Evolueren met portfoliomonitoring

Toolkit analyseert en visualiseert softwaresystemen

Door gebrek aan inzicht beschouwen bedrijven hun softwareportfolio vaak als weerbarstige en onbeheersbare doos van Pandora. Met de juiste informatie over de moeilijkheden, afhankelijkheden en de onderhoudskosten kan dit veranderen. De auteurs ontwikkelden een toolkit voor softwareanalyse en geven een handleiding voor monitoring.

Tobias Kuipers en Joost Visser

Softwareportfoliomonitoring (spm) is een toolgebaseerde methodologie die op een efficiënte manier kwaliteits-, omvangs- en interrelatiegegevens afleidt uit de broncode van softwaresystemen. Aangezien de gegevens worden afgeleid uit systemen die vele miljoenen regels broncode bevatten, en ze op cio-niveau gepresenteerd moeten worden, bevat de methodologie een aantal stappen waarin de gegevens op een hoog niveau worden geaggregeerd. Alleen de relevante gegevens worden getoond, en wel op zo'n manier dat de niet-technische manager geïnformeerde beslissingen kan nemen over de softwaresystemen van zijn bedrijf.

De Software Analyse Toolkit (Sat), waarop de spm-methodologie is gebaseerd, bestaat uit een generiek raamwerk voor softwareanalyse en -visualisatie, en een aantal specifieke instanties van dat raamwerk. Een zeer eenvoudig uitbreidbare component aan de ene kant van het systeem kan alle mogelijke typen broncode en gerelateerde documenten analyseren. De resultaten worden opgeslagen in een generiek datamodel, voorzien van een tijdsaanduiding, zodat later verschillende meetresultaten met elkaar vergeleken kunnen worden. Aan de andere kant van het raamwerk worden de gegevens uit het datamodel (al dan niet als animatie) gepresenteerd.

Analyse

Voorbeelden van analyses zijn de detectie van 'klonen' in de broncode: delen van de code die zo sterk op elkaar lijken dat ze zonder al te veel moeite verwijderd kunnen worden. Ook analyseert de tool voor diverse talen en technologieën afhankelijkheden tussen modules, zowel via aanroepen als data-uitwisseling. Een laatste voorbeeld is het automatisch schatten van het aantal functiepunten van een systeem, met de *backfiring* methode.

Deze voorbeelden zijn, net als alle andere analyses, generiek en schaalbaar. De hoeveelheid broncode in een softwareportfolio varieert tussen de 1 en 100 miljoen regels. Het verwerken van deze hoeveelheden code mag niet meer dan een paar uur duren, en als gevolg daarvan moet de complexiteit van de analysealgoritmes niet te groot zijn. In die zin moeten de analyses

Samenvatting

De toolkit van de auteurs geeft inzicht in de broncode van alle softwaresystemen in een organisatie. Een maandelijkse analyse biedt gevisualiseerde basisgegevens voor projectleiders. In de kwartaalanalyse evalueren en interpreteren monitoringexperts de gegevens voor het ict-management. En in een softwarejaarverslag vatten ze de resultaten samen voor het algemene management.

