BVH

Software Risk Assessment Rapport t.b.v. vts Politie Nederland

25 juni 2008

Software Improvement Group

Arent Janszoon Ernststraat 595-H
NL-1082 LD Amsterdam
t +31 (0)20 314 09 50
f +31 (0)20 314 09 55
info@sig.nl
www.sig.nl

Disclaimer

Alle conclusies in dit rapport zijn gebaseerd op door de opdrachtgever aangeleverde broncode en aanvullende informatie. De juistheid en volledigheid van deze onderliggende gegevens zijn door de Software Improvement Group gecontroleerd, maar zijn desondanks in uiterste instantie de verantwoordelijkheid van de opdrachtgever.

Het onderzoek dat in dit rapport is beschreven is uitgevoerd in opdracht van Marcel Huting van vts Politie Nederland.

Het rapport is geschreven door Joost Visser en Pieter Jan 't Hoen van de Software Improvement Group.

A. J. Ernststraat 595-H
1082 LD Amsterdam
The Netherlands

Management Samenvatting

De Software Improvement Group heeft een Software Risk Assessment uitgevoerd op het BVH systeem om de kwaliteit van de softwareontwikkeling te beoordelen. Bij de assessment zijn de broncode en documentatie van het systeem in aanmerking genomen, alsmede de organisatiestructuur en bemensing van het project. De informatievergaring voor dit onderzoek heeft plaatsgevonden tussen 18 februari en 25 maart 2008.

SIG heeft vastgesteld dat het BVH systeem zeer omvangrijk is en een hoge diversiteit kent in gebruikte technologieën. Met name de koppelingen tussen systeemonderdelen en met externe systemen zijn talrijk en op vele verschillende wijzen gerealiseerd.

Het BVH project kent als risico's dat de installatie, het beheer, en het onderhoud van het BVH systeem bijzonder kostbaar en arbeidsintensief zijn. Door de complexe interacties tussen de systeemonderdelen en met externe systemen, en door de grote technologische diversiteit is het onduidelijk hoe goed BVH te configureren en installeren is in nieuwe verzorgingsgebieden of aan te passen is aan mogelijke nieuwe eisen. Dit blijkt bijvoorbeeld uit het feit dat de installatie van het systeem in een verzorgingsgebied uitgevoerd door systeem experts 9 werkdagen doorlooptijd in beslag neemt.

De kwaliteit van de systeemonderdelen, wanneer los beschouwd, is met enkele uitzonderingen redelijk tot goed. Echter, het systeem als geheel is van zeer lage kwaliteit en slecht onderhoudbaar. De belangrijkste zwakte is de algehele architectuur met verouderde technologie in het centrum en hoge diversiteit in satellietonderdelen en koppelingen. Dit komt ook tot uitdrukking in de documentatie van het systeem, die per systeemonderdeel is uitgewerkt, terwijl overzichtsdocumentatie over het gehele systeem ontbreekt.

Het BVH project kenmerkt zich in sterke mate door de problematiek van het op elkaar afstemmen van de verschillende systeemonderdelen en het herhaald operationeel maken van het gehele systeem. Echter, de huidige organisatiestructuur, inclusief software bouwteam, van het BVH project is voornamelijk gericht op ontwikkeling van de verscheidene deelsystemen. De taak van het bewaken van de algehele software architectuur, alsmede van het ontwikkelen, harmoniseren, en testen van de installatieprocedures zijn onvoldoende belegd.

In overeenstemming met het bovenstaande volgen de antwoorden op de hoofdvraag en bijbehorende subvragen van deze assessment.

