UNIVERSITATEA
DIN BUCUREȘTI

Metode de dezvoltare software

Cerințe software

06.03.2019

Alin Ștefănescu

Cerințe

Prezentare bazată pe:
“Software Engineering – 10th ed. (2015)” de Ian Sommerville
http://iansommerville.com/software-engineering-book

Negocierea cerințelor… în practică

de la dilbert.com

Despre cerințe

- **Cerinţele** sunt descrieri ale serviciilor oferite de sistem şi a constrângerilor sub care acesta va fi dezvoltat și va opera
- Cerinţele pornesc de la afirmaţii abstracte de nivel înalt până la specificaţii matematice funcţionale detaliate
- Cerinţele îndeplinesc mai multe funcţii:
  - pot constitui baza negocierilor pentru un contract – ca atare, trebuie să fie deschise diverselor interpretări
  - pot constitui baza contractului în sine – ca atare, trebuie definite cât mai detaliat

Tipuri de cerințe

- **Cerinţe utilizator** USER STORIES
  - afirmaţii în limbaj natural şi diagrame ale serviciilor oferite de sistem laolaltă cu constrângerile operaţionale.
  - scrise pentru clienţi.
- **Cerinţele sistemului**
  - un document structurat stabilind descrierea detaliată a funcţiilor sistemului, serviciile oferite şi constrângerile operaţionale.
  - poate fi parte a contractului cu clientul.

Utilizarea cerințelor

- **Cerinţele utilizator** se adresează:
  - utilizatorilor finali
  - inginerilor si managerilor clientului
  - proiectanţilor de sistem (arhitecţi)
  - managerilor de contracte

- **Cerinţele de sistem** se adresează:
  - utilizatorilor finali
  - inginerilor clientului
  - proiectanţilor de sistem (arhitecţi)
  - programatorilor

Utilizatorii documentului cu cerințe

- **clienţii**: impun cerinţele şi verifică apoi dacă acestea sunt conforme cu nevoile lor. Pot cere modificări ale cerinţelor.

- **managerii**: utilizează documentul pentru a stabili termenii contractului şi a planifica procesul de dezvoltare

- **inginerii de sistem**: utilizează documentul pentru a înţelege ce trebuie dezvoltat

- **inginerii de la testare**: utilizează documentul pentru a proiecta testele de validare

Cerinţe funcţionale şi non-funcţionale

- **Cerinţe funcţionale:**
  - *afirmaţii despre servicii* pe care sistemul trebuie să le conţină, cum trebuie el să răspundă la anumite intrări şi cum să reacţioneze în anumite situaţii.

- **Cerinţe non-funcţionale**:
  - *constrângeri ale serviciilor* și funcţiilor oferite de sistem cum ar fi: constrângeri de timp, constrângeri ale procesului de dezvoltare, standarde, etc.

Cerințe funcționale

- Descriu funcţionalitatea sistemului şi serviciile oferite
- Depind de tipul softului, de utilizatorii avuţi în vedere şi de tipul sistemului pe care softul este utilizat
- Cerinţele funcţionale ale utilizatorilor pot fi descrieri de ansamblu dar cerinţele funcţionale ale sistemului trebuie să descrie în detaliu serviciile oferite

Cerințe non-funcționale

- Definesc *proprietăţi şi constrângeri ale sistemului.*
  D.ex.: fiabilitatea, timpul de răspuns, cerinţele pentru spaţiul de stocare, cerinţe ale sistemului de intrări-ieşiri etc.

- La întocmirea lor se va ţine cont de un anumit mediu de dezvoltare, limbaj de programare sau metodă de dezvoltare

- Cerinţele non-funcţionale pot fi mai importante decât cele funcţionale. Dacă nu sunt îndeplinite, sistemul nu va fi util scopului în care a fost dezvoltat.

