

RICHTLIJNEN VOOR EEN SOFTWARE ARCHITECTUUR DIE BIJDRAAGT

Robert Deckers, Ruud Steeghs

Een goede software architectuur geeft richting aan het realiseren van software systemen die de bedrijfsdoelen dienen. Om bij te dragen aan de bedrijfsdoelen moet de software architectuur correct, consistent en communiceerbaar zijn. De mate en balans waarin dit gebeurt, bepaalt of de software architectuur goed is. Dit artikel geeft richtlijnen voor het opstellen van een goede software architectuur¹.

Allereerst wordt de definitie van software architectuur gegeven. De elementen uit die definitie worden apart toegelicht om het begrip van de rol en inhoud van software architectuur te vergroten. Daarna worden de eigenschappen van een goede software architectuur behandeld. Per eigenschap worden vragen gegeven om de eigenschap te toetsen. Het artikel eindigt met 10 vragen voor de business manager aan de software architect. Deze vragen helpen de business manager om te toetsen wat de kwaliteit van de architectuur is. Alhoewel dit artikel zich toespitst op software architectuur is de inhoud ook prima toepasbaar op andere architectuurdisciplines.

Definitie van software architectuur

Om te kunnen bepalen wanneer een software architectuur goed is, moeten we een goed begrip hebben van de rol en inhoud van software architectuur. De definitie is:

De software architectuur van een systeem is een geheel van uitspraken dat richting geeft aan ontwerp, realisatie en evolutie van de software in zijn omgeving.

Het volgende kader licht de begrippen uit de definitie van software architectuur toe.

Software

Software is de Engelse term voor alle computerprogramma's, bibliotheken en bijbehorende data die niet aanwezig zijn bij het starten van een computer maar achteraf worden geladen [WP SW]. Dit artikel beschouwt software als onderdeel van een systeem. Een systeem kan (bijna) geheel uit software bestaan.

Systeem

De term systeem wordt gebruikt voor datgene wat gerealiseerd gaat worden. Een systeem hoeft niet puur uit software te bestaan. Een organisatie, een bedrijfsproces, een mobiele telefoon zijn systemen die uit meer dan alleen software bestaan.

Systemen zijn op verschillende aggregatieniveaus te onderkennen. Bijvoorbeeld:

- Een gehele organisatie
- Een specifiek bedrijfsproces
- Een informatiefunctie die gebruikt wordt in de bedrijfsprocessen.
- Een software service (of component).

Op al deze niveaus kan men een software architectuur opstellen.

Door het systeem zo specifiek mogelijk af te bakenen wordt duidelijk voor welke software de architectuur bedoeld is. Zonder een systeemafbakening kan niet goed getoetst worden of een systeem goed in zijn omgeving past en dus ook niet of de software architectuur de eisen ondersteunt die de omgeving eraan stelt.

¹ Dit artikel is gebaseerd op een hoofdstuk uit het in 2010 te verschijnen boek "DYA|Software, Architectuur op maat", dat onderdeel uit maakt van de DYA suite [DYA].

Geheel van uitspraken

Een geheel van uitspraken betekent dat de uitspraken bij elkaar horen. Het moet duidelijk zijn welke uitspraken wel tot een architectuur en welke uitspraken niet tot een architectuur horen. Architectuuruitspraken moeten zo beschreven zijn dat ze met elkaar in verband te brengen zijn. Dat is nodig om afhankelijkheden in de architectuur als geheel toe te passen, impactanalyses te doen, de realisatie te voorspellen (plannen) en op consistentie te analyseren.

Uitspraken die richting geven

De architectuuruitspraken moeten richting geven aan het realisatieproces. De mensen die het systeem verder realiseren moeten de uitspraken dus begrijpen en vertalen naar handelingen. Een architectuuruitspraak is een beschrijving van de gewenste situatie of een richtlijn die tijdens ontwerp, realisatie en evolutie toegepast wordt.

