Checklister

Appendiks til bogen ’Struktureret Test’

1. Indledning

Checklister bruges til at støtte reviews. Forberedelsen bliver nemmere, og gennemførelsen bliver mere systematisk. Spørgsmålene kan enten bruges til at gå slavisk frem efter, eller de kan bruges til at summere op på et review.

Ikke alle spørgsmål behøver at være relevante for et givent review. Vælg de spørgsmål ud der skal besvares. Dokumenter alle benægtende svar i reviewrapporten, sammen med årsagen. Dokumenter de spørgsmål der ikke kan besvares, sammen med årsagen.

Checklister er en del af et kvalitetsstyringssystem. De indgår dermed i det præventive 0-fejl arbejde. Man kan nøjes med at bruge dem til simpel efterkontrol af færdige produkter. Men jo mere aktivt kvaliteten bliver styret, jo mere bliver checklisterne en del af sikringen snarere end kontrol. Når kvalitetsstyringen er fuldt udbygget, f.eks. efter ISO 9000 standarden, bliver efterkontrollen mindre, og checklisterne vil kun indeholde de nyeste erfaringer og risici. Det øvrige er indbygget i procedurer.

Listerne vil stadig være nyttige. Det er da en fornøjelse at checke at det hele fungerer, og tiden til at hakke en checkliste af, er givet godt ud. Det er ikke de første timer man skal koncentrere sig om at spare i hver udviklingsfase.

Reviews er et godt middel til erfarings- og vidensprednin. Checklister øger yderligere den nytte.

En checkliste skal ejes af en person eller afdeling i virksomheden. Selv når checklisterne er en del af et kvalitetsstyringssystem, skal ansvaret for hver listes udbygning og vedligeholdelse være fuldt defineret. Uajourførte checklister er døende checklister.

Checklister dokumenterer vedtagne standarder, og er på den måde indiskutable. Når spørgsmålet er sat på listen, skal det tages alvorligt. De er en hjælp, således at det der engang er diskuteret, ikke skal trævles igennem igen.

Det betyder ikke at checklisterne er statiske. De skal holdes levende. Hvis der findes en fejl på et senere tidspunkt end ønskeligt, skal der defineres et checkspørgsmål, der fanger den tidligere. Det sker som seriøs videreudvikling, foretaget af den ansvarlige for checklisten.

Spørgsmålenes relevans skal også vurderes, og spørgsmål udgår om nødvendigt. Men brug ikke tiden på de konkrete reviews til det. Nøjes med at markere at der er brug for at vurdere det senere.

Spring ikke en checkliste over, fordi den virker bekendt eller overflødig. Selv den mest erfarne pilot checker tingene før start og landing. Selvom det er en triviel procedure, hvor der gang efter gang udføres de samme handlinger, og checkes de samme ting. Husk at udbyttet nogle gange ligger i, at alt er i skønneste orden.

Checklister er mest synlige på inspektioner. Men ethvert review kan summeres op ved hjælp af dem.

2. Checkliste til review af foranalysen

Foranalysen er udviklernes grundlag for at beskrive det rette system. Foranalysen beskriver den ønskede forretningsmæssige forandring. Et review af foranalysen er en mulighed for at skaffe et komplet, korrekt og stabilt grundlag. Resultatet af foranalysen er en kravspecifikation, med eller uden Use Cases.

Spørgsmål:

1. Er kravspecifikationen komplet således at alle krav er til stede
- den skal kunne skrives under som en kontrakt

2. Er hvert krav kvantificeret i form af et eksakt målbart mål
- det er ikke nok at kræve et hurtigt system, der skal specificeres målbare tider og volumener

3. Er hvert krav operationelt, således at det er let at måle
- det er ikke nok at det kun kan måles på nordpolen hvert andet år

4. Er hvert krav entydigt
- opfatter alle interessenter det på samme måde

5. Er hvert krav kommunikerbart
- en matematisk formel kan være præcis, men ikke nødvendigvis det bedste til formidling

6. Er hvert krav sporbart
- kan man se, hvem der har stillet kravet

7. Er kravene set under eet 'ikke-overlappende'
- specifikke uafhængige krav er bedre end brede formuleringer med potentielt overlap

8. Er kravene set under eet 'ikke modstridende'
- visse modstridende krav kan ikke undgås, fx på områder hvor performance kolliderer med andre systemegenskaber, men de modstridende punkter skal være dokumenterede

9. Er kravene indbyrdes beskrevet på et ensartet og tilstrækkelig detaljeret niveau
- en stor forskel skal motiveres

10. Er kravene vurderet med hensyn til deres økonomiske vægt
- evt. sværvægtere skal være synlige

11. Er der taget stilling til systemets levetid
- en levetid over fem år stiller større krav til vedligeholdelse, udbygning og flytbarhed

12. Indeholder foranalysen en godkendt Cost/Benefit beregning
- og kan den eftervises og efterkalkuleres

13. Er kravene stemt af med de nuværende og kommende forretningsgange på brugsstederne
- specielt hvis der stilles krav på andres vegne, således at det ikke er de egentlige brugere, der er ophav

14. Betyder kravene en kraftig omlægning af forretningsgange
- hvis omlægningen ikke er identificeret, kan det skabe problemer for testen senere

15. Har alle interessenter været involveret i at stille krav og involveret i at godkende dem
- en interessentanalyse kan hjælpe til med at besvare spørgsmålet

16. Er hvert krav godkendt specifikt af brugere/betalere
- en underskrift er minimum

17. Er det enkelte krav testbart senere i forløbet
- hvis det er brugere der skal teste, skal spørgsmålet rettes til dem

18. Dokumenterer kravspecifikationen kriterierne for at godkende ibrugtagningen
- eller i det mindste hvornår de defineres, hvis det er på et senere tidspunkt

19. Er alle krav relevante
- en kravspecifikation må ikke blive for 'fed' af ligegyldige krav

20. Kan analysen gennemføres på foranalysens resultat
- dem der skal gennemføre analysen skal besvare spørgsmålet, og med kravspecifikationen som det eneste grundlag

21. Er der skelnet mellem 'nødvendige' og 'ønskværdige' krav
- i modsat fald skal det slås fast at alt er 'nødvendigt'

22. Er kravene til integration og samspil med andre systemer beskrevet
- på en måde der er målbar og operationel

23. Er der uafklarede forhold, og fremgår de i så fald tydeligt
- dette spørgsmål skal besvares højt af alle tilstedeværende

24. Er der stillet konkrete krav til nøjagtigheden af systemets data og beregninger
- man kan godt nøjes med at specificere et generelt krav, men spørgsmålet må gerne stilles, for at høre om der er tunge sager på tapetet.