Tipuri de cerințe non-funcționale

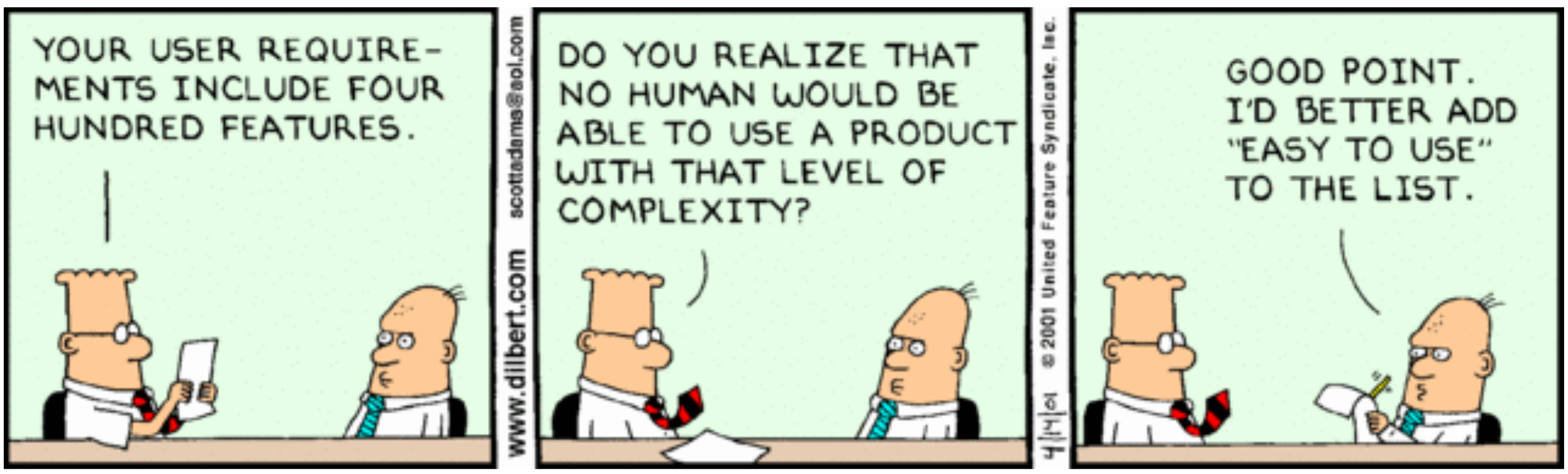
- Cerinţe ale **produsului**
  - Cerinţe care specifică un anumit comportament al produsului. D. ex.: gradul de utilitate, eficiență (viteză de execuţie), fiabilitate, portabilitate etc.
- Cerinţe legate de **organizare**
  - Cerinţe care sunt consecinţe ale politicilor de organizare a producţiei software. D. ex.: standarde utilizate, cerinţe de implementare, cerințe de livrare, securitate etc.
- Cerinţe **externe**
  - Cerinţe asociate unor factori externi. D. ex.: cerinţe de interoperabilitate, cerinţe legislative etc.

Tipuri de cerințe non-funcționale

Proprietate	Exprimare (exemple)
Viteză	Tranzacţii/sec; Timp de răspuns la utilizator
Dimensiune	MB; Nr. de cipuri necesare
Ușurința utilizării	Timp de învățare; dimensiunea “help”-ului
Fiabilitate	Timpul mediu dintre două defecte; Rata de apariţie a defectelor
Robusteţe	Probabilitatea de corupere a datelor la eroare; Timpul de restart după apariţia unui defect
Portabilitatea	Procentul de linii de cod dependente de ţinta implementării; Numărul de sisteme ţintă

Cerințe ale utilizatorilor

- Cerinţele utilizator trebuie să descrie cerinţe funcţionale şi non-funcţionale pe înţelesul utilizatorilor sistemului care nu deţin cunoştinţe tehnice detaliate.



de la dilbert.com

Recomandari pentru scrierea cerințelor

- adoptarea unui **format standard** şi utilizarea lui la toate cerinţele

- utilizarea limbajului într-un format consistent. E bine să se folosească **trebuie** pentru cerinţele obligatorii şi **ar trebui** pentru cele opționale

- **evidenţierea textului** care identifică părţile importante ale cerinţelor

- evitarea jargonului

- dacă este cazul, includerea unei explicații de ce este necesară o anumită cerință