Ontwerp, realisatie en evolutie

Met ontwerp bedoelen we een beschrijving van hoe het systeem in elkaar zit. Dit omvat niet alleen de interne componenten van het systeem en de daarbij toegepaste ontwerppatronen (technisch ontwerp), maar ook de functionele decompositie en bijbehorende decompositie van kwaliteitseisen (functioneel ontwerp).

Met realisatie worden de activiteiten bedoeld die leiden tot een operationeel systeem. Het realiseren omvat specificeren, ontwerpen, programmeren, integreren, testen, plannen, installeren, invoeren, etc. De realisatie omvat zowel de ontwikkeling als beheer.

Met evolutie wordt bedoeld hoe het systeem verandert als gevolg van een veranderende omgeving en/of systeemafbakening.

Omgeving

Elk systeem heeft interactie met zijn omgeving. Een systeem wordt vanuit meerdere omgevingsperspectieven beschouwd. Ten eerste, vanuit de gebruiksomgeving. Dit perspectief beschouwt de toepassing van het systeem in de omgeving waarvoor het bedoeld is. Ten tweede, heeft een systeem ook een constructieomgeving. Hierin wordt de wederzijdse invloed beschouwd tussen het systeem en de technologie en componenten waaruit het systeem is opgebouwd. Een derde perspectief is de levenscyclus van het systeem; van idee, via conceptfase, ontwikkeling, in gebruik name, beheer, tot einde van het gebruik.

Goede software architectuur

Het is de taak van de architect om een zo goed mogelijke architectuur op te stellen. Vele boeken over software architectuur (onder andere [Bass], [Rozanski], [Bosch]) reiken daarvoor technieken aan. Maar, welke technieken zijn op welk moment het beste? Om dat te bepalen moet je als architect weten wat een goede architectuur is. Pas dan wordt het mogelijk om gericht een architectuur op te stellen die bijdraagt.

We delen het begrip "goede software architectuur" op in deelaspecten. Het is makkelijker om te bepalen of de deelaspecten goed ingevuld zijn, dan om een gehele architectuur in een keer te beschouwen. De deelaspecten hebben ook als voordeel dat het makkelijker wordt om aandachtsgebieden voor verbetering van de architectuur te identificeren.

Een architectuur is goed als deze:

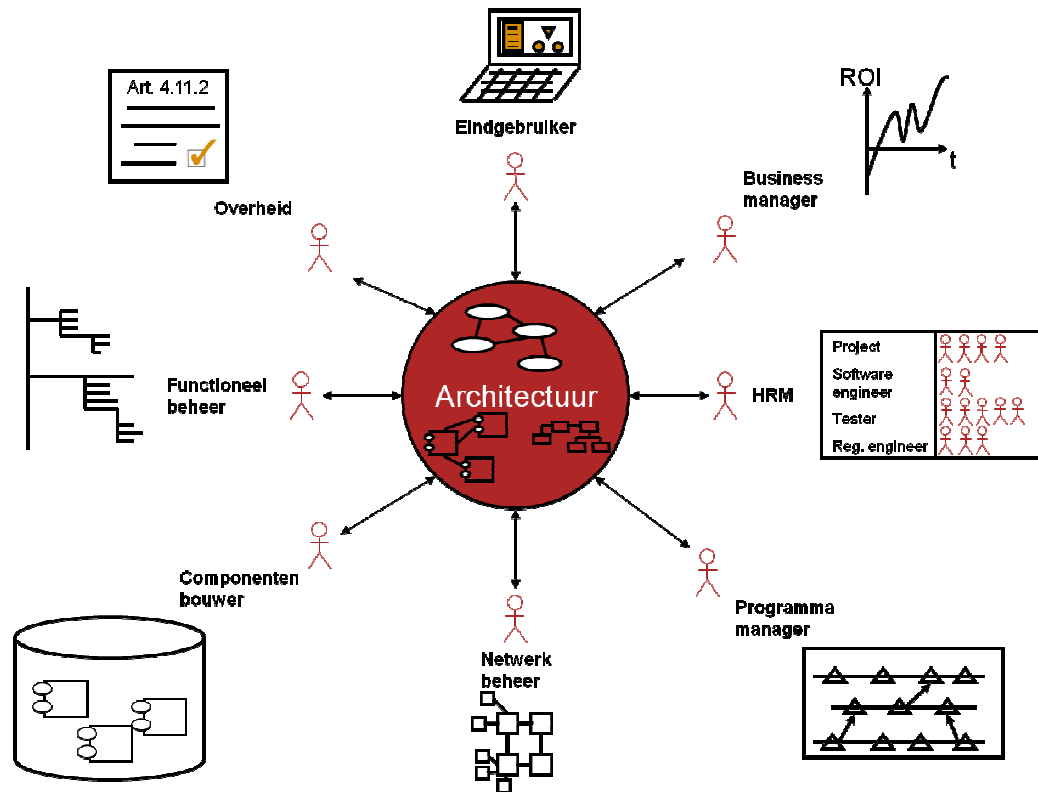
- *correct* is: de architectuur is gebaseerd op gevalideerde uitspraken over de systeemomgeving en de stakeholderbelangen in het bijzonder. De belangen zijn geprioriteerd. De architectuur vormt de juiste balans tussen de belangen.
- *consistent* is: de architectuuruitspraken zijn onderling niet strijdig. Dit geldt niet alleen voor de uitspraken van de architectuur zelf, maar ook in relatie tot de omgeving.
- *gecommuniceerd* is: de stakeholders zijn op de hoogte van hun relatie met de architectuur en ze weten hoe de architectuur hun activiteiten beïnvloedt.

