Kehittää
ohjelmointitehtävien
ratkaisemisessa
tarvittavia
metakognitioita!

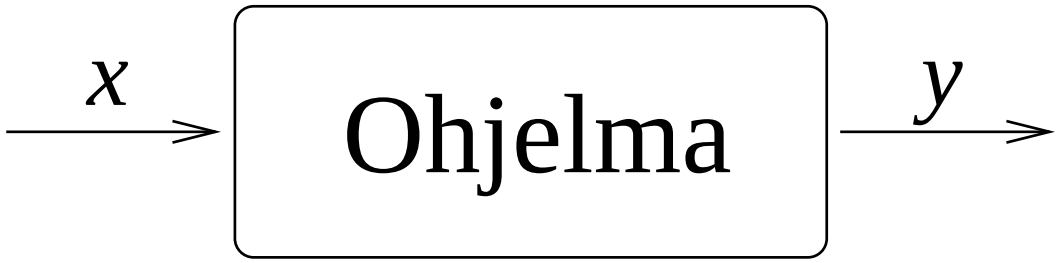
Cleanroom

- funktionaalinen laatikkomalli ohjelmistokehitykseen
- oikein käytettynä tuottaa lähes virheetöntä softaa (vrt. miksi pitikään olla formaali?)
- *cleanroom* $\sim$ puhdas tila puolijohteiden valmistamiseksi
- isänä Harlan Mills, oppi-isinä Dijkstra (rakenteinen ohjelmointi, Wirth (asteittainen tarkentaminen) ja Parnas (modulaarisuus)
- peruslähtökohdat:

  1) ohjelmat ovat sääntöjä matemaattisille funktioille (vrt. aksiomaattinen formalismi ja funktionaalinen ohjelmointi),

  2) ohjelman (suorituksen) oikeellisuutta ja siten ohjelman laatua voidaan testata tilastollisen otantateorian avulla.

- funktionaalisuus: $f : X \to Y$ $\sim$ deterministinen ohjelma input-output -parien välillä



hyvä ohjelma $\sim$ hyvin määritelty funktio eli

täydellinen (*complete*): kaikki määrittelyjoukon alkiot kuvautuvat ainakin yhdelle tulojoukon alkiolle

johdonmukainen (consistent): jokainen määrittelyjoukon alkio kuvautuu korkeintaan yhdelle tulojoukon alkiolle

oikea (*correct*): spesifikaation validointi enemmän tai vähemmän formaalisti

Cleanroom-tiimi

- kolme päätehtävää
  1. järjestelmän spesifiointi
  2. järjestelmän kehittäminen
  3. järjestelmän sertifiointi
- 3–8 henkilöä, joista yksi (muodollinen) johtaja
- laajoissa projekteissa useita tiimejä eri vaiheissa
- jokainen tiimin jäsen tietää, mitä kokonaisuudessaan tapahtuu
- toiminta perustuu tiimin yhteisille *katselmuksille*:
  - *kehityskatselmukset:* mahd. yksinkertaisen toteutuksen idea ja tekninen toteutus
  - *verifiointikatselmukset:* osajärjestelmän oikeellisuus ja täydellisyys assertioiden avulla; vaihetuote joko valmis tai muutoksin uudelle katselmuskierrokselle
- HUOM: mihin tarvittiin etuliitettä *Cleanroom*?

Cleanroom-perusprosessi

- peruspilarina ohjelmistoarkkitehtuuri: tuotelinja- & referenssiarkkitehtuuri, uusi järjestelmä, olemassaolevan järjestelmän uudelleen toteutus, . . . (kuten muissakin malleissa)
- inkrementaalisuus perustuu toteutusjärjestykseen: kehitysprosessien synkronointi, GUI-prototyypit, funktionaaliset riippuvuudet, riskit, monimutkaisuus, uutuus, uudelleenkäyttö ja käytön yleisyys (kuten muissakin malleissa)
- tilastollinen ja tiimin tavoitelähtöinen kontrolli