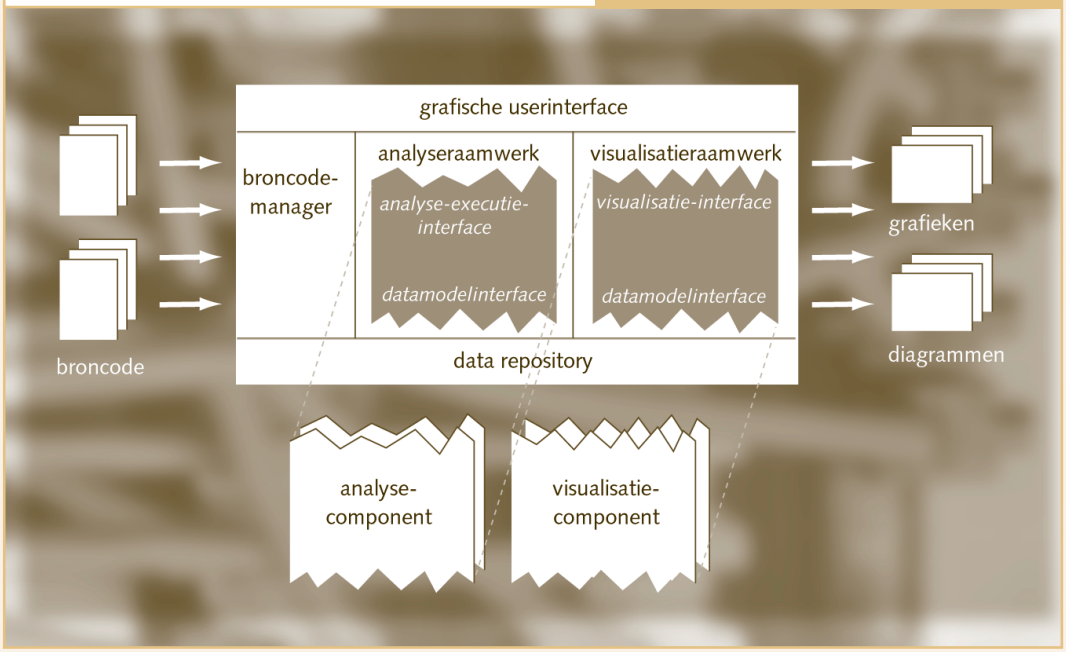
dus schaalbaar zijn. De analyses die de tool uitvoert zijn bovendien in hoge mate generiek, in de zin dat de analysealgoritmes werken op verschillende programmeertalen. Was dit niet het geval, dan zou het te veel tijd kosten om de analyses aan te passen aan gewijzigde inzichten, of aan nieuw ontdekte situaties in de softwareportfolio.

In het voorbeeld is sprake van complexiteit van programmamodules en databaseafhankelijkheden. Tijdens de analysefase rekent de tool voor alle modules in het systeem de cyclomatische complexiteit uit. Voor Java-code rekent hij daartoe het aantal mogelijke executie paden per methode uit. Voor Cobol is dit het aantal executiepaden per programma. Voor beide talen is het aantal executiepaden het aantal jumps dat mogelijk is binnen een methode of een programma. Als er bijvoorbeeld 15 if-statements in een methode voorkomen, is het maximaal aantal te testen paden 16. In de praktijk blijkt dat pro-

Hypothese

Met de tool kan een expert bijvoorbeeld vaststellen dat een aantal modules een specifieke complexiteitsdrempel overschrijdt. Zijn hypothese is dat deze modules samen een aantal gerelateerde functies implementeren, zonder dat die goed gescheiden zijn. Hij stelt vast dat de databaseafhankelijkheden van deze modules zijn hypothese ondersteunen (doordat alle modules dezelfde verzameling databasetabellen gebruiken). Tot slot kan hij nog besluiten een deel van de code in detail te bekijken, totdat hij ervan overtuigd is dat de hypothese waar is.

Architectuur van de softwareanalysetoolkit 1



grammeurs meer fouten maken en dat de code steeds lastiger begrijpen wordt als deze waarden hoger zijn.
Elke programmeertaal die de tool ondersteunt, heeft zijn eigen analyse die deze berekening kan maken. In geval van Java en Cobol gebeurt dit door de code te ontleden en een analyse te maken die over de verkregen parseboom loopt, terwijl het aantal branchstatements wordt geteld.
Ook de databaseafhankelijkheden worden in de analysefase uit de code gehaald. Uit het voorbeeld blijkt dat de databasetabellen die in een module gebruikt worden relevant zijn. Deze verwijzingen worden in de analysefase uit de code gehaald. Afhankelijk van het programmeeridioom dat een organisatie gebruikt, is daar een speciale analyse voor nodig. Als database-access plaatsvindt met embedded SQL, dan kan voor Java en Cobol dezelfde analyse worden gebruikt. Vaak is er echter een tussenlaag die de daadwerkelijke SQL afschermt. In Java wordt vaak een *object-relational mapping* gebruikt. Dan moet de analyse in de gaten houden of de huidige module objecten gebruikt die in de object-relationalmapping genoemd worden, die komen dan immers ook in de database terecht. Om het generieke karakter van de analyses en de visualisaties te faciliteren is er een algemeen datamodel waar alle gegevens in worden opgeslagen.