- De kwaliteit van de softwareontwikkeling van BVH is matig. Enerzijds is de matige kwaliteit bij aanvang van het project van de verschillende systeemonderdelen niet verslechterd en in een aantal gevallen zelfs verbeterd. Anderzijds is de kwaliteit van enkele nieuw ontwikkelde systeemonderdelen ook niet bijzonder hoog.
- De ontwerpdocumentatie is aanwezig en per onderdeel uitgewerkt, maar overzichtsdocumentatie voor het systeem als geheel ontbreekt. De grootste satelliet, AVI, is van lage, doch iets verbeterde kwaliteit. De nieuw ontwikkelde satelliet DagRap en de sterk in omvang toegenomen ZiF satelliet zijn van lage

kwaliteit. De ZoZ satelliet, buiten het project ontwikkeld in afwijkende technologie, is van lage kwaliteit.

- In de organisatorische opzet van BVH is de rol van Software Architect onvoldoende ingevuld en is onvoldoende nadruk gelegd op integratie en installatie activiteiten.
- Qua bezetting is voldoende kwaliteit en capaciteit aanwezig om systeemonderdelen van hoogwaardige kwaliteit op te leveren. Echter, de inspanningen op de verschillende onderdelen behoeven betere coördinatie om de kwaliteit van het systeem als geheel te waarborgen. Hier ligt een taak voor de software architect.
- De condities zijn grotendeels aanwezig om het software ontwikkelingsproces kwalitatief juist uit te voeren. Uitzonderingen hierop, naast de afwezigheid van een sofware architect, zijn gebrek aan gestroomlijnd versie beheer, geautomatiseerde regressietestsuites, en centrale bewaking van code kwaliteit. Ook hebben de condities bij aanvang van het project (lage code kwaliteit, ontbrekende documentatie, ontbrekende tests) de uitvoer van het project bemoeilijkt.

Voor beheersing van de kosten en het terugbrengen van de risico's worden een aantal aanbevelingen gedaan, zowel op de korte termijn als voor de verdere toekomst. De belangrijkste aanbevelingen op de korte termijn zijn versterking van toezicht op de algehele software architectuur door het aanstellen van een software architect, en de verankering in de organisatiestructuur van de activiteiten ten behoeve van het herhaalbaar maken van deployment. De aanbevelingen op middellange termijn zijn gericht op het reduceren van diversiteit in technologieën en complexiteit in software architectuur, bijvoorbeeld door herontwikkeling van koppelingen of bepaalde deelsystemen. Op langere termijn zal het van belang zijn om een strategie te bepalen voor vervanging en/of renovatie van het BVH systeem.

Inhoudsopgave

1 Inleiding

Tegen de achtergrond van het streven naar landelijk gedeelde systemen heeft vts Politie Nederland (vtsPN) in het voorjaar van 2007 het project Basis Voorziening Handhaving (BVH) gestart. BVH is een landelijke applicatie die een van de belangrijkste processen binnen de politie ondersteunt. BVH is een complex systeem dat is opgebouwd uit een verzameling van verschillende systeemonderdelen waartussen gegevensuitwisseling plaatsvindt.

Bij een eerste uitrol van BVH zijn een aantal onverwachte problemen zichtbaar geworden. Teneinde duidelijkheid te krijgen in de ontstane situatie heeft vtsPN eind 2007 SIG gevraagd een oordeel te geven over de kwaliteit van het testproces. Dit onderzoek is inmiddels afgerond en de resultaten zijn bekend bij de politie (Testadvies BVH t.b.v ISC, vts Politie Nederland, Software Improvement Group, 15 januari 2008).

1.1 Vraagstelling

In het verlengde van het bovenstaande heeft vtsPN aan SIG tevens een oordeel gevraagd over de kwaliteit van de software ontwikkeling van BVH. Hierbij zijn vier subvragen te onderscheiden:

