Overdruk

‘Softwareproductkwaliteit’

Florijn & Greefhorst

informatie 0101

Softwareproductkwaliteit

Ervaringen en ontwikkelingen

Met de groeiende interesse voor softwarearchitectuur staat het begrip softwareproductkwaliteit ook weer volop in de belangstelling. Immers, een architect moet de eisen en wensen van diverse belanghebbenden vertalen in een uitgebalanceerde oplossing.

Gert Florijn en Danny Greefhorst

Ondanks diverse (theoretische) ontwikkelingen en standaardisatie-activiteiten rond productkwaliteit blijkt de praktijk toch weerbarstig te zijn. Regelmatig zijn er situaties waarin eisen niet of onvoldoende expliciet of controleerbaar zijn geformuleerd en waardoor een systeem uiteindelijk niet blijkt te voldoen.
In dit artikel wordt daarom nog eens wat dieper ingegaan op het begrip softwareproductkwaliteit. Dit gebeurt aan de hand van ervaringen met het gebruik van en verder onderzoek naar een bestaand kwaliteitsmodel: het *extended ISO-model*[1] (Van Zeist e.a., 1996). Niet alleen worden diverse toepassingsmogelijkheden van dit model besproken, ook wordt ingegaan op beperkingen en nieuwe ontwikkelingen.

Kwaliteitsmodel

De vraag of een systeem 'goed' of 'slecht' is, is niet zo maar te beantwoorden. Een 'goed' systeem biedt natuurlijk de functionaliteit die is gewenst en is afgesproken – bijvoorbeeld voor de administratie en verwerking van theaterboekingen. Maar daarnaast moet het ook voldoen aan allerlei (andere) kwaliteitseisen, bijvoorbeeld dat een boeking binnen een bepaald aantal seconden kan worden doorgevoerd, of dat – in het geval van storingen – het systeem weer snel operationeel kan worden gemaakt. Kwaliteit is een relatief begrip in de zin dat diverse belanghebbenden op verschillende manieren kijken naar een systeem en dus verschillende (en soms zelfs conflicterende) eisen zullen stellen. Zo kan een systeembeheerder vanuit beveiligingsoogpunt bepaalde beperkingen eisen die het gebruiksgemak voor de eindgebruiker negatief beïnvloeden.

Om de gewenste kwaliteit expliciet te kunnen maken is een begrippenkader of kwaliteitsmodel nodig. De kern van zo'n model is een overzicht van eigenschappen of attributen van een systeem waarvoor eisen kunnen worden vastgesteld. Om eisen ook echt te kunnen vastleggen zijn indicatoren nodig: aspecten die zijn te meten aan het systeem of het gebruik ervan. In een specificatie kan dan de indicator worden opgenomen met de gewenste waardes. Zo is 'mean time to repair' een indicator voor de kwaliteitseigenschap 'recoverability' (herstelbaarheid). Voor de eenduidigheid is het verder van belang dat de wijze van meten van zo'n indicator precies wordt vastgelegd.
Het extended ISO- of *Quint*-model biedt zo'n kwaliteitsmodel inclusief indicatoren en meetvoorschriften. Er zijn in totaal 32 eigenschappen in het Quint-model gedefinieerd, die zijn gegroepeerd onder zes

Samenvatting

Softwareproductkwaliteit krijgt steeds meer aandacht. Op basis van eigen ervaringen en onderzoek met het kwaliteitsmodel extended ISO bespreken de auteurs toepassingsmogelijkheden en beperkingen. Om kwaliteit te definiëren is een eenduidig begrippenkader nodig. Het extended ISO- of Quint-model is zo'n begrippenkader, inclusief indicatoren en meetvoorschriften.