25. Er der stillet krav til systemets robusthed
- især hvis nogle interessenter forventer 100 % robusthed, skal det fremgå

26. Er der stillet krav til ensartetheden af systemet
- helst med henvisning til en standard

27. Er der stillet krav til brugervenlighed
- igen kan en standard bruges som henvisning, suppleret med de krav der udspringer af brugernes og brugsstedets forhold

28. Er der stillet krav til testbarheden
- det er nødvendigt at vide hvor megen tid der er til rådighed til test, og hvilke typer af testere der skal kunne gøre det

29. Er der stillet krav om forvaltningsegnetheden
- gerne som rammer for en egentlig forvaltningsaftale, selv på dette tidlige tidspunkt

30. Er der stillet krav til graden af genbrug
- evt. hensigtserklæringer kan ikke bruges, der er brug for at se kravene på tryk

31. Er kravene til ydeevne synlige og omkostningsberegnede
- og især at det fremgår om kravene er komplette

32. Er der stillet krav til sikkerhed omkring dataintegritet, dataopbevaring, datatransport, autorisation etc.
- de må ikke udskydes til analysen

33. Er der stillet krav om systemets flytbarhed
- hvis flytbarhed kun er 'ønskværdig', skal det fremgå

34. Er der driftsmæssige krav der skal være afklaret på dette tidspunkt, og som ikke er med i kravene
- især hvis de skal hentes hos forskellige interessenter

35. Er der stillet krav om fall-back procedurers omfang og ambition
- det kan være en del af mange andre krav, fx sikkerheds- eller driftsmæssige, men det skal checkes at det er tilfældet

36. Er der specificeret en procedure for behandling af ændringsønsker
- det er altid rart at få sin 'Change Control Procedure' (CCP) godkendt, før den er nødvendig at bruge

37. Er der en vurdering af hvor mange ændringsønsker der vil blive fremsat
- selvom det er umuligt at kvantificere, kan diskussionen være nyttig

38. Er den nødvendige forretningsmæssige viden til stede
- er brugerne kompetente i forhold til kravenes ambitioner

39. Er den nødvendige IT-faglige viden til stede
- er IT-siden kompetent i forhold til kravenes ambitioner

40. Er estimaterne på udviklingstid og afleveringsfrister godkendte og realistiske
- der kan godt være afvigende meninger, men det er jo det, man er samlet for, så man kan få det på bordet

41. Er der taget stilling til strategier for udvikling, test og implementering
- de er ofte underforståede, men derfor kan der godt være uoverensstemmelser

42. Er der skrevet en testspecifikation
- eller i hvert fald taget stilling til om og hvornår det skal gøres

43. Er der taget stilling til hvem der skal udarbejde testdata til hver testfase
- det drejer sig især om dem der skal udarbejdes uden for projektgruppen, dvs. til bruger- og accepttest

44. Er estimering foretaget ved hjælp af en formel og anerkendt teknik
- det kan være for eksempel være Function Points

45. Er der taget stilling til hvilke typer af forretningsmæssige risici, der bliver aktuelle
- og udtrykt i brugersprog af brugere

46. Er der taget stilling til hvilke risici der kan opstå for udviklingsforløbets succes
- her kan erfaringen og eksempler hjælpe diskussionen på gled

47. Ud over de allerede stillede spørgsmål kan der så svares ja til at alle standarder og retningslinier for foranalysen er overholdte
- det gælder også evt. vedtagne ISO-, AQAP- og IEEE-standarder.

48. Kan kravspecifikationen erklæres for fejlfri
- i modsat fald skal svarene dokumenteres i reviewrapporten

49. Kan reviewdeltagerne underskrive reviewrapporten
- i modsat fald skal svarene dokumenteres i reviewrapporten

3. Checkliste til review af analysefasen

Review af analysen gennemføres af opgavestillere og testere. Det skal afgøre om der er beskrevet en løsning der helt og fuldt opfylder de stillede krav, inden for det aftalte budget.

Spørgsmål:

1. Kan det klart konstateres, at der er funktioner, der opfylder hvert enkelt krav, der er specificeret i kravspecifikationen
- det kan være svært at svare ja til, men det vil være en test i sig selv

2. Kan det klart konstateres, at der er data, der opfylder hvert enkelt krav, der er specificeret i kravspecifikationen
- det kan være lige så svært at svare på, men det er et endnu vigtigere spørgsmål, hvis målet er 0-fejl

3. Er der funktioner i systemet, der skal sørge for at imødegå de forretningsmæssige risici ved at bruge systemet
- der bør være funktioner, der forhindrer at de opstår, og funktioner der rapporterer noget unormalt

4. Passer de definerede funktioner med de daglige forretningsgange
- ellers skal der aktiviteter på planen, der ændrer forretningsgange

5. Passer funktioner i deres afgrænsning til autorisationen af brugerne
- måske skal autorisationsniveauerne ændres, men der kan også være skudt galt på funktionernes indhold og afgrænsning

6. Passer de data, der bruges i en funktion, med autorisationen af brugerne

- stil spørgsmålet, uanset svaret på det foregående

7. Passer de definerede funktioner med brugernes uddannelse
- ellers skal der uddannelsesaktiviteter på planen før implementering

8. Kræver de definerede funktioner ændringer i andre systemer
- og er de dokumenteret i en oversigtlig form

9. Er alle data, der bruges, dokumenterede
- helst i et Repository

10. Dokumenterer analysen at alle data er til rådighed, på det sted og på det tidspunkt i organisationen hvor de skal bruges
- det er en forudsætning for at systemet ikke senere skal blive opfattet som tungt at arbejde med

11. Omfatter analysen en dokumentation af datas kvalitet og pålidelighed i brugerorganisationen
- det har betydning for den grad af robusthed der skal være en del af funktionerne