Visualisatie

Visualisatie is de andere belangrijke component van het Sat-raamwerk. Er zijn twee typen visualisaties: diagrammen en grafen. Ook deze zijn generiek. Het raamwerk kent een aantal modules die verschillende typen grafieken kunnen tonen. Afhankelijk van de uitgevoerde analyses zijn specifieke visualisaties beschikbaar. Sommige visualisaties laten het verloop van meetgegevens over tijd zien. Met de figuren kan de monitoringexpert vaststellen of bijvoorbeeld de complexiteit van de modules niet boven een specifieke drempelwaarde uitkomt. Daarnaast is het goed te zien of de complexiteit voor specifieke modules aan het stijgen of aan het dalen is.
Om terug te komen op het eerder genoemde voorbeeld: als de monitoringexpert vaststelt dat de complexiteit te hoog is opgelopen, kan hij om de oorzaak te achterhalen bijvoorbeeld kijken naar de databaseafhankelijkheden van de verschillende modules. Dat gebeurt met behulp van de interactieve graafvisualisatiecomponent van Sat. Met deze component kan de monitoringexpert de getoonde graaf interactief manipuleren om beter inzicht te verwerven. Zo kunnen bijvoorbeeld alleen de databasetabellen waaraan gerefereerd wordt vanuit bepaalde modules zichtbaar zijn, of worden alleen de lees- of juist de schrijfacties getoond. Als een groot aantal modules ongeveer dezelfde verzameling tabellen benaderen wordt dat snel inzichtelijk gemaakt. De expert kan vervolgens voorstellen doen om het aldus ontstane probleem op te lossen.

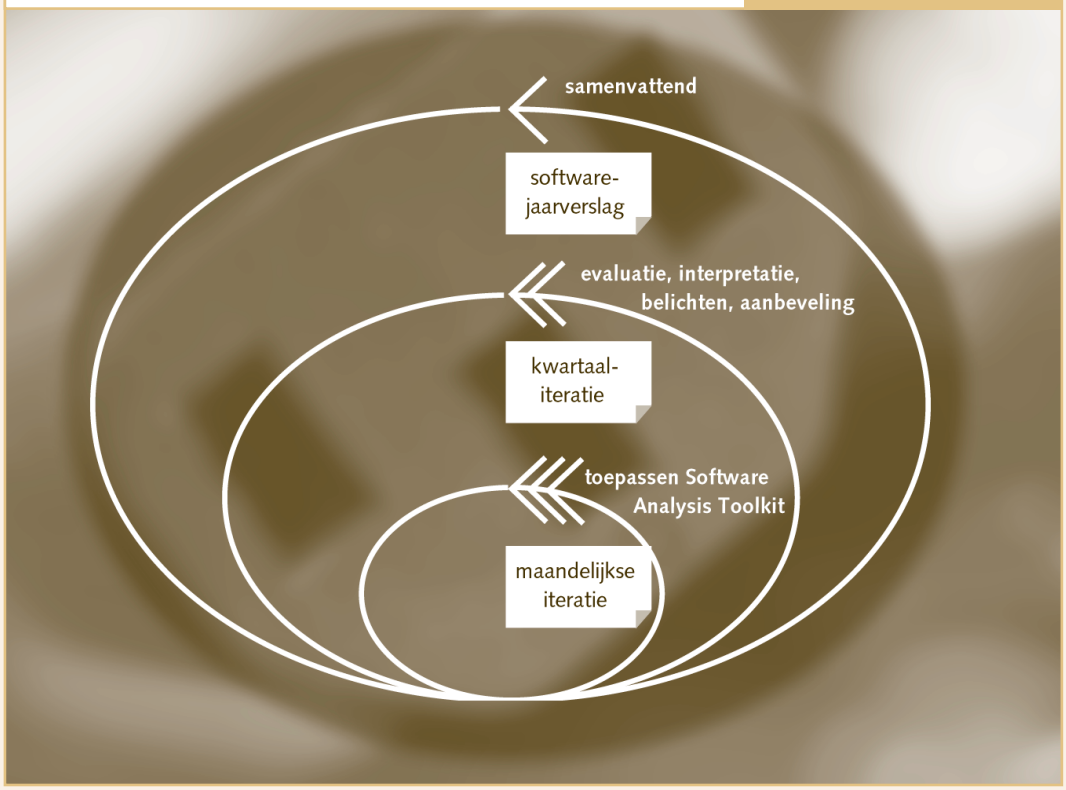
De methodologie

De methodologie bestaat uit drie geneste iteraties, zoals te zien in figuur 2. Korte iteraties worden weergegeven door een kleine ellips, langere iteraties met een grotere ellips. Voor elke iteratie is aangegeven wat er wordt opgeleverd, en welke acties er nodig zijn om tot die oplevering te komen. Elke grotere iteratie neemt de opleveringen van de onderliggende iteraties als invoer. In dit artikel gaan we uit van een kortste iteratie van een maand. In de praktijk wordt afhankelijk van de situatie bepaald wat de geschiktste lengte is.