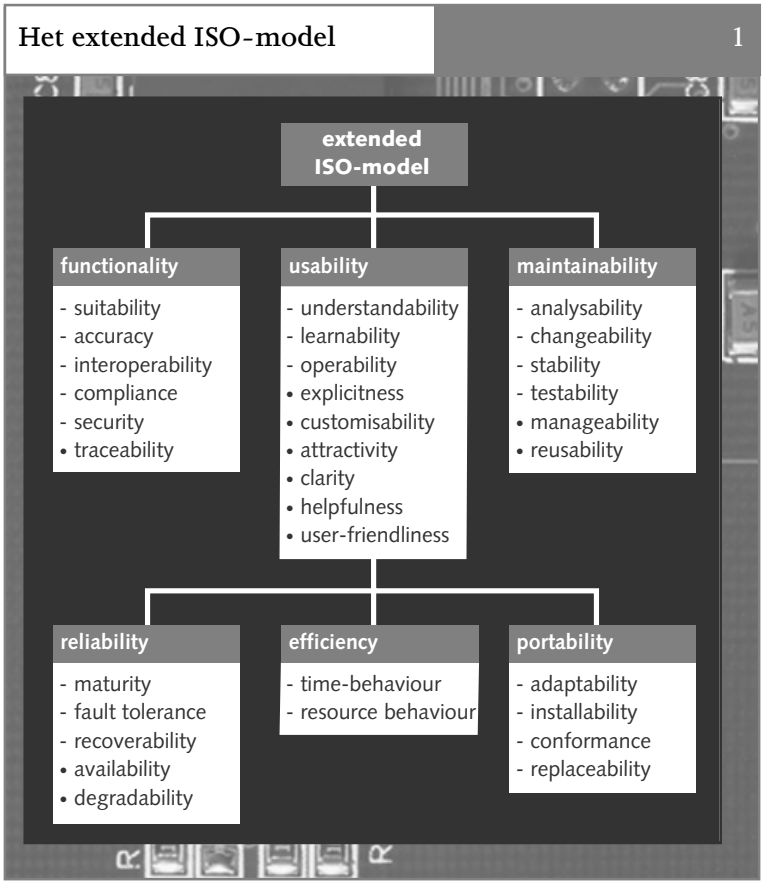
hoofdgebieden: functionality, reliability, usability, efficiency, maintainability en portability (zie figuur 1).

Achterhalen en opstellen van specificaties

Een van de belangrijkste doelen van een kwaliteitsmodel is het precies vastleggen van de specificaties voor een te bouwen systeem. Middels indicatoren kan dat zodanig gebeuren dat ook gecontroleerd kan worden of het geleverde systeem daaraan voldoet. Dit klinkt simpel, maar het opstellen van goede kwaliteitsspecificaties blijkt in de praktijk lang niet altijd zo eenvoudig. Een paar typische problemen zijn:

- het beeld over het toekomstige systeem is onvoldoende uitgekristalliseerd
  Als niet duidelijk is wat het systeem gaat doen, is het ook lastig om er specifieke eisen aan te stellen. Als bijvoorbeeld niet duidelijk is of een systeem interactieve toegang ondersteunt, kunnen de eisen rond bijvoorbeeld bruikbaarheid of tijdgedrag (denk aan reactietijd) niet goed worden vastgesteld.
- niet alle belangen zijn meegenomen
  Wanneer niet alle belanghebbenden zijn geraadpleegd zijn de eisen onvolledig. Bij introductie en gebruik van een systeem komen deze tekortkomingen aan het licht, maar zijn ze vaak moeilijk te verhelpen.
- onrealistische of ongebalanceerde eisen
  Het voldoen aan kwaliteitseisen kost geld en tijd. Daarom is het van belang dat de eisen realistisch zijn en dus dat de belanghebbenden zich realiseren wat de kosten zijn die aan het realiseren van hun eisen zijn verbonden. Dit geldt ook voor het totale eisenprogramma: de eisen van de verschillende belanghebbenden moeten onderling op elkaar zijn afgestemd zodat het geheel ook realiseerbaar blijft. Conflicterende eisen (denk aan snelheid versus flexibiliteit) moeten worden voorkomen.

Het extended ISO-model 1



Een model als Quint kan helpen dit soort problemen te voorkomen. Er zijn bijvoorbeeld positieve ervaringen opgedaan met het gebruik van Quint in gestructureerde workshops waarin het toekomstbeeld omtrent een systeem wordt geanalyseerd. In deze aanpak selecteren de diverse belanghebbenden van het systeem – ieder vanuit hun rol – de belangrijkste kwaliteitseigenschappen en identificeren ze scenario's die weergeven wat het systeem in de toekomst (idealiter) zou moeten bieden. Op basis van discussie tussen de belanghebbenden worden de attributen en scenario's vervolgens geprioriteerd, waarbij aspecten als realiseerbaarheid, kostenschattingen, time-to-market en marktpotentieel worden beschouwd. Hiermee wordt de essentiële functionaliteit van het systeem duidelijk(er) en worden tevens de be-

langrijkste eisen vastgesteld. Specifiekere en meer gedetailleerde eisen kunnen daarna worden vastgelegd.[2]