12. Genererer systemet selv data, og er de en del af analysens dokumentation
- de skal også dokumenteres i en Repository

13. Er alle data undersøgt for redundans
- ellers kan man være sikker på at mængden af redundante data øges

14. Lever analysen op til kravet om svartider, datamængder etc.
- spørgsmålet stilles traditionelt først efter designfasen, men der kan det være for kostbart at ændre systemet

15. Er der afhængigheder til andre systemer udover dem der var en del af kravene
- der skal argumenteres for hver afhængighed

16. Er de driftsmæssige behov analyseret
- ellers skal der aktiviteter på planen, der gør det

17. Er behovet for nye/ændrede driftsprocedurer identificeret
- med distribuerede arkitekturer bliver denne del af analysen ikke mindre

18. Er behovet for den fremtidige support identificeret
- før det er gjort kan der ikke beregnes en cost/benefit

19. Har analysen dokumenteret behovet for et revisionsspor, også kaldet 'audit trail', og andre revisionsmæssige behov
- denne del af analysen kan være udført sideløbende, separat fra det almindelige udviklingsarbejde

20. Kan det med den ønskede grad af tydelighed konstateres, at det endelige system vil have den krævede grad af brugervenlighed

- kravet til denne egenskab skal påvises opfyldt i analysefasen

21. Kan det med den ønskede grad af tydelighed konstateres, at det endelige system vil have den krævede grad af indbyggede hjælp- og støttemuligheder
- kravet skal påvises opfyldt i analysefasen

22. Kan det med den ønskede grad af tydelighed konstateres at det endelige system vil have den krævede grad af sikkerhed
- kravet til denne egenskab skal påvises opfyldt i analysefasen

23. Dokumenterer analysen omfanget og udseendet på systemdokumentationen
- det bør ligge som standard, men der kan være nogle valgmuligheder med hensyn til hvilke dele, der skal være blivende dokumentation

24. Dokumenterer analysen udseendet og omfanget af brugerdokumentationen
- det bør være en del af kontrakten, det skal blot konstateres opfyldt i analyseresultatet

25. Omfatter analysen en vurdering af hver funktions testbarhed
- resultatet af analysen er en samlet testplan for system- og/eller brugertest, og derfor skal tiderne holde

26. Er alle dele af analysen godkendt af de relevante interessenter
- hver del skal reviewes og godkendes, både for funktioners og datas vedkommende

27. Kan designerne godkende analysen som mulig at designe på den til rådighed stående tid
- en særlig variant på foregående spørgsmål

28. Stemmer analysens resultat med det vedtagne budget for at udvikle systemet
- der er brug for at estimere hvor mange mandetimer, den resterende del af arbejdet kræver, og at det er inden for budgettet

29. Giver analysen grund til at tro at grundlaget for Cost/Benefit beregningen har ændret sig
- det er et gedigent faresignal hvis en del af det forventede udbytte ikke kan påvises

30. Ud over de allerede stillede spørgsmål kan der så svares ja til at alle standarder og retningslinier for analysen er overholdte
- dette gælder også evt. vedtagne ISO-, AQAP- og IEEE-standarder

31. Kan analyseresultatet erklæres for fejlfrit
- i modsat fald skal svarene dokumenteres i reviewrapporten

32. Kan reviewdeltagerne underskrive reviewrapporten
- i modsat fald skal svarene dokumenteres i reviewrapporten

4. Checkliste til review af designfasen

Det kræver god håndværksmæssig kunnen at designe et enkelt og testbart system på grundlag af analysens løsningsbeskrivelse. Designreviewet skal konstatere, at det er gået godt, og checklisten skal sørge for, at det kan konstateres hurtigt og troværdigt.

Det er nyttigt at involvere opgavestillere og andre designere. Men det kalder også på medvirken af testere.

Spørgsmål:

1. Er analysen fuldt og helt omsat i designet
- spørgsmålet skal gerne besvares af både analytikere og designere

2. Er der mere med i designet end i analysen
- det skal motiveres, for principielt er det forbudt

3. Er der taget stilling til arkitektur
- bruges der standardløsninger hvor det er muligt

4. Optræder alle funktioner synligt i systemet
- med klar reference til løsningen

5. Optræder alle data synligt i systemet
- med klar reference til løsningen

6. Er alle de kommende fysiske komponenter beskrevet
- i en form der kan bruges som grundlag for konstruktion

7. Bliver hver entitet/klasse oprettet, læst, opdateret og slettet i en synligt, veldefineret og dokumenteret komponent
- evt. dokumenteret i et CRUD-diagram

8. Er alle attributterne på entiteter/klasser beskrevet
- det er en klar misforståelse at udsætte noget til konstruktionen

9. Er alle grænseflader beskrevet
- fordelt på de tre typer af interne, eksterne til andre systemer i samme driftsmiljø og eksterne til andre driftsmiljøer

10. Er alle skærmbilleder beskrevet og godkendt af opdragsgiver eller brugere
- de kan evt. fremstilles af brugerne selv

11. Er alle databaser definerede
- med en kraftig involvering af DBA, og evt. leveret af dem som underleverance

12. Er kravene til tilgængelighed kendte
- og er det sandsynliggjort at de kan opfyldes

13. Er kravene til ensartethed kendte

- og opfyldes de af designet

14. Er kravene til brugervenlighed kendte
- og opfyldes de af designet

15. Er designet testbart i sig selv
- det er det vi er i gang med at finde ud af

16. Leder designet til et testbart system
- der skal være grundlag for systemtest

17. Kan designet udbygges fremover
- og leder det til et fysisk system, der er let at udbygge

18. Er der tilstræbt genbrug
- som regel skal der gøres en indsats for at få det genbrug der teknisk er mulig

19. Øger det nye system den fremtidige mulighed for genbrug
- er det ønskelige øgede omfang fastsat formelt

20. Er ændringer til andre systemer identificeret og accepteret
- problemet er at de kan give ændringer i ens eget system, hvis de ikke kan gennemføres som forudsat

21. Er samspillet mellem manuelle rutiner og systemet beskrevet
- på en måde der kan forstås af brugerne af de manuelle rutiner