Maandelijkse iteratie

De binnenste iteratie wordt maandelijks doorlopen. In deze iteratie wordt de Software Analyse Toolkit toegepast op de softwareportfolio. De uitvoer hiervan is een grote hoeveelheid basisgegevens, waaronder metrieken, afhankelijkheidsinformatie, en bijvoorbeeld overtredingen van programmeerstandaarden of -conventies. Al deze gegevens worden in een *repository* opgeslagen. Hieruit wordt een rapport gegenereerd dat de feiten op een begrijpelijke manier toont. De gegevens zijn gegroepeerd en gefilterd en gevisualiseerd in diagrammen en grafieken die aan de informatiebehoefte van de projectleiders voldoen. In het ideale geval is de maandelijkse applicatie van Sat een volledig geautomatiseerd proces. In de praktijk is er echter meestal een menselijke stap in het proces nodig. Het op te leveren product van een maanditeratie, de *deliverable*, is dus een technisch zeer gedetailleerde rapportage over de huidige toestand van de softwareportfolio. Deze rapportage wordt beschikbaar gesteld aan de projectleiders en andere technische staf in de

De spm-methodologie bestaat uit drie geneste iteraties 2



vorm van een hyperdocument, een collectie gelinkte webpagina's met grafen en grafieken.

Kwartaaliteraties

Elke drie maanden worden alle gegevens uit de maandelijkse iteraties bij elkaar gevoegd, geïnterpreteerd en geëvalueerd door monitoringexperts. Verder worden de gegevens gerelateerd aan de businessdoelen zoals het management van de organisatie die heeft geformuleerd (via interviews en eventueel workshops worden deze doelen boven water gehaald). Het ict-management krijgt de bevindingen gepresenteerd, samen met concrete voorstellen voor een reactie. In de interpretatie van de meetgegevens zet de monitoringexpert verschillende selecties van de gegevens tegen elkaar uit, om bijvoorbeeld trends, correlaties en uitzonderingen waar te nemen.
In de evaluatie van de gegevens spreekt de monitoringexpert een waardeoordeel uit over de softwareportfolio. Dit oordeel is niet gebaseerd op smaak of voorkeur, maar op gepubliceerde kwaliteitsstandaarden en industriegemiddelden. Hij maakt hiervoor gebruik van geaccepteerde kennis over software-engineering.
De evaluatie en interpretatie van de technische gegevens zijn, samen met het boven water krijgen van de ict-gerelateerde businessdoelstellingen van essentieel belang voor het belangrijkste deel van de kwartaaliteratie: het opstellen van aanbevelingen. Deze aanbevelingen kunnen van verschillende niveaus zijn. Het kunnen zeer gedetailleerde korte-termijnaanbevelingen zijn, bijvoorbeeld om een specifieke interface opnieuw te ontwerpen, specifieke gegevens van een hiërarchische naar een relationele database te migreren, of een component van een derde partij te upgraden. Ook kunnen aanbevelingen algemener zijn, bijvoorbeeld om twee functioneel vrijwel gelijke maar technisch totaal verschillende systemen te integreren, of het procedurele karakter van de objectgeoriënteerde gedeelten van de portfolio te verminderen.
In sommige gevallen zijn de monitoringgegevens niet voldoende om een concrete aanbeveling te doen. In zulke gevallen is een *software risk assessment* nodig (Van Deursen & Kuipers, 2003). Zo'n assessment is een diepgravend onderzoek naar een specifiek systeem, gebaseerd op broncode-analyse en interviews met betrokkenen bij dat systeem. Het leidt tot een beknopt rapport met concrete en gedetailleerde aanbevelingen.
Het is duidelijk dat de activiteiten die worden uitgevoerd door de monitoringexperts niet alleen technische expertise veronderstellen, maar ook managementconsultancy-expertise. Deze combi-

natie van expertises is niet erg gebruikelijk, maar is essentieel om ervoor te zorgen dat de aanbevelingen zowel technisch correct zijn als gerechtvaardigd vanuit een commercieel standpunt.
De deliverable van de kwartaaliteratie is een rapport voor het ict-management. De monitoringexperts presenteren dit rapport persoonlijk, zodat zij eventuele vragen en wijzigingen in strategie direct kunnen bespreken en meenemen in de volgende iteratie.