Alhoewel uitspraken als "een architectuur moet flexibel zijn, modulair zijn, toekomstvast zijn" vaak van toepassing zijn, zijn zij niet universeel geldig. Er zijn altijd voorbeelden van systemen waarvoor die uitspraken niet gelden (of op zijn minst niet het belangrijkste zijn). Correctheid, consistentie en communiceerbaarheid zijn architectuurattributen die van toepassing zijn op alle systemen die ontwikkeld worden door mensen en waarbij de architectuur als doel heeft om de belangrijkste systeemeigenschappen te borgen.

De architectuur moet rekening houden met vele verschillende stakeholders en informatiebronnen. Figuur 1 laat zien dat verschillende stakeholders verschillende viewpoints met verschillende notaties hanteren om over hun taak te communiceren en te redeneren. De architectuur zal uitgedrukt moeten worden in de taal van de stakeholders en moeten passen op hun uitspraken en beeldvorming. In Figuur 1 is te zien dat de architectuur in overeenstemming moet zijn met bijvoorbeeld:

- de projectroadmap van de programmamanager,
- de netwerktopologie van de netwerkbeheerder/netwerkarchitect,
- de componenten aanpak en library van de componenten bouwer (en beheerder),
- de financiële planning van de business manager,
- de wetgeving vanuit de overheid,
- de functionele structuur van functioneel beheer,
- de user interface omgeving van de eindgebruiker,
- het opleidings- en aannamebeleid van de HRM afdeling

Natuurlijk zijn er in de praktijk nog veel meer stakeholders en specificaties waarop de architectuur moet passen. Daarnaast zullen de verschillende architectuurmodellen en -principes ook onderling in overeenstemming moeten zijn. Om deze te communiceren zal er een vaak vertaling naar de taal van de stakeholders gemaakt moeten worden.



Figuur 1 Architectuur: intern, in relatie tot de omgeving en door mensen gehanteerd

In de volgende paragrafen wordt dieper ingegaan op correctheid, consistentie en communiceerbaarheid van architectuur.