Cleanroom-**referenssimalli**

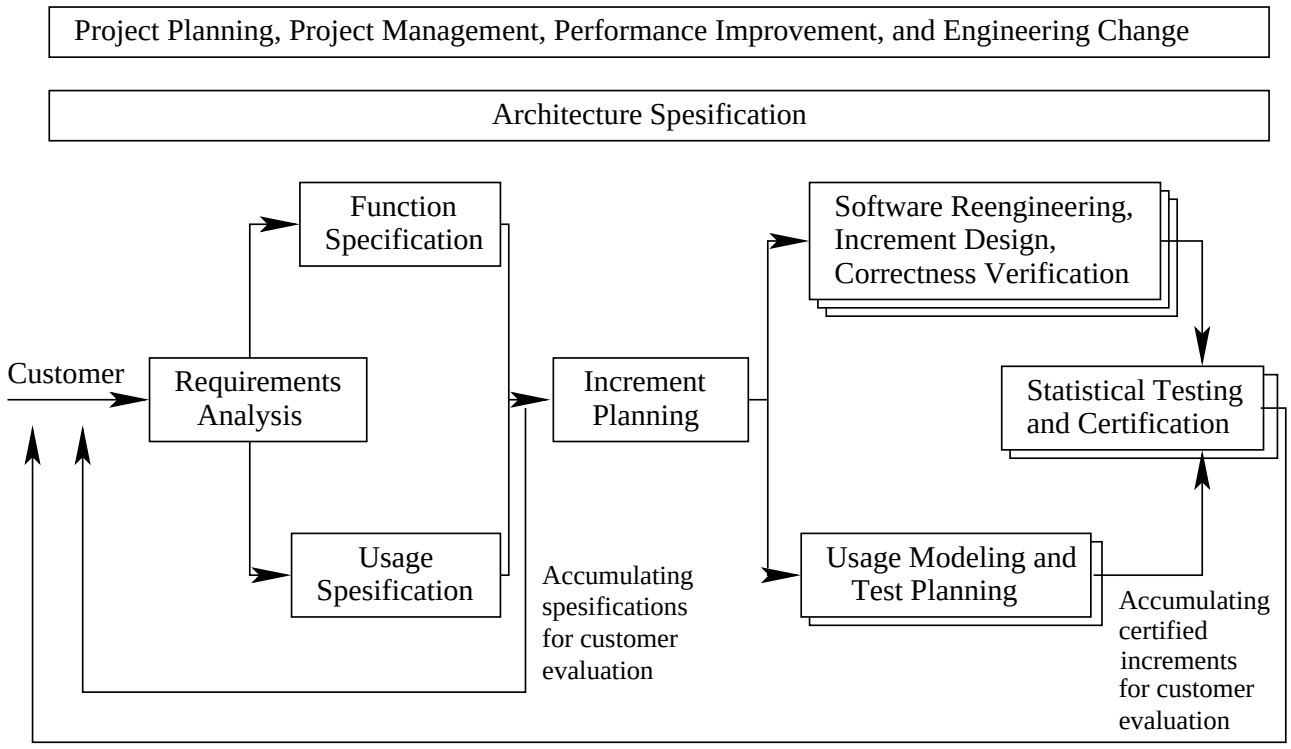
- määrittää integroidut työprosessit ja vaihetuotteet, mm.

  Management: *Project Planning, Project Management, Performance Improvement, Engineering Change*