De combinatie van verschillende gezichtspunten met de systematiek van het kwaliteitsmodel zorgt voor een breed en volledig beeld van het doelsysteem en van de voorwaarden waaraan het moet voldoen. Het inschatten van de impact van scenario's vereist natuurlijk wel de nodige kennis op architectuurniveau, of soms zelfs het uitvoeren van architecturale prototypes. Dit betekent dat ook meerdere cycli nodig kunnen zijn.
Verder is het van belang dat alle relevante gezichtspunten en bijbehorende belanghebbenden betrokken worden. Quint onderscheidt daarbij drie hoofdgroepen: gebruikers, ontwikkelaars (inclusief beheerders) en evaluatoren.[3] De belanghebbenden moeten het model kunnen toepassen vanuit hun rol. Dat wil zeggen dat zij kwaliteitsattributen kunnen begrijpen en beschouwen vanuit hun eigen perspectief, en hun eisen daaromtrent kunnen vaststellen. Een goed begrip van de attributen is daarbij belangrijk. De praktijk leert dat definities alleen hiertoe niet voldoende zijn. Een voorbeeld van een mogelijk aandachtspunt is een essentieel hulpmiddel om dit begrip te bevorderen. Daarnaast moet men in staat zijn om ook buiten de eigen rol te denken om de posities en voorgestelde eisen van andere belanghebbenden te kunnen begrijpen en bespreken.
Het ligt voor de hand dat niet alle kwaliteitsattributen in elke situatie een even prominente rol spelen of dezelfde (soort) invulling krijgen. Zo zal accuracy in de context van financiële systemen tot andersoortige eisen leiden dan in een grafische omgeving. Anderzijds is het gevaarlijk om te snel te denken dat bepaalde attributen niet relevant zijn voor een bepaald soort systeem. Zo zal een systeem dat geen userinterface heeft maar alleen een API, toch bepaalde bruikbaarheidskenmerken moeten hebben.

Het moge duidelijk zijn dat een iteratieve werkwijze waarbij een systeem in incrementen wordt ontwikkeld vraagt om incrementele definitie van de kwaliteitseisen. Het is echter van belang de eisen die betrekking hebben op het systeem als geheel, reeds in een eerste iteratie vast te leggen. Daarnaast stelt een iteratieve aanpak ook grotere eisen aan de beheersing van eisen, doordat wijzigingen in eisen nu eenmaal als natuurlijk worden beschouwd.

Borgen van kwaliteitseisen

In het algemeen geldt dat eisen zo vroeg mogelijk in een ontwikkeltraject dienen te worden geverifieerd en gevalideerd, aangezien de kosten voor aanpassingen dan vele malen lager liggen dan later in het traject. Het probleem is echter dat veel karakteristieken van software pas in een laat stadium meetbaar zijn, omdat zij uitgaan van een werkend systeem. Om hier aan tegemoet te komen zijn twee oplossingsrichtingen te onderkennen. Enerzijds kan het helpen zo vroeg mogelijk over eigenschappen van het systeem te redeneren, door het opstellen van een softwarearchitectuur. Anderzijds kan een iteratieve aanpak worden gebruikt om belangrijke eisen aan een systeem snel te valideren. Hierbij kunnen architecturale prototypes snel inzicht bieden in de haalbaarheid van bepaalde eisen.
Een softwarearchitectuur documenteert de belangrijkste ontwerpbeslissingen voor een systeem, en vormt daarmee het uitgangspunt voor het uiteindelijke systeem. Een architectuur is dan ook de aangewezen plaats om zo veel mogelijk kwaliteitseisen die aan het systeem worden gesteld te borgen. Het uiteindelijke systeem zal deze uitgangskaders moeten respecteren, en zal de kwaliteitseisen uiteindelijk moeten beklinken.