Correctheid van de architectuur wordt bereikt door het volgende:

- **Voldoende omgevingsanalyse:** er zijn voldoende stakeholders, stakeholderbelangen en andere informatie over de omgeving van het systeem in beschouwing genomen. Deze informatie moet ook specifiek genoeg zijn om de architectuur op te baseren. Bijvoorbeeld als de architectuur nieuwe technologie gebruikt, dan moet hiervoor ook de HR-manager niet vergeten worden om opleidingen en werving te verzorgen.
- **Bewuste afweging tussen belangen:** een richtinggevende architectuur is het resultaat van een bewuste en traceerbare afweging tussen verschillende en mogelijk tegenstrijdige belangen en andere omgevingsuitspraken. Stakeholders begrijpen dat hun belangen het onderwerp zijn geweest van een afweging met andere belangen, als gevolg van verschil in prioriteit tussen belangen. Voorbeeld: Bij gebruik van webservices zal versleuteling van elk afzonderlijk bericht ten koste gaan van de verwerkingstijd. De afweging die gemaakt moet worden is of de responstijd belangrijker is dan de beveiliging.
- **Gevalideerde omgevingsuitspraken:** de stakeholders herkennen en erkennen hun concerns en doelen. De uitspraken zijn hiervoor geprioriteerd. De prioritering kan gebruikt worden om afwegingen te maken tussen systeemeigenschappen die de belangen dienen. De prioritering wordt ook gevalideerd. Hiervoor is het zinvol om het totaalbeeld over de prioritering met de belangrijkste stakeholders te bespreken. Ook andere uitgangspunten over de omgeving worden gecontroleerd op geldigheid. Voorbeeld: aannames over toekomstige requirements vanuit de markt moeten gestaafd worden met onderzoeksrapporten over die markt.

Samengevat gaat het bij correctheid om de vraag "past de architectuur in haar omgeving?" In Figuur 1 is dit te zien als het hebben van de juiste omgevingsviews en de afbeelding van de architectuuruitspraken daarop.

Een bedrijfsinformatiesysteem moet tenminste passen op:

- de bedrijfsprocessen waar het systeem een ondersteuning voor moet zijn.
- De aard van de gebruikerslocatie en het niveau van de gebruikers.
- De interfaces en kwaliteitseisen van andere (software) systemen waarmee het systeem interactie heeft.
- De technische infrastructuur waar het systeem op moet passen.
- De strategie en roadmap voor de informatievoorziening waar het systeem onderdeel van is.
- De ontwikkel-/beheerprocessen (inclusief mensen, middelen en organisatie) waarin het systeem gerealiseerd wordt.

Om te komen tot een correcte architectuur kunnen onderstaande vragen beantwoord worden.

De volgende vragen dragen bij aan een voldoende omgevingsanalyse:

- Wat is de afbakening van het systeem? Denk aan afbakening in klantbehoefte, functionaliteit, technologie, realisatie en levensduur.
- Welke elementen uit de omgeving worden beschouwd (ook in de tijd)? Wat neem je wel mee en wat niet? Denk aan concurrenten, marktontwikkelingen, technologische ontwikkelingen, mogelijke partners.
- Wie zijn de stakeholders? Wat zijn hun belangen?
- Wat is belangrijke omgevingsinformatie? Denk hierbij aan inputdocumenten, marktinformatie, wetgeving,...
- Hoe wordt de omgevingsinformatie (per item) gebruikt als input voor de architectuur?

Een bewuste afweging tussen de belangen wordt bereikt door het beantwoorden van deze vragen:

- Wat is de prioritering van de verschillende belangen?
- Welke afwegingen zijn gemaakt in de architectuur?
- Hoe zijn de afwegingen terug te traceren naar de stakeholderbelangen en de prioritering?
- Op welke andere omgevingsinformatie zijn de afwegingen gebaseerd?
- Hoe is duidelijk of de stakeholders de afweging begrijpen?

Deze vragen helpen bij het valideren van de omgevingsuitspraken:

- Hoe (h)erkennen de stakeholders de beschreven belangen?
- Hoe zijn de andere uitgangspunten over de omgeving gecontroleerd op geldigheid?
- Hoe wordt de prioritering van belangen gevalideerd? Met name door de systeemeigenaar.