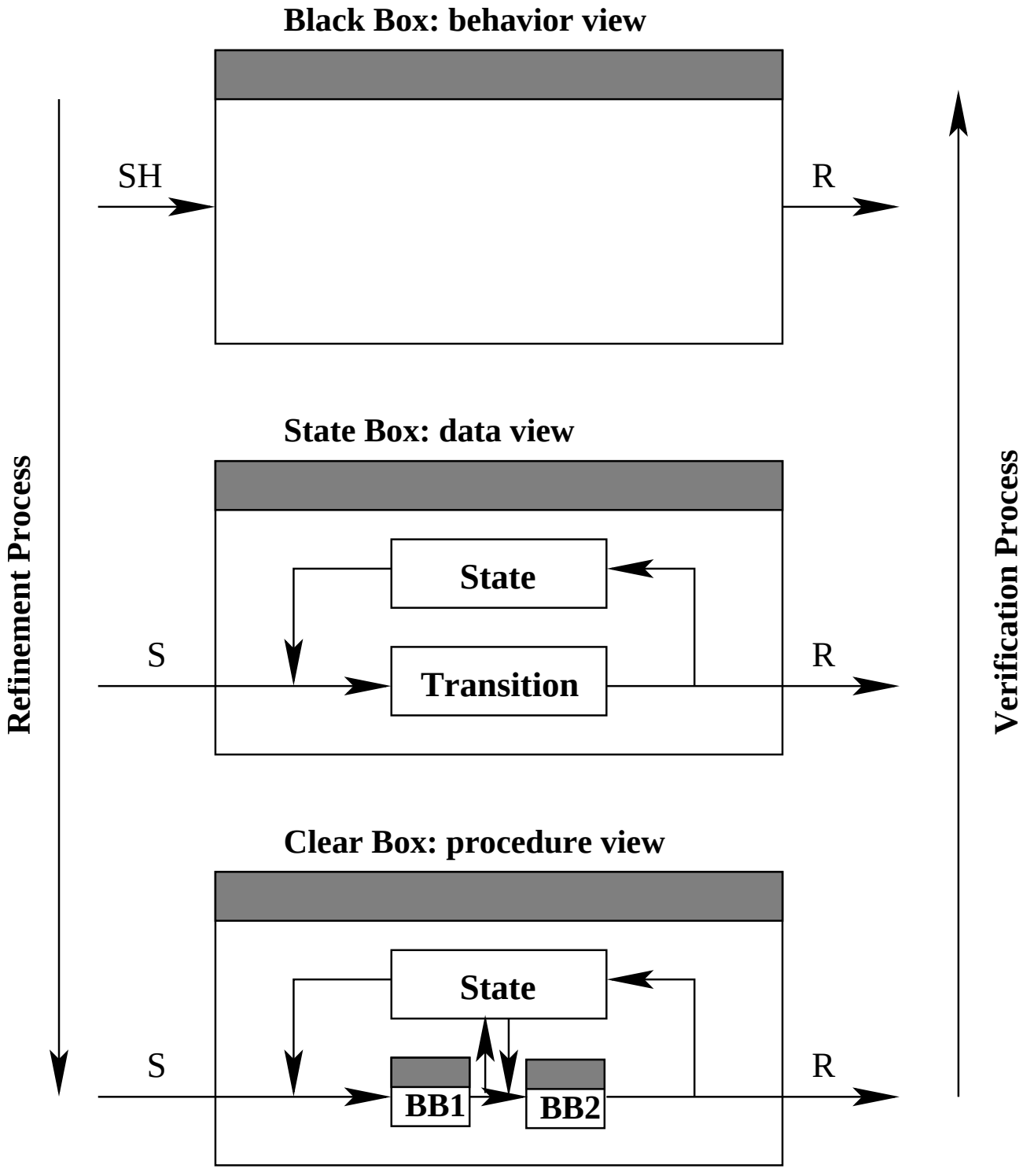
  Specification: *Requirements Analysis, Functional Spesification, Usage Spesification, Architecture Spesification, Increment Planning*

  Development: *Software Reengineering, Incremental Design, Correctness Verification*

  Certification: *Usage Modeling and Test Planning, Statistical Testing and Certification*