22. Følger designdokumentationen en vedtagen standard
- og synes andre, at den er læselig

23. Er navnestandarden fulgt
- detailspørgsmål på det foregående

24. Er 'exception-handling' beskrevet
- det vil afhængigt af systemet dreje sig om 'error-handling', 'error-correction', 're-entry', afvisninger, log af afvisninger, og fejlmeddelelser til brugere

25. Er backup og recovery fastlagt
- og er niveauet i overensstemmelse med en evt. politik på området

26. Er alle attributter beskrevet i et Repository
- eller en tilsvarende registrering der sikrer ajourførte og tilgængelige beskrivelser

27. Er alle valideringsregler beskrevet
- en databeskrivelse er ikke komplet uden

28. Er kravene til drift og support beskrevet
- de kan ikke afleveres tidligt nok for at give mulighed for at forberede sig

29. Er accessmetoder fastlagt og beskrevet
- gerne i form af formel accessmodellering

30. Er alle logiske input designede og beskrevne
- det er f.eks. skærmbilleder, transaktionsdata, lyspenne, berøringsfølsomme skærme, menuvalg, stregkoder og sensorer

31. Er alle logiske output designede og beskrevne
- det er f.eks. skærmbilleder, logfiler, transaktionsdata, svar, notifikationer og websider

32. Er krav til kapacitet kendte og beskrevne
- RAM, CPU, buffere og grafikkort

33. Er der specielle krav til systemprogrammel
- for eksempel operativsystem, transaktionsservere, DBMS, Application Server

34. Er krav til konvertering kendte og beskrevne
- konvertering kan være årsag til ændringer i analyse og design, hvis den ikke låses fast senest i designfasen

35. Er brugervejledninger og indbygget hjælp og støtte, designede
- det skal være færdigt før konstruktionsfasen begynder

36. Er uddannelsen i systemet designet, og godkendt af målgruppen
- kursusindhold og uddannelsesmateriale koster tid at fremstille og skal designes fornuftigt

37. Er alle underleverandører adviseret
- og på plads med formelle aftaler

38. Ud over de allerede stillede spørgsmål kan der så svares ja til at alle standards og retningslinier for designet er overholdte
- det gælder også evt. vedtagne ISO-, AQAP- og IEEE-standarder

39. Kan designresultatet erklæres for fejlfrit
- i modsat fald skal svarene dokumenteres i reviewrapporten

40. Kan reviewdeltagerne underskrive reviewrapporten
- i modsat fald skal svarene dokumenteres i reviewrapporten

5. Checkliste til review af kode

Det er en forudsætning for review af kode, at der har været gennemført en designfase, og at designet er reviewet. Kodens fysiske mængde kræver, at det detaljerede design er ført frem på komponentniveau. Reviewet af designet har derfor omfattet en del af de generelle egenskaber ved koden:

 robustheden for komponenten som helhed

 vedligeholdelsesvenlighed for hele designet
 nøjagtigheden af formler og beregninger
 nøjagtigheden (precision) af datavariable

Bemærkningerne skal ses i lyset af ønsket om at review på kode skal reduceres til mindst muligt. Alt hvad der kan reviewes og godkendes i faser forud for kodning, skal blive det. Kode er et relativt grundlag at kontrollere og måle på. Mængden af den gør den også mindre egnet til review.

Spørgsmål, der går på det specifikke sprog, er ikke medtaget. Der må opstilles standarder og checklister for det eller de sprog der benyttes.

Checklisten tager udgangspunkt i, at der reviewes en komponent ad gangen. Det er designere og kollegaer, der er reviewdeltagere.

Spørgsmål:

1. Er designede komponenter kodet i tilsvarende moduler
- eller begrund hvorfor der er flyttet rundt på koden i forhold til designet

2. Er hver komponent komplet i forhold til sin opgave
- eller får det hjælp af andre moduler

3. Er kodens kontrolstruktur acceptabel
- den letteste måde at kontrollere det på, er at bruge et af de simple forvaltnings-værktøjer, der viser struktur og sammenhænge, ved at bruge koden som input

4. Er kodens indrykning og disponering efter standard
- eller som minimum fornuftig

5. Er koden læsbar
- og udpeg på reviewet de konkrete steder der ikke umiddelbart er til at forstå

6. Er der dele af koden, der virker overflødige
- eller er det på nogen måde overraskende, at den er der

7. Er navnestandarden fulgt
- og er alle navne, uanset standarden, til at læse og forstå

8. Er koden robust i en grad, der hindrer nedbrud
- den generelle robusthed er reviewet i designet, men der skal checkes interne risici, fx:
    - mellemregninger
    - over- og underflow
    - zerodivide
    - løkkers start og slut
    - opslag i tabeller med en variabel

9. Er der interne buffere, dynamiske tabeller eller andet, der kan få overflow
- en uddybning af foregående spørgsmål

10. Bliver alle returværdier i in- og eksterne kald validerede
- stol aldrig på et andet modul

11. Er alle datavariable initierede
- stol aldrig på en kompilers eller et operativsystems adfærd, den ændrer sig i morgen

12. Er der beregnet en McCabe kompleksitet for modulet, og er den acceptabel (<10)
- det er nemmere at gøre på koden end på designet, hvor det skal tælles med håndkraft

13. Er 'exception-handling' tydelig
- den må ikke ligge som en manglende eller tom else-udgang, fordi det ikke kan ske

14. Er alle beregninger korrekte i forhold til kravspecifikationen
- det er især et spørgsmål om at programmøren har forstået algoritmers brug og indhold

15. Er alle beregninger korrekte i forhold til kompilerens oversættelse af dem
- det er bedst at sikre sig med parenteser og flere instrukser, hvis man er usikker

16. Er reviewerne trygge ved beregningerne
- det er et spørgsmål om gennemskuelighed og logisk opstilling af koden

17. Er koden vedligeholdelsesvenlig
- reviewerne skal vurdere, hvor nemt det bliver at ændre i den

18. Kan komponenten forstås på 15 minutter
- eller på den tid der er specificeret i standarden

19. Kan den forventede performance opfyldes
- det vil vise sig i systemtesten, men evt. flaskehalse kan identificeres på denne White-Box facon