Consistentie² wordt bereikt door het volgende:

- Bewust vastlegging: het moet duidelijk zijn wat wel en niet tot de architectuurbeschrijving hoort. Dit wordt geregeld door in het ontwikkelproces te definiëren welke documenten er zijn en wat er in staat. Ook moet duidelijk zijn hoe de architectuuruitspraken beheerd worden. Hiervoor kan versiebeheer gebruikt worden en de tracering naar omgevingsuitspraken door middel van referenties geborgd zijn.

² Vanuit het perspectief dat architectuur een systeemeigenschap is, is deze per definitie consistent. Echter, een architectuurbeschrijving is mogelijk wel inconsistent en een systeem kan inconsistent zijn met de voorgeschreven architectuur.

- Verificatie op tegenstrijdigheid: alle architectuuruitspraken en relevante omgevingsuitspraken vormen een geheel. Dit geheel moet geanalyseerd worden op tegenstrijdigheden. Dit wordt gedaan door alle architectuuruitspraken met betrekking tot eenzelfde aspect met elkaar in verband te brengen. Dit kan middels het architectuur redeneemodel [ARM]
- Aantoonbare realiseerbaarheid: De architectuur moet zodanig zijn opgesteld dat deze toegepast kan worden bij het realiseren van het systeem. Hiervoor moet aangetoond worden of het systeem ook daadwerkelijk volgens de architectuur te bouwen is. Bijvoorbeeld doordat een architectuurprincipe reeds eerder toegepast is of bewezen is middels een proof of concept.

Samengevat gaat het bij consistentie om de vraag "zit de architectuur goed in elkaar?" In Figuur 1 is dit te zien als de onderlinge relatie tussen de architectuuruitspraken.

Consistentie geldt niet alleen als eis binnen de architectuurbeschrijving maar ook in relatie tot uitspraken over de omgeving. Zo moet de broncode consistent zijn met de architectuurprincipes die erop van toepassing zijn. Maar bijvoorbeeld ook de structuur en terminologie van de software requirements van een ontwikkelproject moeten passen op de architectuur. Het heeft weinig zin om een opdeling van de requirements te maken die niet te relateren is aan de onderdelen van het systeem. Daarnaast geldt ook dat de projectroadmap en werkpakketten uit projectplanningen moeten passen op de systeemopdeling in de architectuur.

Om te komen tot een consistente architectuur kunnen onderstaande vragen beantwoord worden.

De volgende vragen dragen bij aan een voldoende een bewuste vastlegging:

- Wat hoort wel en niet tot de architectuurbeschrijving?
- Hoe is de architectuur vastgelegd?
- Hoe wordt de architectuur beheerd?

Het verifiëren op tegenstrijdigheden kan door middel van deze vragen:

- Welke terminologie wordt gebruikt in de architectuurbeschrijving?
- Hoe wordt ervoor gezorgd dat de terminologie begrepen wordt?
- Hoe wordt de architectuur geanalyseerd op tegenstrijdigheden?
- Hoe is duidelijk dat elke architectuuruitspraak aansluit op een omgevingsuitspraak?
 - Passen ze op een stakeholderbelang of uitgangspunt over de omgeving?
- Hoe is duidelijk dat de belangrijkste architectuuruitspraken dezelfde belangen dienen?

De volgende vragen dragen bij het aantonen van de realiseerbaarheid:

- Hoe wordt ervoor gezorgd dat de architectuur beschreven zodat deze bruikbaar is tijdens realisatie?
- Hoe wordt ervoor gezorgd dat de architectuur toegepast wordt tijdens realisatie?
- Hoe wordt aangetoond of het systeem volgens de architectuur te bouwen is?
- Hoe wordt aangetoond of het systeem volgens de architectuur gebouwd is?