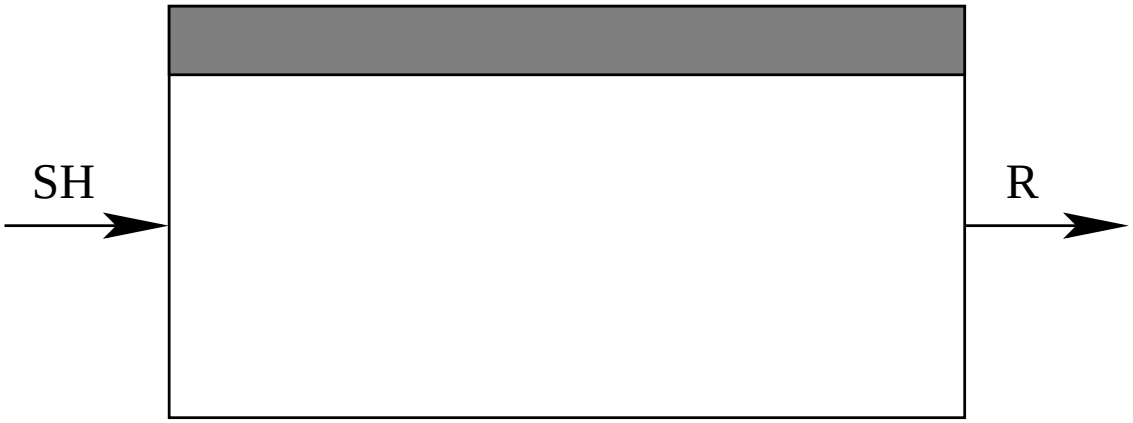


Cleanroom-spesifikaatio



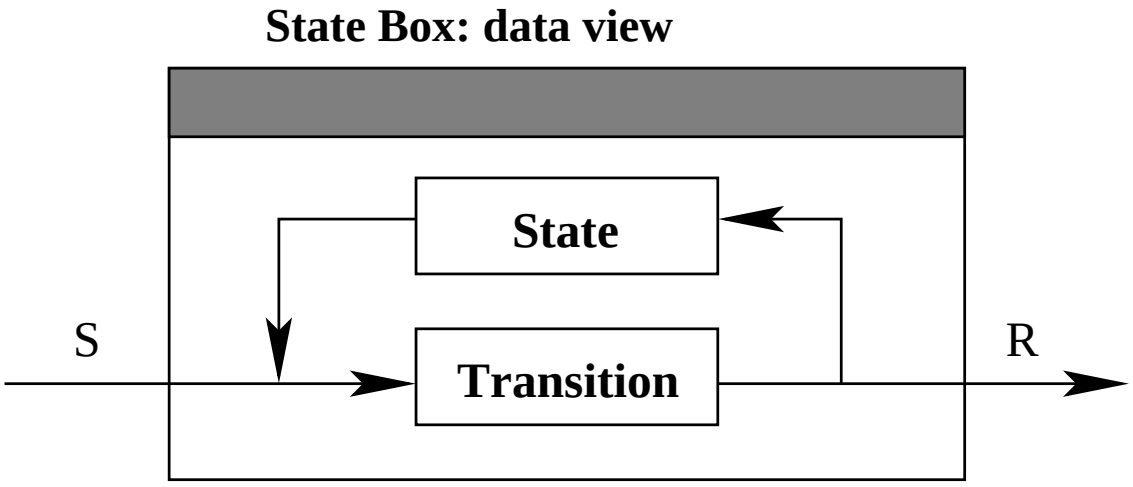
Cleanroom-spesifikaatio: BB

Black Box: behavior view



- käyttäjän (henkilö, muu järjestelmä, laite, yms.) järjestelmältä edellyttämä ulkoinen käyttäytyminen syötehistorian (Stimulus History, SH) ja sitä vastaavan outputin (Responssi, R) suhteen
- koostuu herätelistasta, vastelistasta ja koko SH:sta
- tilaton ja rakenteeton funktiokuvaus, jonka tulee olla täydellinen, johdonmukainen ja jäljitettävästi oikea
- käyttäjälle BB:t määrittävät halutun käyttäytymisen
- kehittäjälle BB:t määrittävät suunniteltavan ja toteutettavan käyttäytymisen
- testaajalle BB:t määrittävät validoitavan käyttäytymisen
- vrt. (yleistetyt) käyttötapaukset ja niiden aktorit