20. Er koden testbar med hensyn til antallet af veje
- er antallet kendt

21. Er koden testbar med hensyn til den tid, der er til rådighed
- et spørgsmål der kan forekomme udviklere ligegyldigt, men ikke nødvendigvis for testere

22. Ud over de allerede stillede spørgsmål kan der så svares ja til at alle standards og retningslinier for konstruktion er overholdte
- det gælder også evt. vedtagne ISO-, AQAP- og IEEE-standarder

23. Kan konstruktionen erklæres for fejlfri
- i modsat fald skal svarene dokumenteres i reviewrapporten

24. Kan reviewdeltagerne underskrive reviewrapporten
- i modsat fald skal svarene dokumenteres i reviewrapporten

6. Checkliste til 'Test af test'

Hvor godt er testen lagt til rette? Og hvornår er det et godt spørgsmål at stille? Svarene bruges til at måle om testen er planlagt godt nok. Men det er ikke nok at stille det overordnede spørgsmål: Er vi forberedt godt nok? Der er brug for en konkretisering af det.

Checklistens indhold kan spredes ud på de øvrige checklister. Efter hver udviklingsfase skal en del af testen være planlagt, og de tilsvarende checkspørgsmål skal stilles. De er samlet her for overblikkets skyld, og for at give en model der kan bruges til en samlet godkendelse. Modellen er den konsulenter bruger, når de skal vurdere, om en test er planlagt godt nok. Modellen er inklusive scoring, og med retningslinier for at bruge resultatet.

Checklisten er ideel som afslutning på et testplanlægningsseminar. Spørgsmålene kan besvares enkeltvis af deltagerne og derefter gennemgås samlet, for at få en opfattelse af status.

Hvert spørgsmål kan besvares med et X i JA, et kryds i NEJ, eller en BÅDE/OG besvarelse ved at sætte kryds i begge kolonner.

Grunden til at der må besvares med både JA og NEJ er at modellen bruges i praksis. Når et rigtigt projekts planlægning vurderes, er der uvægerlig en håndfuld af spørgsmålene, der kun kan besvares fornuftigt med et både/og. Prøv selv.

Svarene behandles i to omgange. Først ved at give nogle point for JA/NEJ og BÅDE/OG. Derefter kategoriseres de rene 'NEJ'.

	Spørgsmål	**Ja**	**Nej**	
1	Der vedligeholdes en liste over fundne fejl			**4**
2	Der er udpeget en ansvarlig for test			**3**
3	Alle testere gennemgår en vedtagen uddannelse			**1**
4	Systemtesten udføres af deciderede testere			**3**
5	Testen planlægges i en selvstændig planlægningsproces			**3**
6	Testaktiviteterne optræder synligt på projektplanen			**4**
7	Alle testaktiviteter er bemandede			**3**
8	Alle testaktiviteter er tidsestimerede			**3**
9	Der er udarbejdet dokumenterede testcases			**4**
10	Testcases giver tilsammen en ønsket og kendt testdækning			**4**
11	Begrebet testdækning bruges overhovedet			**1**
12	Regressionstest kan udføres automatisk			**2**
13	Hver testsessions forløb logges			**4**
14	Kriteriet for afslutning af testen kan måles objektivt			**2**
15	Testen sker i niveauer, f.eks. modul, system, integration, etc.			**4**
16	Testerne anvender fælles kendte og anerkendte testteknikker			**1**
17	Resultatet af en test reviewes og godkendes			**3**
18	Interessenterne kan deltage i, eller følge, forløbet			**1**
19	Testcases planlægges før kodning			**4**
20	Testforløbets fremdrift og færdiggørelsesgrad måles løbende			**2**

21	Testplanlægningen er godkendt formelt			**3**
22	Testere har de ønskværdige og nødvendige værktøjer			**2**
23	Testforløbet er planlagt på grundlag af formelle retningslinier			**1**
24	Testforløbet designes og dokumenteres ens i organisationen			**1**
25	Det nødvendige tidsforbrug til test er estimeret før testen starter			**2**
26	Komponenters testbarhed og kompleksitet beregnes maskinelt			**2**
27	Fejlrapporter registreres i en database			**2**
28	Udviklere og testere bruger samme testbegreber			**1**

Første behandling af svarene: optælling af point

En JA-markering tæller 0 point
En NEJ-markering tæller 4 point
En markering i begge kolonner tæller 2 point

Den samlede score bedømmes således:

0 - 28 point:
Dette testforløb er velplanlagt. Det har gode muligheder for et roligt og vellykket forløb.

29 - 56 point:
Det ser godt ud, forudsat visse justeringer. Der er både mulighed for et tilfredsstillende som et utilfredsstillende forløb.

57 - 84 point:
Forløbet bliver højst sandsynligt utilfredsstillende. Det er nødvendigt at sætte aktiviteter i gang til at imødegå en klar risiko for et turbulent forløb.

85 - 112 point:
Testen skal tilrettelægges forfra.

I de tilfælde hvor der er brug for at sætte forbedrende tiltag i gang, er spørgsmålet så nu: hvad skal der ændres på.

For det første kan alle spørgsmålene vurderes enkeltvis. Det vil give grund til at ændre på nogle forhold. Men spørgsmålenes sammensætning er ikke tilfældig. For at få en indikation af hvad der skal forbedres, er der behov for at kategorisere svarene. Ellers kan man risikere at rette på forhold, der faktisk ikke giver den ønskede forbedring.

Anden behandling af svarene: kategorisering
Alle de rene NEJ'er, d.v.s. dem der har udløst fire point, skal placeres i en af fire kategorier. Tallet 1,2,3 eller 4 der er noteret ved hvert spørgsmål, repræsenterer kategorierne. Antallet af rene NEJ'er med et 1-tal optælles. Ligeså antallet af 2-taller, 3-taller og 4-taller.

De fire kategorier er:

1. Uddannelse
2. Værktøjer
3. Ressourcer & Organisation
4. Metoder & Teknikker

Fordelingen af de rene NEJ'er i hver kategori peger på de typer af mangler, der skal udbedres. Hvis det viser sig at der er behov for flere ressourcer, vil det ikke være smart at sætte ind med uddannelse.