Jaarlijkse iteratie

Eenmaal per jaar worden alle deliverables van het jaar samengevat in het softwarejaarverslag. De doelgroep voor dit jaarverslag is het algemene management van de organisatie. Het jaarverslag beschrijft ook de bereikte resultaten op ict-gebied en de gestelde doelen in lekentermen. Indien beschikbaar worden de resultaten in het softwarejaarverslag gerelateerd aan financiële gegevens.

Organisatorische context

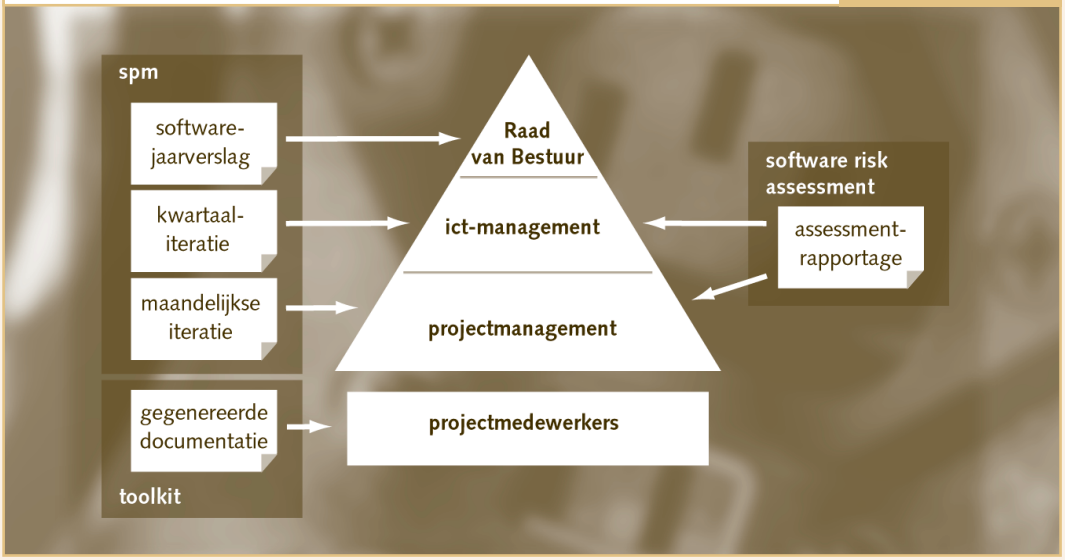
De diverse rapportages die tijdens spm worden geproduceerd, worden aan verschillende managementlagen binnen de organisatie gepresenteerd. In figuur 3 is dat weergegeven. De maandelijkse rapportages worden verstrekt aan projectleiders. De kwartaalrapportages worden gepresenteerd aan het ict-management, namelijk de cio en zijn directe ondergeschikten. Het softwarejaarverslag is bedoeld voor het algemene management van de organisatie.

Invoering

We hebben spm zo ontworpen dat het niet verstorend werkt voor een organisatie. Een organisatie kan hier mee starten zonder dat de organisatie hoeft te veranderen. Sterker nog, spm is een instrument om verandering op een geleidelijke en beheersbare manier te faciliteren, niet een instrument dat pas werkt als de organisatie al veranderd is.
Om spm op te zetten en in te regelen voor een specifieke organisatie raden wij aan een periode van drie maanden te reserveren. In deze periode moeten er drie zaken geregeld worden. Ten eerste moet er *commitment* zijn vanuit het management om spm uit te rollen. De organisatie moet bereid zijn om gegeven de resultaten van spm bepaalde situaties te veranderen. Ten tweede moet er op het terrein van softwarelogistiek het een en ander geregeld worden om periodieke metingen aan de systemen mogelijk te maken. Tot slot moet er een modus worden afgesproken om regelmatige terugkoppeling tussen de diverse partijen (verschillende niveaus van management en de monitoringexperts) naadloos te laten verlopen.

Softwareportfoliomonitoring en de relaties met deliverables en managementniveaus

3



Literatuur

Deursen, A. van, & T. Kuipers (2003). Source-based software risk assessment. In *Proceedings 21 International Conference on Software Maintenance (ICSM'03). IEEE Computer Society*, 2003.

Tobias Kuipers
is chief technical officer bij de Software Improvement Group. E-mail: Tobias.Kuipers@software-improvers.com.

Joost Visser
is post-doctoraal onderzoeker aan de Informaticafaculteit van de Universiteit van Minho in Portugal. E-mail: Joost.Visser@di.uminho.pt.