a facilita colaborarea in echipa, mentineti calitate sistemului utilizind tehnici de QA automat in timpul dezvoltarii , cum ar fi analiza dependentelor si analiza static a codului.

**Principii de implementare a nivelelor :**

1.separarea ariilor de fucntionalitati - aplicatia trebuie impartita in functionalitati distincte, care au intersectie cit mai mica posibila.

2.stabiliti explicit modul de comunicare intre nivele -este important stabilirea protocoalelor, si foramtul de date trimis.

3.utilizati abstractii pentru a implementa cuparea slaba intre nivele.

4.Nu utilizati diferite tipuri de componente la acelasi nivel logic

5.pastrati formatul de date consistent in interiorul unui nivel

**Idee cheie pentru proiectare:**

**1)** determinarea tipului de aplicatii –pe ce fel de platforma,ce tip de aplciatii(ex service) internet conection

2)strategii de instalare –in dependent de aplicatii si strategiile difera, (deployment a instala pe google play,appstore)

3)tehnologii potrivite – native android, ios, window phone.

4)attribute de calitate –metrici de calitate

5)factori aspect commune – crosscutting -se stabiles mecanisme de Securitate, comunicare, de asemean recomandabil sa se ia in consideratie autentificare,