Eks.: En projektgruppe har en besvarelse med fire rene NEJ'er, og det er på spørgsmålene 12, 14, 21 og 25. Vi ser at det giver tre i kategori 2 og 1 i kategori 3. Der er brug for at diskutere om de nødvendige værktøjer er til rådighed.

Spørgsmålenes placering i en kategori er diskutabel. Spørgsmålene er diskutable i det hele taget. Men ved at besvare dem får man den diskussion, der er brug for. Spørgsmålene skal bruges til at træffe beslutninger. Det betyder at man ikke får garanti for et succesfuldt projektforløb ved at kunne svare JA til det hele. Men man har forbedret sin planlægning markant, ved at have taget stilling til alle spørgsmålene.

Det samme gælder for kategorierne. Ved at placere NEJ'erne, får man en diskussion og beslutning om den tilsvarende kategori.

7. Checkliste til review af datamodel

Datamodellen skal reviewes og godkendes som en væsentlig del af testforløbet. Der er en vis diskussion om brugere, og/eller andre ikke IT kyndige, kan teste en datamodel. Checklisten kan bruges under alle omstændigheder. Enten er brugere deltagere på et fortolkende review, hvor de forklarer udviklerne hvordan modellen forstås. Eller også er de deltagere på en Walk-Through/Play-Through, hvor testdata eller scenarier, spilles igennem af brugere og udviklere i fællesskab.

Reviewet skal teste datamodellen, som en færdig model af systemet. Det sikrer den indholdsmæssige validering. Reviewet skal også undersøge om dataanalysen er udført ordentlig. Det er grunden til, at der er spørgsmål om andet end lige den tegnede model. F.eks. også om hvordan entiteterne er identificeret.

Spørgsmål:

Modellen

1. Er modellen komplet
- eller er der underforståede dele, der ikke er vist

2. Er alle entiteter vist
- det er et detailspørgsmål på det foregående, men reviewdeltagerne skal sige ja

3. Er relationerne korrekte
- og fuldt definerede

4. Er attributterne fuldt definerede
- det skal konstateres uden for datamodellen, typisk i et Repository

5. Kan datamodellen tolkes af brugere inden for forretningsområdet
- dokumenter de dele i reviewrapporten der ikke kan tolkes

6. Kan datamodellen forstås af udviklerne
- det er farligt med områder, der indholdsmæssigt er modelleret efter andres forskrifter, 'med ført hånd', uden at udvikleren har forstået det 'helt'

7. Er modellen ikke-redundant
- hver del af den skal være unik og nødvendig

8. Er modellen korrekt i forhold til standarden
- eller i det mindste i forhold til projektets egne retningslinier

De følgende spørgsmål er nødvendige at besvare i 0-fejl udvikling. Hvis modellen skal være komplet, korrekt og stabil. De er besvaret af udviklerne som en del af dataanalysen. Men de skal besvares igen i testen af modellen, for at være sikker på, at der ikke er huller.

9. Er hver entitet identificeret og verificeret med hensyn en forekomst af den:
- hvor den opstår i brugerorganisationen
- hvor og hvordan den indrapporteres
- hvem der indrapporterer den
- hvor og hvordan i systemet den oprettes
- hvor og hvordan i systemet den læses
- hvor og hvordan i systemet den ændres
- hvor og hvordan i systemet den slettes
- hvor og hvordan i systemet den kommunikeres til andre systemer
- hvor og hvordan i systemet den rapporteres som output
- hvor og hvordan i systemet den indgår i beregninger
- hvordan den kan testes i systemtesten
- hvordan den kan testes i brugertesten
- hvordan den vedligeholdes

8. Checkliste til review af moduldiagram

Mange udviklere sætter pris på at fremstille et moduldiagram. Det fungerer som en bro imellem løsningen der er specificeret i analysen, og det fysiske system på maskinen.
Checklisten tager sit udgangspunkt i, at der til reviewet foreligger et samlet moduldiagram over hele produktet.

Spørgsmål:

1. Har det enkelte modul den højest mulige binding (Cohesion)

2. Har det enkelte modul den lavest mulige kobling (Coupling)

3. Er nedbrydningen (faktoriseringen) komplet

4. Når moduler kaldes af andre: vurder rimeligheden

5. Passer grænseflader på begge sider af et kald

6. Udfører flere moduler det samme, det være sig fysisk eller logisk

7. Er nogle moduler unødigt specialiserede

8. Er strukturen overdreven 'kort' 'høj', 'tyk' eller 'tynd'

9. Er modulstørrelsen fornuftig i forhold til de kommende programmer

10. Er der opnået det mulige 'fan-in'

11. Er 'fan-out' ok

12. Ligger 'scope-of-effect' inden for 'scope-of-control'

13. Er der moduler med hukommelse (state memory)

14. Forekommer der 'decision-split'

15. Er der balance i diagrammet mellem den indgående og udgående proces ('afferent' kontra 'efferent')

16. Er kalderegler for moduler overholdt

17. Overføres/returneres der datavariable uden om kaldet

18. Overføres/returneres der kontrolvariable uden om kaldet

19. Vurdering af modulers binding:

- Hvis den eneste fornuftige måde at beskrive modulets funktionalitet på, er ved hjælp af en sammensat sætning, eller en sætning med et komma, eller en sætning med flere verber, har modulet sandsynligvis en lavere binding end 'funktionel'.

- Hvis sætningen indeholder tidsorienterede ord som 'først', 'næste', 'efter', 'derpå', 'start', 'skridt', 'da' eller 'indtil' har modulet sandsynligvis 'temporær' eller 'procedural' binding. Det kan, dog i de færreste tilfælde, indikere 'sekventiel' binding.

- Hvis der ikke følger et specifikt objekt umiddelbart efter verbet, er det sandsynligvis 'logisk' bundet. F.eks. 'valider alle transaktioner'.

- Ord som 'initier', 'oprydning', 'housekeeping' og 'afslutning' peger på temporær binding.

- 'Funktionelt' bundne moduler kan beskrives ved hjælp af een sætning, uden bisætninger, og ved hjælp af eet verbum.

9. Checkliste til review af klassemodel

Checklisten gælder både for klassemodellen der dokumenterer analysen, og for den detaljerede model der dokumenterer designet. Spørgsmålene er i så høj grad de samme, at det er lettere at justere på inspektionstidspunktet.