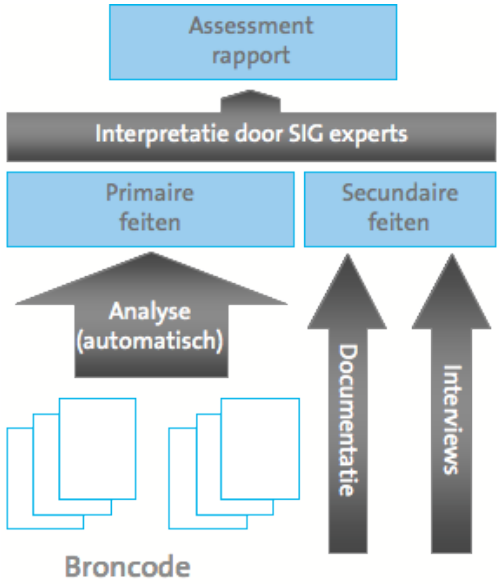
1. Wat is de kwaliteit van de ontwerpdocumentatie en de software van de verschillende onderdelen waaruit de BVH bestaat?
2. Op welke wijze is het realiseren van BVH organisatorisch opgezet om te borgen dat de verschillende onderdelen in samenhang functioneren?
3. Is per BVH onderdeel qua bezetting de juiste kwaliteit en capaciteit aanwezig om kwalitatief hoogwaardige software op te leveren?
4. Zijn de condities aanwezig om het software ontwikkelingsproces kwalitatief juist uit te kunnen voeren?

Om tot het gevraagde oordeel te komen, inclusief beantwoording van de geformuleerde subvragen, heeft SIG een Software Risk Assessment uitgevoerd op BVH.

1.2 Methodiek van Software Risk Assessments

Tijdens een Software Risk Assessment worden op de broncode van een systeem analyses uitgevoerd met behulp van tools van SIG. De gegevens die uit deze analyses naar voren komen, worden gekoppeld aan informatie uit documentatie en gesprekken met systeemexperts en stakeholders. De specialisten van SIG gaan aan de hand van deze informatie het systeem verder onderzoeken om risico's te identificeren. Hierbij wordt vergelijkende informatie uit eerdere assessments toegepast. Het proces is schematisch weergegeven in Figuur 1.

Op deze manier wordt een objectief en onafhankelijk technisch inzicht in een systeem verkregen, waardoor gerichte adviezen kunnen worden geformuleerd. De functionele kant van een systeem wordt in een Software Risk Assessment niet behandeld.



Figuur 1: Overzicht van de activiteiten en deliverables van een Software Risk Assessment.

Dit rapport presenteert de resultaten van het Software Risk Assessment. Hierbij wordt uitgegaan van het ISO 9126 model voor de kwaliteit van software. Dit model definieert met betrekking tot onderhoudbaarheid van software vier aspecten die invloed hebben op onderhoudbaarheid. Dit zijn *analysability*, *changeability*, *stability*, en *testability*. Het model geeft ook definities over hoe deze aspecten te meten. Analyseerbaarheid kan bijvoorbeeld gemeten worden door statistieken bij te houden over hoe lang een ontwikkelaar nodig heeft om fouten op te sporen.

	High level architecture	Modularisation	Separation of concerns	Complexity	Method length	Duplication	Volume	Exception handling	Test quality	Process
Analysability		X	X		X	X	X			
Changeability	X		X	X		X				
Stability								X	X	X
Testability			X	X	X				X	

Tabel 1 Het generieke model van de Software Improvement Group over de relatie tussen de ISO 9126 karakteristieken voor onderhoudbaarheid en metingen uit statische broncode en documentatie analyse.

Het is ook mogelijk om deze informatie af te leiden uit de directe analyse van broncode. In Tabel 1 staat een generiek model, ontwikkeld door de Software Improvement Group, waarin de ISO 9126 karakteristieken worden gerelateerd aan statische analyse van broncode en bijbehorende documentatie. In de matrix staan in de eerste kolom de ISO 9126 attributen. Op de bovenste rij staan een tiental metingen. Een kruis in de matrix geeft aan dat een meting meeweegt tot een specifiek karakteristiek. Zo staat er bij-

voorbeeld een kruis bij volume en analyseerbaarheid. Dat betekent dat de omvang van een systeem invloed heeft op de analyseerbaarheid van software.