Een voorbeeld van het vroegtijdig borgen van kwaliteitseisen is het binnen SERC uitgevoerde onderzoek naar performancemodellering. Hierbij wordt in een vroeg stadium een (UML-)model van het systeem gecreëerd dat wordt aangevuld met voor performance relevante (geschatte) informatie. Nog voordat een systeem wordt gerealiseerd kan deze informatie gebruikt worden om performancevoorspellingen te doen, waarbij knelpunten kunnen worden gesignaleerd. Tijdens het ontwikkelproces zal er verschuiving plaatsvinden van schattingen naar meetwaarden, waardoor de voorspelde performance van het uiteindelijke systeem gedurende het ontwikkelproces steeds reëler wordt.

Het vroegtijdig kunnen redeneren over de mate waarin systemen voldoen aan kwaliteitseisen is natuurlijk mooi, maar idealiter worden systemen ontwikkeld die inherent

voldoen aan kwaliteitseisen. Hiertoe zouden we moeten beschikken over een soort kookboek dat voor iedere kwaliteitseis aangeeft hoe deze kan worden vervuld in het systeem. Het is echter belangrijk op te merken dat een dergelijk kookboek geen kwaliteitsgaranties kan geven, aangezien kwaliteitseisen ook in sterke mate aan elkaar gerelateerd zijn. Zo kan het verhogen van de betrouwbaarheid van een systeem door het dubbel uitvoeren van componenten al snel een negatieve invloed hebben op de performance.

Het is dus noodzaak de relaties tussen verschillende kwaliteitsattributen in kaart te brengen. Hiermee kan inzicht worden gecreëerd in de impact die een kwaliteitsattribuut heeft op andere kwaliteitsattributen, en de afwegingen die hierbij moeten worden gemaakt. Als hulpmiddel kunnen kwaliteitseisen worden gerelateerd aan architecturale elementen die invloed hebben op de kwaliteitseis. Elementen waaraan meerdere kwaliteitseisen zijn gekoppeld zijn waarschijnlijk plaatsen waar afwegingen dienen te worden gemaakt.

In het algemeen is het relateren van (kwaliteits)eisen aan architecturale elementen van belang. Niet alleen ontstaat hierdoor duidelijkheid over de wijze waarop aan kwaliteitseisen wordt voldaan, ook is het een handig hulpmiddel om de impact van wijzigingen te bepalen. Het relateren van kwaliteitseisen is echter niet altijd even eenvoudig, aangezien eisen slechts bij uitzondering kunnen worden toegewezen aan specifieke elementen. Zo is bijvoorbeeld de performance van een systeem nu eenmaal niet gelokaliseerd in één component. Een uitzondering is learnability, dat vaak in het user-interfacedeel van het systeem kan worden gerealiseerd (denk aan een aparte tutorial).
Zoals eerder aangegeven dient gedurende het gehele ontwikkeltraject aandacht te worden besteed aan de kwaliteit van het systeem. Om deze te bepalen dienen er indicatoren te zijn gedefinieerd die kunnen worden gebruikt om de kwaliteitseisen te valideren. Hierbij kan voor de controle van de kwaliteitsattributen worden gebruikgemaakt van standaardraamwerken en -processen. Ook methoden voor het bepalen van specifieke kwaliteitsattributen kunnen worden ingezet. Zo'n methode is bijvoorbeeld *SAAM*, de software architecture analysis method (Bass e.a., 1998), die zich voornamelijk richt op de onderhoudbaarheid van een systeem. Kernpunt in SAAM is het scenario. Het scenario beschrijft een typisch gebruik of verwachte wijziging in het systeem vanuit het oogpunt van een belanghebbende. Wanneer realisatie van een scenario zou leiden tot aanpassingen in de architectuur, wordt dit geregistreerd (waar moeten aanpassingen plaatsvinden) en wordt tevens een schatting gegeven van de omvang van de aanpassingen (bijvoorbeeld in kosten). Vervolgens worden de effecten van alle scenario's gecombineerd om tot een totale score van de architectuur te komen. Op basis van een suite van scenario's kan bijvoorbeeld voor een bepaalde architectuur worden gekozen.