Een **gecommuniceerde** architectuur wordt bereikt door:

- Vertaalbaarheid naar acties: De betrokkenen rond de architectuur ondernemen op basis van de architectuuruitspraken de juiste acties. Dit houdt in dat het detailniveau van de modellen van het systeem de architectuurrichtlijnen genoeg moet zijn. De uitspraken moeten ook niet te gedetailleerd zijn. Anders worden de mensen die ze moeten toepassen te veel ingeperkt in de keuzes die zij nog moeten maken. Bijvoorbeeld: de projectleider moet een projectplanning maken op basis van de architectuur. De lead engineer weet hoe de principes

toe te passen in de programmacode. De technisch applicatie beheerder weet hoe het systeem te installeren en te configureren.

- Voldoende verankering: Er zijn genoeg stakeholders betrokken in de communicatie en zij moeten informatie van voldoende diepgang krijgen om de erkenning en toepassing van de architectuur te borgen. Bijvoorbeeld de DYA-aanpak [DYA] geeft richtlijnen hoe de architectuur in de organisatie verankerd moet worden.
- Bewuste uitvoering: De communicatie is gecoördineerd. De communicatievorm en inhoud is gepland en de architect weet wie waarvan op de hoogte is en/of input geleverd heeft. Bijvoorbeeld door middel van een communicatiematrix waarin is vastgelegd welke stakeholder, op welk moment, op welke manieren waarover gecommuniceerd wordt.

Samengevat gaat het bij communicatie om de vraag "weet iedereen wat hij moet weten?" In Figuur 1 is dit zien als het gebruiken van notaties in een view, die passen bij de betreffende stakeholder.

Om te komen tot een gecommuniceerde architectuur kunnen onderstaande vragen beantwoord worden.

De volgende vragen dragen bij aan de vertaalbaarheid naar acties door stakeholders:

- Welke rol speelt de architectuur in de activiteiten van de stakeholders? Wat moet elke stakeholder met de architectuur doen?
- Hoe is duidelijk of de stakeholders informatie van voldoende diepgang hebben? (om de juiste acties te kunnen ondernemen)
- Hoe weet je of de architectuur goed vertaald is? Wat doe je als het niet zo is?

Voldoende verankering wordt bereikt door het beantwoorden van de volgende vragen:

- Hoe is duidelijk of de belangrijkste/invloedrijkste stakeholders betrokken zijn in de communicatie?
- Wie erkent de architectuur en wie niet?
- Wat wordt gecommuniceerd over:
 - architectuurvisie,
 - voorspelde gevolgen,
 - behaalde successen
- Hoe wordt de architectuur ingebed in ontwikkelproces/-straat ?

Deze vragen dragen bij aan een bewuste uitvoering van de communicatie:

- Hoe wordt de communicatie gecoördineerd?
- Hoe wordt de communicatievorm en inhoud gepland/vastgelegd?
- Hoe is bekend wie waarvan op de hoogte is? (zou moeten zijn)

Tien vragen van de manager aan de software architect

In de voorgaande paragrafen is besproken wat het streven is om te komen tot een goede software architectuur. In een ontwikkelproces komt vaak de mijlpaal "architectuur gereed" voor. Echter, hoe bepaalt de QA manager, business manager, de projectleider of een programmeur of de architectuur ook daadwerkelijk gereed is? Omdat architectuur de meer kwalitatieve aspecten van het systeem adresseert en per situatie een andere verschijningsvorm kan hebben, zal het niet makkelijk zijn om een simpele test te doen. Er zijn echter verschillende toetsen mogelijk, met verschillende kosten en diepgang.

De meest uitgebreide is om door een onafhankelijke partij een architectuurtoets te laten doen. Dit kan aan de hand van een architectuur assessment methode [ATAM, SAAM].

Een andere mogelijkheid is om bij de stakeholders na te gaan of ze hun belangen in de architectuur vertegenwoordigd zien of snappen waarom dat niet het geval is. In het bijzonder moeten:

- de vertegenwoordigers van de opdrachtgever de prioritering en afweging van de belangen accorderen.
- de ontwikkelaars en ontwerpers de architectuur toepassen.
- de programma- en projectmanagers de architectuur omzetten in een planning.