In deze rapportage staat het bovengenoemde algemeen toepasbare model centraal in de beoordeling van het systeem. De beoordeling wordt voorafgegaan door een samenvattende beschrijving van de bevindingen. De beoordeling wordt compleet gemaakt met een expliciete definitie van de risico's en de aanbevelingen.

1.3 Onderzochte bronnen

De Software Risk Assessment is uitgevoerd op de broncode en documentatie van het BVH systeem, en door middel van meerdere interviews. De informatievergaring voor dit onderzoek heeft plaatsgevonden tussen 18 februari en 25 maart 2008.

1.3.1 Gehouden Sessies

De onderstaande tabel geeft een overzicht van de interview sessies en andere contactmomenten die in het kader van het onderzoek hebben plaatsgevonden.

Datum	Sessie
18-02-2008	Kickoff meeting
26-02-2008	Interview verkenning organisatiestructuur BVH
28-02-2008	Technisch interview BVH core
28-02-2008	Technisch interview satellieten
10-03-2008	Technisch interview GSH
10-03-2008	Technisch interview Xpol en externe koppelingen
14-03-2008	Technisch interview Xpol GUI (Seagull technologie)
18-03-2008	Technisch interview AVI
20-03-2008	Interview integratie activiteiten
20-03-2008	Bespreking organisatiestructuur en bemensing
25-03-2008	Strategisch interview
26-03-2008	Voorbespreking eindpresentatie

Tabel 2, Overzicht sessies.

Tijdens bovengenoemde sessies en andere contactmomenten is met de volgende personen gesproken:

- Jos Bakker (Seagull)
- Newal Chierandjoe
- Bart Dijkstra
- Arjan Emck
- Benny Gadovnik

- Kees van Gemert
- Martin van Gunst
- Marcel Huting
- Hans Jellema
- Jurgen Kamphuis
- Henk Kouwenhoven
- Theo Oldehinkel
- Rene Terhorst (Seagull)
- Arie Verheul
- Koos van der Wulp

1.3.2 Onderzochte broncode en documentatie

Het BVH systeem bestaat uit meerdere deelsystemen. Voor deelsystemen die al bestonden bij aanvang van het project zijn twee versies van de broncode en documentatie opgevraagd, corresponderend met de aanvangssituatie (nulmeting) en met de huidige situatie. Voor deelsystemen die nieuw zijn ontwikkeld binnen het project is alleen broncode en documentatie voor de huidige situatie aangeleverd.

Tabel 3 geeft een overzicht van de aangeleverde broncode versies per systeemonderdeel (zie de appendix voor een uitgebreider overzicht).

Systeem onderdeel	Beschikbaar gestelde versies
BVH core	Nulmeting en huidige versie
Xpol Correctie Tool	Nulmeting en huidige versie
AVI	Nulmeting en huidige versie
Zoeken in Formulieren	Nulmeting en huidige versie
Zicht op Zaken	Nulmeting en huidige versie
Afspraak op Locatie	Alleen huidige versie, halverwege project toegevoegd
Dag Rapportage	Alleen huidige versie, nieuwbouw
ITX4Word	Alleen huidige versie, nieuwbouw
Xpol GUI	Alleen huidige versie, nieuwbouw

Tabel 3, Overzicht van aangeleverde broncode.

De geleverde documentatie is zeer omvangrijk en bestaat uit meer dan 1400 MS-Word documenten in verschillende stadia van afronding, 98 Excel documenten, en 41 losse Visual Studio diagrammen. Tabel 4 bevat een overzicht van de belangrijkste documenten per BVH onderdeel. Daarnaast geeft het document “Beheerscan_samenvatting_(1[1].0).doc” een overzicht van de BVH applicatie onderdelen. Separaat is een kopie van de UITrol documentatie geleverd van Fase 1 tot en met 3: een A4 ordner met circa 400 bladzijden, waarop met de hand aantekeningen zijn toegevoegd. Dit document is gaande de huidige assessment tot stand gebracht als product van de activiteiten van het UIT team.