Cerințele sistemului

- cerinţele sistemului sunt specificate mai detaliat decât cerinţele utilizator

- scopul principal al lor este acela de a fi baza proiectării sistemului

- cerinţele trebuie să exprime **ce poate face** sistemul, iar proiectul trebuie să exprime **cum se poate implementa** sistemul

- ele pot fi incorporate în contract

Structura document de specificare a cerinţelor

- Prefaţă
- Introducere
- Glosar de termeni
- Definirea cerinţelor utilizatorilor
- Arhitectura sistemului
- Specificarea cerinţelor de sistem
- Modelarea sistemului
- Evoluţia sistemului
- Anexe
- Index

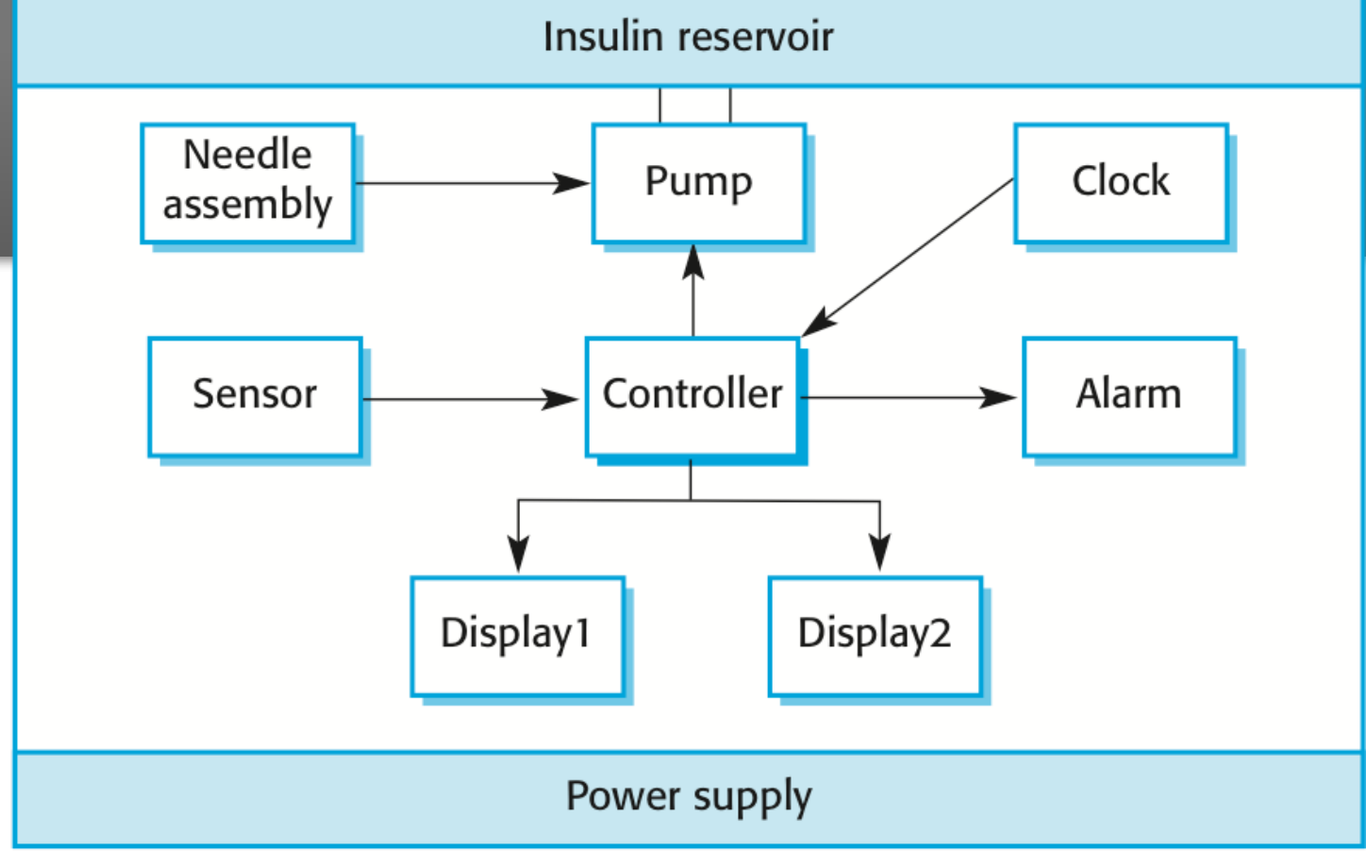
Acesta este doar un exemplu de structura