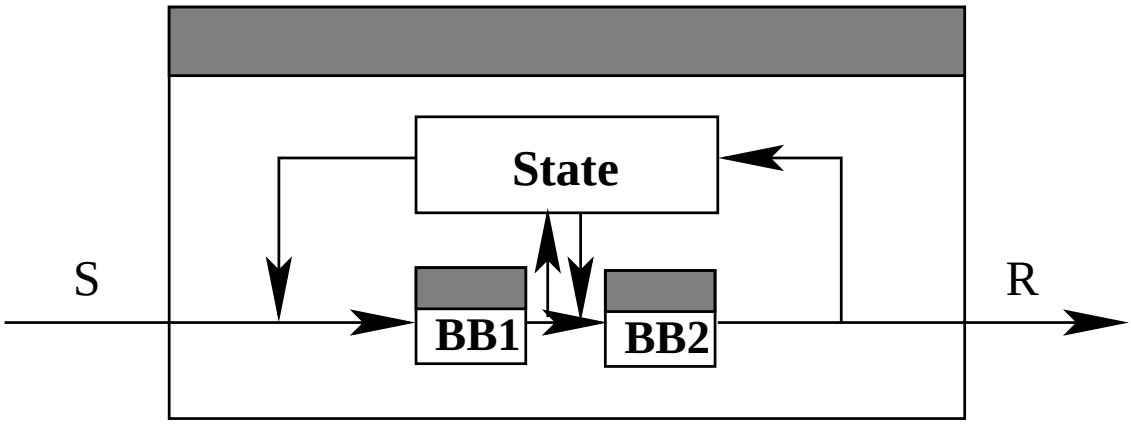
Cleanroom-spesifikaatio: SB



- BB:stä johdettu ensimmäinen tarkennusaskel kohti implementaatiota
- kuvaa tilansiirron nykyiseltä tilalta ja ärsykkeeltä (tilansiirron laukeaminen) uudelle tilalle
- perustuu tilamuuttujien identifiointiin
- (funktionaalisesti) täydellinen, johdonmukainen ja jäljitettävästi oikea

Cleanroom-spesifikaatio: CB

Clear Box: procedure view



- algoritminen implementaatio SB:n toteuttamiseksi
- voi sisältää uusia BB:itä osaratkaisujen mallintamiseksi (*top-down*)
- kontrollirakenteina (kaikenkattavat) haarautumiset, iteraatiot ja rinnakkaisuus
- (funktionaalisesti) täydellinen, johdonmukainen ja jäljitettävästi oikea - tottakai
- voidaan esittää joko suunnittelunotaation tai toteutuskielen avulla

Laatikkomallin perusteet

- verifiointi ns. oikeellisuuskysymyksien avulla
- ohjelma: yhdistelmä kontrollirakenteita sekvenssi (**do**), haarautuminen (**if** ... **then** ... **else**) ja iteraatio (**while** ... **do**)
  $\Rightarrow$
  järjestelmä: yhdistelmä laatikkorakenteita BB, SB ja CB
- periaatteet
  1. viittauksellinen näkyvyys (*Referential Transparency*): järjestelmäkomponentin vaatimukset spesifioidaan täsmällisesti heti
  2. transaktioiden sulkeutuvuus (*Transaction Closure*): järjestelmän tai komponentin transaktioiden täydellinen esittäminen
  3. tilan "kulkeutuminen" (*State Migration*): muuttujat kulkeutuvat ja rajoittuvat mahdollisimman pieniin ja itsenäisiin komponentteihin (*encapsulation*)
  4. yhteiset palvelut (*Common Services*): usein tarvittavat komponentit määritellään yhteisiksi palveluiksi (vrt. abstraktit tietorakenteet ja STL)

Laatikkoprosessimalli

1. Määrittele vaatimukset
2. Spesifioi ja validoi BB
   - määritä järjestelmän rajat ja spesifioi kaikki ärsyke-vastin -parit
   - spesifioi BB kuvaussäännöt
   - validoi BB omistajien ja käyttäjien kanssa
3. Spesifioi ja verifioi SB
   - spesifioi tilamuuttujat ja niiden alkuarvot
   - spesifioi tilansiirtofunktio
   - johda SB:n BB käyttäytyminen ja vertaa tämän ja kohdan 2. BB:n ekvivalenttiutta
4. Suunnittele ja verifioi CB
   - suunnittele CB:n kontrollirakenteet ja operaatiot
   - ota käyttöön uusia ja uudelleen käytettäviä BB:itä tarpeellinen määrä
   - johda CB:n SB-käyttäytyminen ja vertaa tämän ja kohdan 3. SB:n ekvivalenttiutta

- vaiheiden käsittelyn intensiteetti projektityypin mukaan (painopiste ulkoisessa käyttäytymisessä (BB) $\vee$ kompleksisissa tietotyypeissä (SB) $\vee$ vaativissa algoritmeissa (CB)

Cleanroom **OT:n kentässä**

Cleanroom **vrs. oliot:** voidaan käyttää oliototeutuksen yhteydessä samaistuksilla