Systeem onderdeel	Beschikbare documentatie
BVH core	Release notes, systeemdocumentatie
Xpol Correctie Tool	Beheerscan, FO nulmeting en huidige versie
AVI	Beheerscan, FO nulmeting en huidige versie
Zoeken in Formulieren	Beheerscan, FO nulmeting en huidige versie
Zicht op Zaken	FO nulmeting en huidige versie
Afspraak op Locatie	Quickscan document
Dag Rapportage	Beheerscan, FO nulmeting en huidige versie
ITX4Word	Release notes
Xpol GUI	BVH GUI Systeemdocumentatie

Tabel 4, Overzicht van de beschikbare documentatie

1.4 Structuur van dit rapport

Dit rapport vormt de weerslag van de resultaten van de Software Risk Assessment. De technische bevindingen zijn gebaseerd op analyse van de broncode, bestudering van de documentatie, en interviews met betrokkenen. De beschreven bevindingen komen voornamelijk voort uit de broncode zelf en zijn derhalve technisch van aard. Er is gestreefd naar een mate van detail waardoor de bevindingen concreet blijven. Voor de technisch minder geïnteresseerde lezer is een management samenvatting bijgesloten.

Dit rapport heeft de volgende structuur. In hoofdstuk 2 wordt een beschrijving van het systeem gegeven. Daaruit volgt de beoordeling van het systeem met de identificatie van de risico's in hoofdstuk 3. Hoofdstuk 3 behandeld ook in het kort de doorgevoerde wijzigingen in de aangeleverde broncode en de invloed op de beoordelingen. De Organisatie structuur en de bemensing wordt behandelt in hoofdstuk 4. De beantwoording van de onderzoeksvragen vindt plaats in hoofdstuk 5. De conclusies en aanbevelingen zijn weergegeven in hoofdstuk 6. In de Appendix A staan de waarnemingen beschreven.

2 Beschrijving BVH

Dit hoofdstuk beschrijft het BVH systeem op basis van de interviews, documentatie en analyse van de broncode. De beschrijving van het systeem en het high level design zijn in dit hoofdstuk opgenomen. De details van de waarnemingen aan het BVH systeem staan in Bijlage A. De beoordeling van de waarnemingen is te vinden in het volgende hoofdstuk.

2.1 Beschrijving van het systeem

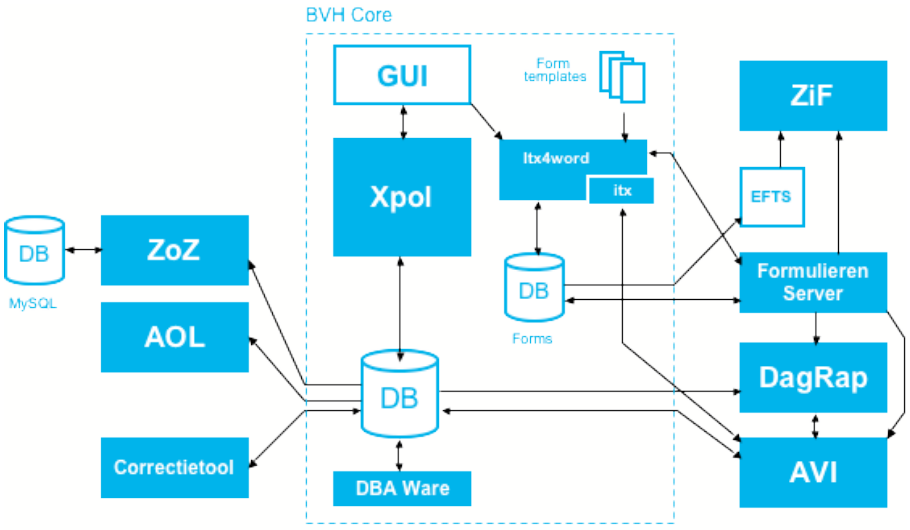
De ontwikkeling van de Basis Voorziening Handhaving (BVH) past in het streven om te komen tot landelijk gedeelde informatie systemen voor de diverse politie korpsen. In maart 2007 is gestart met het BVH project dat tot doel heeft om het bestaande handhavingsysteem Xpol, aangevuld met diverse satellietapplicaties en een grafische user interface, geschikt en beschikbaar te maken voor landelijk gebruik. Inmiddels is het project in de initiële fase van uitrol.