een kwaliteitsmodel is nooit af; hanteer het dus niet in een keurslijf

Ontwikkelingen

Het ISO/IEC 9126-model waarop het Quint-model is gebaseerd, dateert uit 1991. Inmiddels is daar een nieuwere versie van in ontwikkeling. Deze maakt onderscheid tussen de interne en de externe kwaliteit van software, en bijbehorende metrieken. Interne kwaliteit is alleen meetbaar door naar de interne opbouw van het systeem te kijken. Externe kwaliteit wordt gedefinieerd door het buitenaf waarneembaar gedrag van de software, en wordt beïnvloed door de interne kwaliteit van de software. Een andere term die wordt geïntroduceerd in het nieuwe ISO/IEC 9126-model is *quality in use*, wat overeenkomt met de kwaliteit van de software zoals de gebruiker die in zijn dagelijkse werk ervaart. Deze quality in use wordt gedefinieerd door vier karakteristieken: effectiveness, productivity, safety en satisfaction. Zowel de interne als de externe kwaliteit van de software zijn van invloed op de quality in use.

Andere wijzigingen in het nieuwe model zijn de introductie van normatieve subkarakteristieken die

grotendeels overeenkomen met de informatieve subkarakteristieken uit de vorige versie, en de separate definitie van het evaluatieproces in de ISO/IEC 14598-standaard. Recent is de reïntegratie van deze twee standaarden voorgesteld onder de naam Square (Software Product Quality Requirement and Evaluation), waarbij ook expliciete aandacht komt voor elementaire (ontwerp)-metrieken en kwaliteitseisen.

Gert Florijn en Danny Greefhorst

zijn werkzaam bij het Software Engineering Research Centre (SERC) te Utrecht.

Conclusies

Er is in de praktijk steeds meer belangstelling voor software-productkwaliteit. Om hier zinvol mee om te gaan is een kwaliteitsmodel nuttig en noodzakelijk. De hier beschreven ervaringen laten zien dat zo'n kwaliteitsmodel op diverse momenten in het ontwikkeltraject zeer goede diensten kan bieden. En daarbij gaat het zowel om het definiëren van eisen als om het borgen van de kwaliteit. We moeten ons hierbij wel realiseren dat een kwaliteitsmodel nooit 'af' is en dus ook niet als een keurslijf moet worden gehanteerd. Praktijkgebruik leidt tot (situationele) uitbreidingen en aanpassingen. Zo is het soms nuttig een deelaspect van een bepaald attribuut een prominentere status te geven – een recent voorbeeld hiervan is 'schaalbaarheid'. Verder kunnen de toepassingen in verschillende domeinen zorgen voor betere toelichtingen en een rijker pakket aan indicatoren. De terugkoppeling van deze ervaringen naar het kwaliteitsmodel is dus cruciaal.

Noten

1. **Het extended ISO-model is gebaseerd op het ISO/IEC 9126-model en is mede door SERC opgesteld.**
2. **Merk op dat deze aanpak enigszins vergelijkbaar is met de Quality Attribute Workshop zoals voorgesteld in Barbacci (2000).**
3. **In de praktijk blijkt een wat vollediger checklist van mogelijke belanghebbenden nuttig te zijn.**

Literatuur

Barbacci, M:. 'Quality Attribute Workshops', *news@sei interactive*, vol 3, nr. 2, spring 2000, http://interactive.sei.cmu.edu/news@sei/columns/the_architect/2000/spring/the_architect.htm.

Bass, L., P. Clements and R. Kazman: *Software Architecture in Practice*, SEI Series, Addison-Wesley, 1998.

Zeist, B. van et.al.: *Kwaliteit van software producten, praktijkervaring met een kwaliteitsmodel*, Kluwer Bedrijfsinformatie, 1996.

Aankondiging seminar

Op **20 februari 2001** organiseert SERC het seminar **'Theorie in praktijk – softwareproductkwaliteit'**.

Het seminar start met een overzicht van de ontwikkelingen op het gebied van modellen voor softwareproductkwaliteit.

Sprekers van verschillende bedrijven en instellingen zullen vervolgens toelichten welke rol aandachtsgebieden uit dergelijke modellen in de (ontwikkel)praktijk kunnen spelen.

Aspecten rond productkwaliteitsgebieden als useability, reliability, efficiency, maintainability en portability zullen aan bod komen.

Schrijf u nu in voor dit seminar of kijk voor meer informatie op http://www.serc.nl

Bellen of emailen kan ook:
030-2545412 of seminars@serc.nl.