- olio $\sim$ tilakone
- BB $\sim$ olion ulkoinen käyttäytyminen (rajapinta)
- SB $\sim$ attribuutit eli olion tila
- CB $\sim$ metodit, joiden avulla tilaa muutetaan

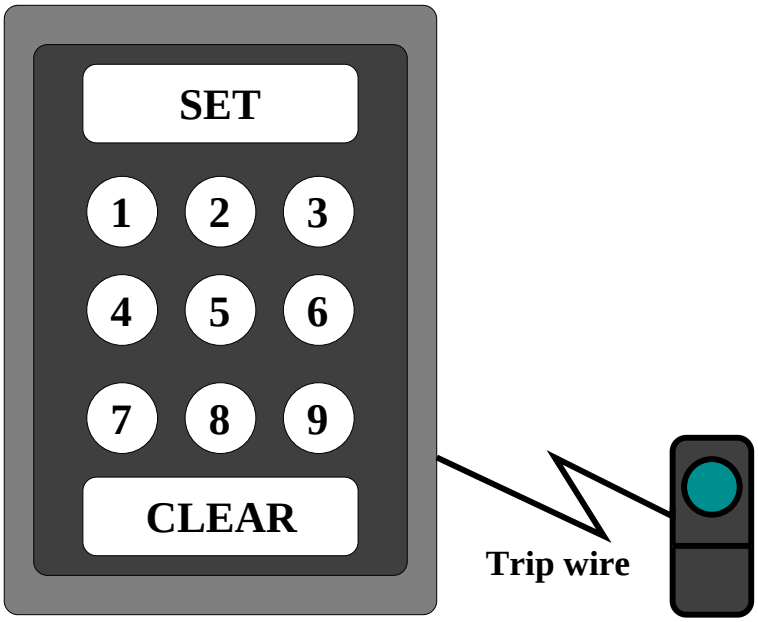
Uudelleenkäyttö: vaatii komponentin toiminnan täsmällisen määrityksen sekä tietämyksen toteutuksen laadusta ja luotettavuudesta tietyssä toimintaympäristössä

Arkkitehtuuri: laatikko-skeemojen käyttö helpottaa järjestelmän komponenttien ja niiden välisten kytkentöjen hahmottamista ($\sim$ arkkitehtuuri) ja muuttamista

Tarkastukset ja katselmukset: tulevat täsmällisemmiksi funktiteorian kautta, jossa lähtökohtana on, että ohjelma (koodi) implementoi funktion (spesifikaation); tiimin rooli

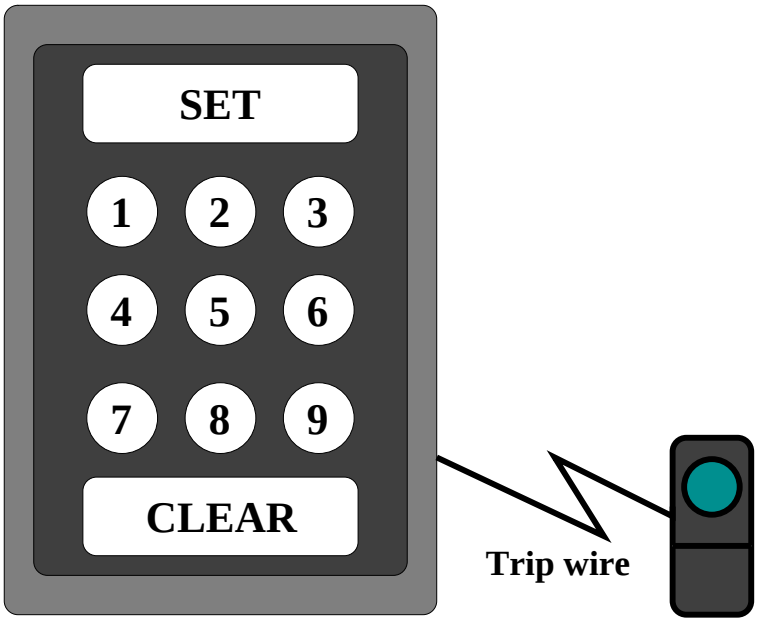
Testaus: käyttömalleja ja Box-skeemoja voidaan käyttää myös muiden testausmenetelmien kanssa (black box -testing, white box -testing, regressio, . . .)