2.2 High level design

De applicatie is onderzocht op high level design. Het high level design beschrijft de applicatie hoofdlijnen. Hieronder volgt de interne en externe architectuur, en de relatie met andere applicaties. In Appendix A worden de onderdelen modularisatie en separation of concerns van het high level design beschreven.

2.2.1 Interne architectuur

De huidige interne architectuur van BVH is schematisch weergegeven in Figuur 2.



Figuur 2, De interne architectuur van BVH.

Het BVH systeem is gebaseerd op de bestaande handhavingsapplicatie Xpol (nu ook BVH-core genoemd). Xpol is ontwikkeld in de vierde-generatie programmeertaal Accell en maakt gebruik van Sybase database technologie. Xpol biedt een karaktergeoriënteerde user interface (niet-grafische schermen).

In het BVH systeem is aan Xpol een grafische user interface (GUI) toegevoegd, gebruikmakend van zogenaamde *screen-scraping* technologie van Seagul. Er bestaat een redelijk directe correspondentie tussen de nieuwe grafische schermen en de bestaande niet-grafische schermen van de onderliggende Xpol applicatie. Als onderdeel van de GUI is een op MS-Word georiënteerde tekstverwerker opgenomen (ITX4Word). De landelijke standaard formulierenset is compatible gemaakt met deze tekstverwerker.

Bij Xpol horen een aantal satellietapplicaties:

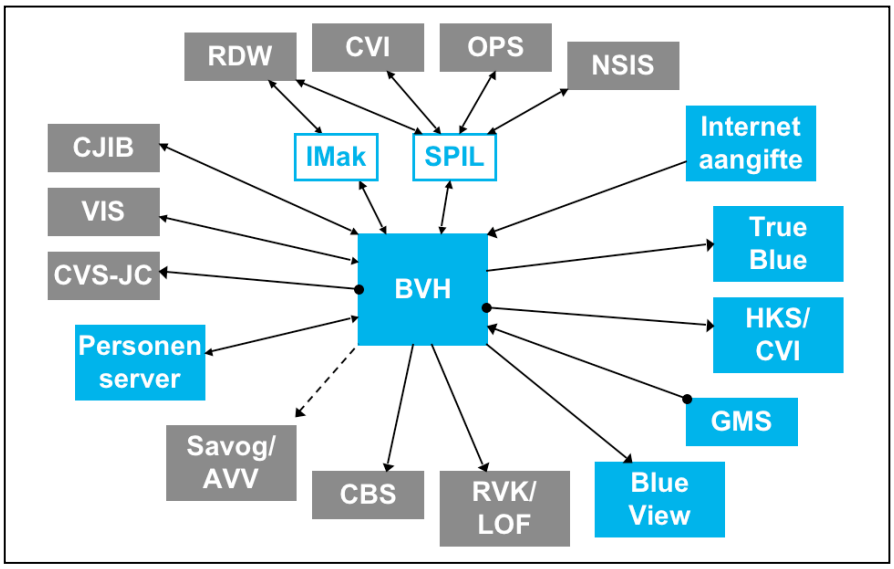
- Aangifte Via Intranet (AVI): Een Java applicatie.
- Dagrapportage (DagRap): Een VB6 applicatie opnieuw ontwikkeld in Java.
- Zoeken in Formulieren (ZiF): Een C# applicatie.
- Afspraak op Locatie (AoL): Een C# applicatie.
- Zicht op Zaken (ZoZ): Een PHP applicatie met MySQL database.
- Xpol Correctietool (XPCT): Een ASP.Net applicatie.