Spørgsmål:

1. Kan modellen forstås af brugere.
- selvfølgelig skal selve notationsformen forklares, men den samlede model skal kunne fortolkes af brugere

2. Kan den enkelte klasse forstås af brugere.
- identificer de klasser der ikke kan, og find årsagen.

3. Er modellen dokumenteret, således at der ikke er dele, der skal forklares.
- er der viden i udviklingsgruppen, der skal forklares eller tillægges, fordi den ikke er dokumenteret

4. Er systemets afgrænsning korrekt i forhold til den forretningsmæssige nytte.
- et vigtigt spørgsmål, fordi det er en af de mulige fælder, man kan falde i. Hele verden er fuld af objekter, der nemt kan defineres som klasser. Samtidig er der ikke matematiske regler for valg af klasser, så man skal sikre sig, at systemet ikke kun defineres af udviklere, der får øje på objekter.

5. Check klassernes navne. Kan de godkendes af brugerne.
- et spørgsmål der erfaringsmæssigt åbner videre diskussioner.

6. Check at der ikke er forvekslet subklasser med attributter, check begge veje.
- faktisk en ret almindelig fejl af 'nybegyndere'. Hvornår er 'bil' en klasse, og hvornår er det en attribut på klassen 'transportmidler'. Det er ikke en subklasse, hvis der skal registreres de mængder, man ejer, af hvert transportmiddel. Så langt de fleste gange giver det sig selv, men det skal checkes på inspektionen.

7. Kan alle klasser genkendes af brugerne som konkrete og eksisterende.
- simpelt check, for at se om der er skabt nye klasser af hensyn til systemet. Der skal skelnes mellem konkrete, som kan godkendes af brugerne, og abstrakte, der f.eks. bruges til at være 'overklasse' for flere subklasser. Typisk for at repræsentere fælles attributter eller metoder.

8. Er associationerne mellem klasser korrekte, og kan de valideres som sådan af brugere.
- korrekte associationer, og ikke mindst komplette, er vigtige, og at få dem godkendt og afgrænset.

9. Er associationernes attributter korrekte.

- hvis den aktuelle valgte metode tillader at associationer har attributter.

10. Er associationerne dikteret af systemets forretningsmæssige nytte eller optræder nogle af hensyn til systemets konstruktion.
- det er et godt spørgsmål til at verificere OO-analysen.

11. Er operationerne så klare, at så man kan definere konkrete metoder til dem.
- spørgsmålet er vigtigst efter analysefasen. Der vil man have operationerne defineret pr. klasse. Senere er det for sent.

12. Når der er flere metoder til en operation, udfører de så den samme operation.
- spørgsmålet stilles efter det detaljerede design. Det er en almindelig fejl at flere metoder er defineret til en operation, men hvor metoderne alligevel ikke kan siges at være ens, blot med en forskellig fysisk implementering.

13. Check antallet af associationer mellem klasser. Kontroller alle der ikke er binære.
- det er problematisk med mangesidede associationer, hvor tre eller flere klasser skal ses i sammenhæng. Det giver typisk flere fejl senere. Men det er f.eks. tilladt i OMT-metoden.

14. Check 'Traceability'. I implementeringsfasen skal enhver del af systemet, det være sig en attribut, metode, klasse, etc. kunne føres tilbage til designet. Enhver del af designet skal igen kunne føres tilbage til en del af analysen.
- selvom 'traceability' netop øges ved at bruge OO, er der grund til at sikre sig. Dette er en af de store fordele der *skal* nås med OO, af hensyn til den fremtidige systemforvaltning. Der er penge på spil i dette spørgsmål.

15. Check 'indkapsling'. Er den maksimal, således at alle attributter og metoder, der ikke er nødvendige at kende udenfor en klasse, heller ikke er det.
- dette er også et af de centrale spørgsmål, som er vigtigt, fordi det er en af fordelene, der skal opnås. Ved at få indkapslet en del af attributterne og operationerne, fås en nemmere forvaltning, og mindre omskrivning, når systemet ændres, eller der rettes en fejl.

16. Check hierarkiets dybde med hensyn til klasser og subklasser, og den dertil hørende nedarvning af attributter og operationer. Hvis der er mere end fem niveauer, skal det begrundes med solide brugsmæssige argumenter.
- selvfølgelig kan det være rigtigt med et dybt hierarki. Men der kan også være sket det, at udviklerne er blevet teoretikere. Et tegn på det er store, smukt konstruerede hierarkier, med omfangsrige begrundelser for de enkelte niveauer af nedarvethed og videre delegering. De bør testes op imod virkelighedens verden.

17. Kan hver 'message' godkendes af brugerne.
- de skal forekomme som udveksling af meddelelser mellem den virkelige verdens virkelige objekter.

18. Er 'multiple inheritance' nødvendig når den optræder, og er den håndteret korrekt.
- selvom det kan give fordele, er det også en mulighed for at skabe kompleksitet.

10. Checkliste til overdragelsesreview

Systemet er nu færdigudviklet og -testet. I overdragelsen krydses de punkter af, der skulle have været checket og godkendt før. De bliver det igen, men med en enkelt ny detalje: Testerne på reviewet er systemforvaltere, der godkender at systemet kan accepteres, til de kommende års aktive systemforvaltning.

Spørgsmål:

1. Det forretningsmæssige
Opfylder systemet de forretningsmæssige krav.
Findes der en liste over udestående ændringsønsker, og hvad er status på de enkelte punkter.

2. Test
Er der sat et mål for testens dækningsgrad.
Er den ønskede dækningsgrad nået.
Er den opnåede dækningsgrad dokumenteret.
Eksisterer der et testmiljø til det nye system.
Eksisterer der testdatabaser til systemet, med testklasser og -metoder.
Er testen dokumenteret i form af testaktiviteter.
Eksisterer der testcases til systemet.
Er der udarbejdet en testdrejebog til system-/brugertest.

3. Kvalitetsstyring
Er den gennemført.
Er den dokumenteret.
Er den godkendt som tilstrækkelig.
Er der brugt kendte standarder for analyse, design og konstruktion.
Er der brugt en kendt standard for dokumentationen.