Esimerkki: Turvahälytin



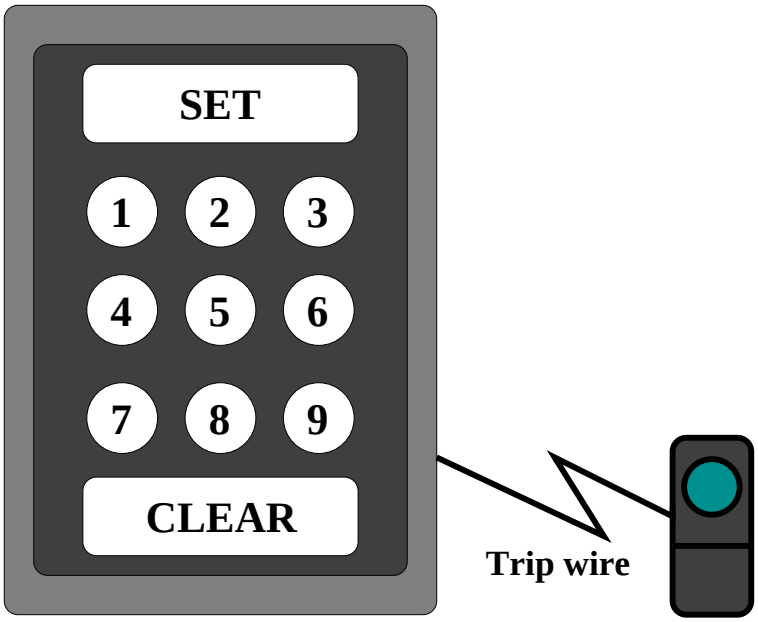
Program statement: Hälyttimessä on liikkeentunnistin, joka lähettää laitteelle signaalin liikettä havaitessaan. Hälytin aktivoidaan painamalla Set-nappia, joka on valaistu laitteen ollessa aktivoitu. Liikesignaalin vastaanottaminen laitteen ollessa aktivoituna laukaisee hälytysäänen, jonka lopettamiseksi täytyy laitteeseen näppäillä kolminumeroinen tunnuskoodi laitteen deaktivoimiseksi. Jos koodia näppäiltäessä tulee virhe, täytyy koodinantosekvenssi resetoida painamalla Clear-nappia. Hälyttimeen ei voi itse ohjelmoida tunnuskoodia, vaan jokaiseen laitteeseen liittyy siihen ennalta asetettu, muuttumaton koodi.

Toiminnalliset vaatimukset



Nr	Vaatimus
1	Hälyttimessä on liikkeentunnistin, joka lähettää laitteelle signaalin liikettä havaitessaan.
2	Hälytin aktivoidaan painamalla Set-nappia.
3	Set-nappi on valaistu laitteen ollessa aktivoitu.
4	Liikesignaalin vastaanottaminen laitteen ollessa aktivoituna laukaisee hälytysäänen.
5	Hälytysäänen lopettamiseksi laitteeseen täytyy näppäillä kolminumeroinen tunnuskoodi.
6	Oikean tunnuskoodin antaminen deaktivoi laitteen.
7	Jos koodia näppäiltäessä tulee virhe, täytyy koodinantosekvenssi resetoida painamalla Clear-nappia ennen uuden koodin antamista.

Heräte-vaste -rajapinnat



heräte	lyh.	kuvaus	vaatimusseuranta
Set	S	laitteen aktivointi	2
Trip	T	tunnistimen signaali	1
BadDigit	B	väärä koodinumero	7
Clear	C	koodin resetointi	7
GoodDigit	G	osa oikeaa koodisekvenssiä	5, 6

vaste	kuvaus	vaatimusseuranta
Light on	Set-näppäin valaistu	3
Light off	Set-näppäin ei valaistu	6
Alarm on	hälytysääni päällä	4
Alarm off	hälytysääni pois päältä	5