Deze satellietapplicaties werken direct op de Sybase database van Xpol. Daarnaast bestaan er enkele additionele koppelingen tussen de satellieten en de Xpol applicaties zelf.

Naast het toevoegen en aanpassen of herontwikkelen van deze satellietapplicaties omvat de scope van het BVH project aanpassingen aan het Xpol datamodel om dit op lijn te brengen met de GegevensSet Handhaving (GSH).

2.2.2 Externe architectuur

Het BVH systeem is gekoppeld met een groot aantal landelijke en enkele regionale systemen. De landelijke koppelingen zijn aangegeven in Figuur 3.



Figuur 3: Externe architectuur BVH, zoals beschreven in het document "Landelijke Koppelingen BVH" en ontleend aan de technische interviews. De NSIS en CVI koppelingen zijn niet vermeld in het Project Initiatie Document van BVH. Systemen binnen het politie domein zijn blauw weergegeven, terwijl systemen daarbuiten grijs zijn. Pijlen geven de richting van data uitwisseling weer, waar een zwart bolletje ontvangstbevestiging weergeeft.

Deze landelijke koppelingen vinden plaats door bestandenoverdracht middels FTP of TCP/IP, en in sommige gevallen door het verzenden van e-mail. Zowel XML als ASCII bestanden worden uitgewisseld.

Naast de landelijke koppelingen kent BVH ook regio-specifieke koppelingen. De ontwikkeling en het testen van deze koppelingen valt buiten de scope van het BVH project. Voor deze koppelingen worden slechts koppel*vlakken* (interfaces) aangeboden.

2.2.3 Systeem architectuur

De systeem architectuur van BVH kent onder andere de volgende technologieën:

- Java/Oracle Application Server, C#, ASP.Net, Accell/SQL, PHP
- Sybase, MySQL
- Tru64 Unix, Windows 2003, Linux, Citrix
- IIS/.NET, Apache
- Legasuite (Seagull)
- Internet Explorer
- MS-Word, HTML, XML, PDF, RTF, ASCII
- FTP, MSMQ, http, email, ODBC, RPC, Interactive SQL, JDBC, OLEDB, telnet

Kortom, de BVH systeem architectuur is gebaseerd op een grote diversiteit aan technologieën op alle niveau's (hardware, operating systems, programmeertalen, communicatieprotocollen, gebruikersinterfaces). Tabel 5 geeft een overzicht van de gebruikte technologie per systeemonderdeel.

	Xpol	XPTC	GUI	itx4wd	FrmSv	AVI	DagRap	ZiF	ZoZ	AoL	DFS	GLog
Accell	866K											
C	57K		2K	31K								
shell	*38K*											
VBA			1K				2k					
xslt			6K									
Java						110K	16K					
JSP						*20K*	*7K*					
C#		11K		29K	9K			7K		4K	23K	5K
SQL	*153K*											
PHP									*24K*			
JS						*5K*	*11K*		*7K*			
ITX.fo	75K											

Tabel 5, Overzicht technologie per systeemonderdeel. Omvang in regels code.

2.3 Beschrijving van het project

Het Project Initiatie Document (PID) voor BVH is in maart 2007 goedgekeurd door de Raad van Hoofdcommissarissen (RVHC) en het BVH project is vervolgens in april 2007 gestart.

Het project wordt uitgevoerd vtsPN. Voor het realiseren van de GUI wordt gebruik gemaakt van de diensten van de externe leverancier Seagull.

In september 2007 is er een eerste release van het BVH systeem opgeleverd. Deze release was niet functioneel compleet en had nog niet het volledige testtraject doorlopen.