4. Dokumentation
Er den til stede.
Er den afprøvet.
Er den godkendt.
Er den komplet.

5. Implementering
Er bruger forberedt på igangsætning, herunder ændringer i:
- uddannelse
- organisation
- jobbeskrivelser
- forretningsgange
- andre systemer

Er drift/produktion forberedt på igangsætning, herunder ændringer i:
- uddannelse
- procedurer
- afledte ændringer

Har der været en 'fryseperiode', jævnfør Aktiv Systemforvaltning

6. Forvaltningsegnethed
Er der formuleret en målsætning for forvaltningsegnethed.
Er den opnået.
Er den dokumenteret.
Er den afprøvet.
Har systemet kørt fejlfrit i en aftalt periode.
Er tidligere fejl dokumenterede.
Findes der en fejlstatistik/-analyse,
- f.eks. en 'first-pass' 'pass-rate'.
Er der registreret gennemførte ændringer og tilføjelser.
Er der udpeget en systemejer.
Er der udpeget en forvaltningsansvarlig.
Er der beregnet kompleksitet for programmerne,
- f.eks. McCabe eller Halstead.
Ville du gerne forvalte det.

7. Check øvrige systemegenskaber
Nøjagtighed
Robusthed
Ensartethed
Brugervenlighed
Hjælp- og Støttemuligheder
Testbarhed
Udbygningsvenlighed
Genbrug
Ydeevne
Sikkerhed

11. Checkliste til driftsreview

Efter ibrugtagning skal systemet jævnligt checkes for sin helbredstilstand. Der er brug for at finde ud af, hvornår det trænger til en gang 'fitness'. Reviewet skal afgøre om:
- systemet skal stabiliseres i sin nuværende form
- levetiden skal forlænges aktivt
- det er modent til redesign

Spørgsmål:

1. Forvaltningens status
Er der en forvaltningsaftale på systemet
Er aftalen i bekræftende fald overholdt af begge parter
Hvor mange ikke-planlagte timer er der brugt

2. Rettelser
Har der været usædvanlig få eller usædvanlig mange rettelser siden sidste review
Kan rettelserne grupperes, efter nogle fælles karakteristika
Hvad kostede rettelserne at gennemføre
Hvem påpegede, at der var en fejl
Var der et driftsnedbrud forud for rettelsen

I bekræftende fald, hvorfor
Hvad kostede nedbruddet
Hvad var årsagen til fejlen
Har der været fejl før i denne del af systemet
Gennemføres rettelser tilstrækkelig hurtigt

3. Tilpasninger
Har der været usædvanlig få eller usædvanlig mange tilpasninger siden sidste review
Er der planlagte tilpasninger
Hvad består tilpasningerne i, og kan de karakteriseres
Hvem har bedt om tilpasningerne
Hvad kostede tilpasningerne
Hvad er udbyttet
Er de gået i drift uden problemer
Gennemføres tilpasninger tilstrækkelig hurtigt

4. Udbygninger
Har der været usædvanlig få eller usædvanlig mange udbygninger siden sidste review
Hvad kostede udbygningerne
Hvilke fordele har de givet
Hvem har bedt om udbygningerne
Er udbygningerne siden sidste review gået tilfredsstillende
Har udbygningerne været nemme at gennemføre i systemerne, sammenlignet med andre systemer
Gennemføres udbygninger tilstrækkelig hurtigt
Har brugerne mulighed for selv at gennemføre udbygninger

5. Sanering
Har der været gennemført saneringer i systemet siden sidste review
Hvorfor / Hvorfor ikke
Er der dele af systemet der skal undersøges til næste review, med henblik på mulig sanering
Lagres der data der ikke bruges af funktioner
Er der information eller statistik indbygget i systemet, der oplyser om brugen eller ikke-brugen
Hvordan kan vi iøvrigt konstatere en potentiel sanering i netop dette system

6. Generelt om indgrebene i systemet
Har brugerne været tilfredse med forløbet af indgrebene
Har IT selv været tilfreds
Har alle indgrebene været styrede og planlagte
Har dokumentationen været god nok
Er dokumentationen ført ajour: rettet, tilpasset, udbygget og saneret
Er testmiljøet ført ajour, d.v.s. rettet, tilpasset, udbygget og saneret
Er testdata ført ajour, d.v.s. rettet, tilpasset, udbygget og saneret

7. Nedbrud
Hvor mange nedbrud har der været siden sidste driftsreview

Hvor mange nedbrud oplevede brugerne
Er det flere nedbrud end 'normalt'
Hvad har været konsekvenserne af nedbruddene
Hvad kostede nedbruddene, helst i kroner, ellers i tid
Har der været særlige reaktioner på nedbruddene fra bruger, drift, udvikling eller andre

8. Belastning
Har der været øget belastning i form af transaktioner, brugere, data eller lignende
Er der performance-problemer
Er der ændringer i svartider, CPU-forbrug, plads, eller i frekvensen af brugen
Er der ændret frekvens i brugen af dele af systemet

9. Systemegenskaber
Hver systemegenskab: er der markant ændring i forhold til sidste driftsreview

10. Grænseflader
Hvordan fungerer grænsefladerne til andre systemer
Har der været mange ændringer på strukturen på input
Har der været mange ændringer på strukturen på output
Er der opstået nye grænseflader til/fra systemet

11. Den daglige håndtering
Vurder automatisering af driften, er den fuld/omfattende nok
Vurder mængde og kvalitet af manuelle procedurer
Har der været afvigelser i forhold til normal drift
Har afvigelserne krævet en særlig indsats/håndtering

Kvikstart

Udpeg de steder i projektforløb og forvaltningsopgaver der skal checkes ved hjælp af checklister. Udpeg en 'ejer' af hver checkliste. Sørg for at der er en ægte forståelse af, at ejerskabet betyder en aktiv holdning til at rette, tilpasse, udbygge og sanere checklistens indhold. Bed ejeren foreslå indholdet af listen. Det nemmeste er at lægge ud indholdet af dette kapitel som grundlag. Sørg for at listen revideres løbende, men sørg også som minimum for et formelt review en gang om året.

Gør listerne online-tilgængelige, så der ikke hersker tvivl om nyeste version.

Tilbage til toppen af tekst