BB: SH & R

Taulukko A: Järjestelmän sekvenssit numeroituna

sekvenssi	vaste	ekviv.	vaatimusseuranta
Pituus = 0			
Empty	Null		D1: Laite on alunperin deaktivoitu.
Pituus = 1			
S	Light on		2, 3
T	Illegal		D1
B	Illegal		D1
C	Illegal		D1
G	Illegal		D1
Pituus = 2			
SS	Null	S	D2: Jo aktiivisen laitteen aktivoinnilla ei ole vaikutusta.
ST	Alarm on		4
SB	Null		D3: Aktiivinen laite ei reagoi väärään koodinumeroon.
SC	Null	S	D4: Aktiivinen laite ei reagoi koodinsyötön resetointiin.
SG	Null		D5: Aktiivinen laite ei reagoi oikeaan koodi-numeroon ennen kuin koko koodi on annettu.
Pituus = 3			
STS	Null	ST	D2
STT	Null	ST	D6: Hälytysäänen ollessa jo päällä tunnistimen signaalilla ei ole merkitystä ennen laitteen deaktiointia.
STB	Null		D3
STC	Null	ST	D4
STG	Null		D5
SBS	Null	SB	D2
SBT	Alarm on	STB	4
SBB	Null	SB	D3
SBC	Null	S	D4, 7
SBG	Illegal		7

Taulukko A: (jatkuu)

Pituus = 3 (jatkuu)			
SGS	Null	SG	D2
SGT	Alarm on	STB	4, D7: Oikean koodinumeron antamista ennen tunnistimen signaalia pidetään toimenpiteenä, joka vaatii Clear-näppäimen painamista ja oikean koodin antamista hälyttimen deaktivoimiseksi.
SGB	Null	SB	D3
SGC	Null	S	D4
SGG	Null		D5
Pituus = 4			
STBS	Null	STB	D2
STBT	Null	STB	D6
STBB	Null	STB	D3
STBC	Null	ST	D4, 7
STBG	Illegal		7
STGS	Null	STG	D2
STGT	Null	STG	D6
STGB	Null	STB	D3
STGC	Null	ST	D4
STGG	Null		D5
SGGS	Null	SGG	D2
SGGT	Null	STB	4, D7
SGGB	Null	SB	D3
SGGC	Null	S	D4
SGGG	Light off	Empty	6
Pituus = 5			
STGGS	Null	STGG	D2
STGGT	Null	STGG	D6
STGGB	Null	STB	D3
STGGC	Null	ST	D4
STGGG	Alarm off, light off	Empty	3, 5, 6

BB:n validointi

- jokaiselle herätteelle on määrätty vaste (täydellisyys)
- jokaiselle herätteelle on määrätty yksikäsitteinen vaste (johdonmukaisuus)
- johdettujen vaatimusten oikeellisuus tarkastetaan yhdessä vaatimusmäärittelijän kanssa

BB ⟶ SB

Taulukko B: Kanonisten sekvenssien analyysi

kanoninen sekvenssi	**tilamuuttujat**	**arvo ennen herätettä**	**arvo jälkeen herätteen**
Empty	—	—	—
S Käyttäjä on aktivoinut laitteen painamalla Set-näppäintä.	Device	OFF	ON
ST Laite on päällä ja liikesignaali käynnistää hälytyksen	Device Alarm	ON OFF	ON ON
SB Laite on päällä ja käyttäjä antaa väärän koodinumeron.	Device Code	ON NONE	ON ERROR
SG Laite on päällä ja käyttäjä on antanut ensimmäisen oikean koodinumeron.	Device Code	ON NONE	ON 1_OK
STB Laite on päällä, liikesignaali on käynnistänyt hälytyksen ja käyttäjä on antanut väärän koodinumeron. Koodin antaminen täytyy resetoida painamalla Clear-nappia ennen kuin hälytys voidaan saada pois päältä.	Device Alarm Code	ON ON NONE	ON ON ERROR
STG Laite on päällä, liikesignaali on käynnistänyt hälytyksen ja käyttäjä on antanut ensimmmäisen oikean koodinumeron.	Device Alarm Code	ON ON NONE	ON ON 1_OK
SGG Laite on päällä ja käyttäjä on antanut kaksi oikeaa koodinumeroa.	Device Code	ON 1_OK	ON 2_OK
STGG Laite on päällä, liikesignaali on käynnistänyt hälytyksen ja käyttäjä on antanut kaksi oikeaa koodinumeroa.	Device Alarm Code	ON ON 1_OK	ON ON 2_OK