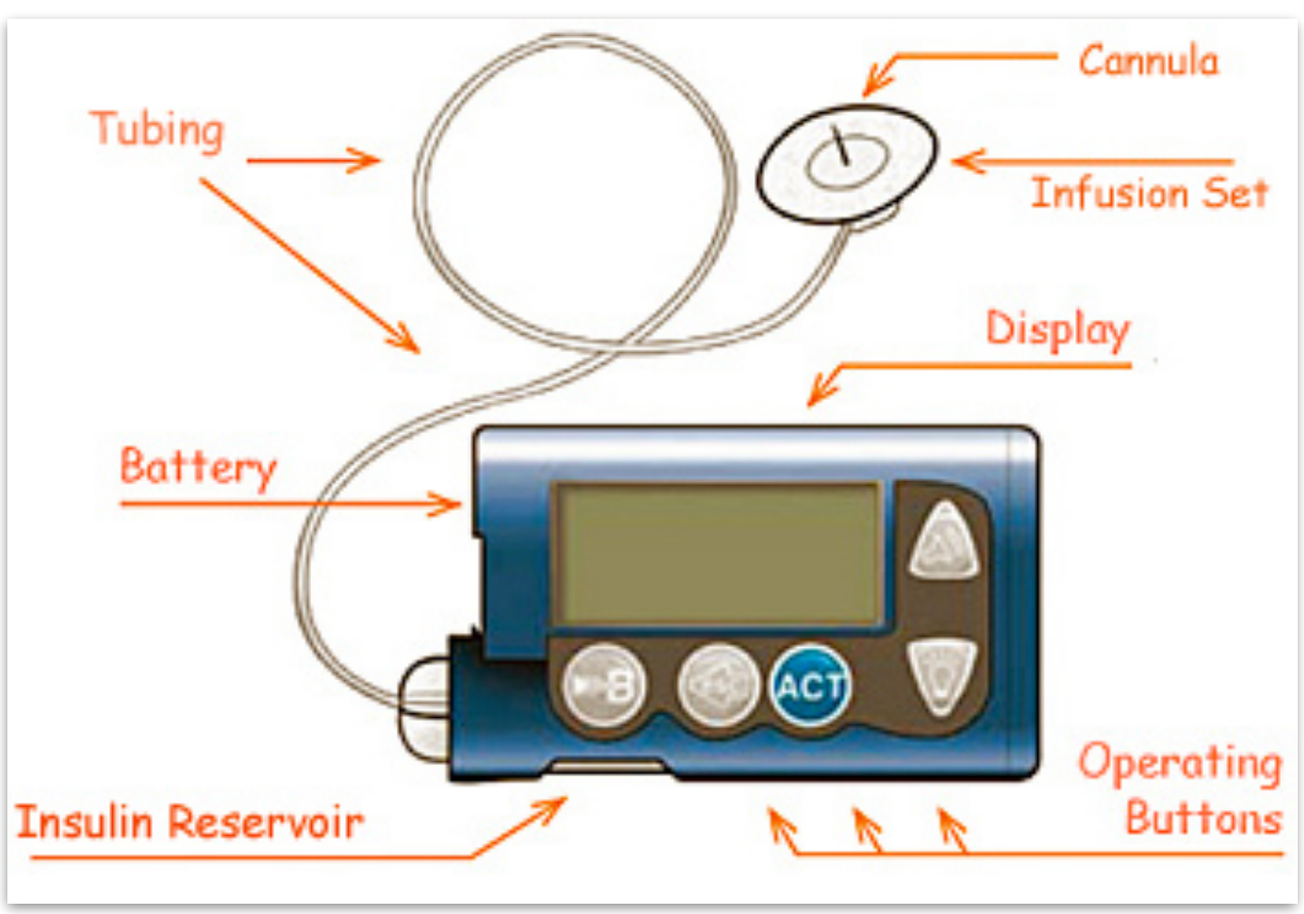
Posibilități de reprezentare a cerințelor