Een relatief eenvoudige manier is om de aan de architect te vragen hoe het staat met de architectuur. Managers die verstand van architectuur hebben kunnen dat gericht doen. Voor managers die dat in mindere mate hebben, zijn een aantal basisvragen voor handen. Het doel van deze vragen is een beeld te krijgen van de kwaliteit van de architectuur. De basisvragen aan de architect zijn:

1. Wie zijn de stakeholders en wat zijn hun vijf belangrijkste concerns met betrekking tot het systeem?
2. Wat is de belangrijkste rol die elke stakeholder speelt bij de totstandkoming van de architectuur?
3. Wat is de afbakening van de architectuur? In het bijzonder met betrekking tot:
 - a. Toepassingsgebied
 - b. Technologie
 - c. Ontwikkel- en beheerproces
 - d. Levensduur
4. Wat zijn de 5 belangrijkste belangenafwegingen (van tegenstrijdige belangen) in de architectuur en hoe zijn deze beheerd, uitgedragen en bewaakt?
5. Wat zijn de 5 belangrijkste systeemkwaliteiten waar de architectuur een oplossing voor biedt? Welke stakeholderbelangen zijn daarmee gediend?
6. Wat zijn de belangrijkste issues en hoe zijn deze geadresseerd? Noem de 5 belangrijkste businessissues met betrekking tot het systeem, de 5 belangrijkste ontwerpissues en 5 belangrijkste realisatie-issues.
7. Op welke manier is de consistentie van de 5 belangrijkste eisen en hun oplossing aangetoond?
8. Welke architectuurviews zijn er om te redeneren, uit te leggen, uit te rollen, te beschrijven, etc? Aan welke stakeholders zijn deze gericht?
9. Op welke manier zijn de stakeholders op de hoogte gebracht van de manier waarop de architectuur hun eisen (niet) inwilligt en wat er van hun verwacht wordt / welke acties zij moeten ondernemen?
10. Wat is er zo geweldig aan de architectuur wat ik niet moet vergeten?

De architect moet op alle bovenstaande vragen antwoord kunnen geven of hij moet kunnen zeggen waarom hij geen antwoord heeft. Eventueel worden de antwoorden besproken met verschillende stakeholders, zodat de business manager de geldigheid kan toetsen.

Over de auteurs

Robert Deckers is als Senior Software Architect en architectuur consultant werkzaam bij Sogeti. Naast zijn kennis en ervaring op architectuurgebied heeft hij veel ervaring in informatieanalyse, requirements management, modelleertechnieken, model driven development en architectuurinnovatie. Hij is te bereiken via robert.deckers@sogeti.nl

Ruud Steeghs heeft uitgebreide ervaring als Software Architect van complexe IT-systemen. Daarnaast helpt hij organisaties bij het toepassen van (software) architectuur. Hij is bereikbaar via ruud.steeghs@sogeti.nl

Referenties

- [Bass] Len Bass, et al. Software Architecture in Practice, 2nd Edition, ISBN 0-321-15495-9
- [Rozanski] Nick Rozanski, Eoin Woords. Software Systems Architecture, Working with views and perspectives, ISBN 0-321-11229-6
- [Bosch] Jan Bosch. Design and Use of Software Architectures: Adopting and Evolving a Product-Line Approach, ISBN 0-201-67494-7
- [DYA] DYA, snelheid en samenhang in business- en ICT-architectuur, ISBN10 9072194624
www.dya.info
- [WP SW] <http://nl.wikipedia.org/wiki/Software>
- [ATAM] Clements, et al. Evaluating Software Architectures: Methods and Case Studies, ISBN 020170482X
- [SAAM] Clements, et al. Evaluating Software Architectures: Methods and Case Studies, ISBN 020170482X
- [ARM] http://www.dya.info/Home/dya_suite/dya_software/publicaties.jsp