- **limbaj natural**: Cerințele sunt organizate în paragrafe numerotate.
- **limbaj natural structurat**: utilizarea unui format standard sau a machetelor în conjuncţie cu limbajul natural
- **limbaj grafic** suplimentat cu adnotări textuale (mai ales pentru cerinţe sistem), cum ar fi UML.
- **specificaţii matematice**: concepte matematice lucrând cu maşini cu stări finite sau relații peste mulțimi (limbajul Z). Elimină ambiguităţile, dar pot fi dificil de înţeles.

Exemplu - pompă de insulină

- Colectează date de la senzorul pentru glicemie (zahărul din sânge) și calculează cantitatea de insulină care trebuie injectată
- Calculul se bazează pe diferențele glicemiei în timp.
- Trimite semnale cu valoarea dozei de insulină unei micropompe.
- Funcționarea corectă este critică pentru sănătatea pacientului.





de la lenny-diabetes.com

Exemple de cerințe în limbaj natural

- Cerința 3.2: Sistemul măsoară glicemia la fiecare 10 minute și, dacă e nevoie, pompează insulină.

  (*Explicație: Modificarea glicemiei e relativ lentă, astfel că măsurători mai frecvente nu sunt necesare. Pe de altă parte, măsurători prea rare pot rata valori periculoase ale glicemiei.*)

- Cerința 3.6: Sistemul va rula un test intern o dată pe minut pentru a verifica funcționarea corectă a pompei, conform tabelului 1.

  (*Explicație: Un test intern poate detecta probleme hardware sau software, alertând utilizatorul în cazul unor disfuncționalități.*)

Exemplu de cerință structurată

Insulin Pump / Control Software / SRS / 3.3.2

Function: Calculează doza de insulină pentru a menține glicemia în limite normale.

Description: Calculează doza de insulină care trebuie injectată pentru a menține glicemia în limitele normale de 3-7 unități.

Inputs: Valoarea curentă (*r2*); două valori anterioare (*r0* și *r1*).

Source: Valoarea citită de senzor. Celelalte valori din memorie.

Outputs: *CompDose* - doza de insulină care trebuie injectată.

Destination: Ciclul de control al pompei.

Exemplu de cerință structurată continuat

Insulin Pump / Control Software / SRS / 3.3.2 (continuare)

Action: *CompDose* este zero dacă glicemia e stabilă sau scade, sau dacă nivelul ei crește, dar rata de creștere e în scădere. Dacă nivelul crește și rata de creștere de asemenea crește, atunci *CompDose* se calculează împărțind la 4 diferența dintre *r2* și *r1* și rotunjind rezultatul. Dacă rezultatul rotunjit este zero, atunci *CompDose* ia valoarea predefinită a dozei minime.

Requirements: Două valori anterioare pentru a putea calcula rata de schimbare a glicemiei.

Pre-condition: Rezervorul de insulină conține cel puțin valoarea maximă permisă a unei doze de insulină.

Post-condition: *r0* este înlocuit cu *r1* și *r1* cu *r2*.

Side effects: Niciunul.

Specificație tabelară (completează limbajul natural)

Condiție	Acțiune
Nivelul glicemiei scade: (*r2* < *r1*)	*CompDose* := 0
Nivelul glicemiei stabil: (*r2* = *r1*)	*CompDose* := 0
Nivelul glicemiei crește și rata de creștere descrește: 0 < (*r2*-*r1*) < (*r1*-*r0*)	*CompDose* := 0
Nivelul glicemiei crește și rata de creștere e stabilă sau crește: (*r2*-*r1*) ≥ (*r1*-*r0*) > 0	*CompDose* := *round* ((*r2*-*r1*)/*4*) Dacă rezultatul de mai sus e 0, *CompDose* := *MinimumDose*