De functionele scope van het project is gedurende de loop van het project niet constant gebleven. Onder andere zijn bij besluit van de stuurgroep de satellietapplicaties Zicht op Zaken (ZoZ) en Afspraak op Locatie (AoL) toegevoegd. In het PID was van ZoZ al sprake als mogelijke uitbreiding, maar de afspraak functionaliteit zou oorspronkelijk met generieke office applicaties worden ondersteund.

In de periode tussen 25 oktober en 23 november 2007 heeft een landelijke gebruikers acceptatietest (GAT) plaatsgevonden van BVH, welke tot doel had om de werkbaarheid van het systeem te valideren. Deze test heeft 21 ernstige bevindingen aan het licht gebracht en een kleine 200 minder urgente bevindingen. Als conclusie van de GAT is geconstateerd dat de werkprocessen die door BVH ondersteund dienen te worden met enkele uitzonderingen goed kunnen worden doorlopen.

Op 21 november 2007 is een release van het BVH systeem door het bouwteam opgeleverd welke vervolgens door het infrastructuurteam is geïnstalleerd in de centrale testomgeving. Vóórdat op deze versie functionele acceptatie tests zijn uitgevoerd is het systeem regionaal geïnstalleerd in verzorgingsgebied Zuid. Het doel was om het systeem vóór 1 december in deze regio beschikbaar te stellen.

Bij het testen van het regionaal geïnstalleerde systeem op 29 november kwamen een aantal ernstige bevindingen aan het licht. Op 30 november en op 1 december is gepoogd deze problemen te verhelpen. Dit is niet binnen die tijdspanne gelukt en op 1 december is de beslissing gevallen om de uitrol te staken.

Inmiddels is de uitrol van BVH opnieuw in gang gezet. Mid-december is een een team opgezet om de uitrol in de verzorgingsgebieden te ondersteunen. De activiteiten van dit zogenaamde UIT team voor uitrol van BVH in het eerste verzorgingsgebied worden in 4 iteraties uitgevoerd. Gedurende elke iteratie wordt de installatieprocedure doorlopen en worden alle installatiestappen nauwkeurig gedocumenteerd en verder geautomatiseerd. Elke iteratie beslaat ongeveer 9 werkdagen.

3 Beoordeling broncode en documentatie

Dit hoofdstuk beschrijft de beoordeling van de onderhoudbaarheid en daarmee de huidige technische staat van het BVH systeem. Deze beoordeling is gebaseerd op de waarnemingen uit Appendix A en is samengevat in 3.1. Via de generieke beoordelingsmatrix van de Software Improvement Group wordt de beoordeling in 3.2 toegepast op de ISO 9126 definitie van onderhoudbaarheid. Een identificatie van de risico's volgt in 3.3.

3.1 Beoordeling van de waarnemingen

Het BVH systeem is groot en complex, zoals al beschreven in Sectie 2.1. Het systeem is opgebouwd uit een centrale kern, en een aantal satellieten. De delen van de applicatie zijn verbonden door een verscheidenheid van protocollen, en de delen van applicatie zijn gebaseerd op een groot aantal verschillende technologieën. De beoordeling van het systeem als geheel is gelaagd opgebouwd.

Het BVH systeem is op twee niveaus beoordeeld: als geheel systeem in Sectie 3.1.1; de BVH core, de afzonderlijke satellieten en de bibliotheekcomponenten in Secties 3.1.3 tot en met 3.1.13. De algemene beoordeling op BVH is gebaseerd op analyses op systeem niveau, en van de BVH core, en op de aggregatie van de beoordelingen van de afzonderlijke satellieten.

De beoordeling heeft 5 schalen, variërend van zeer slecht tot zeer goed.

3.1.1 Beoordeling van de waarnemingen op algehele systeem niveau

De beoordeling van de waarnemingen uit het vorige hoofdstuk staat in onderstaande Tabel. Deze beoordeling is gebaseerd op analyses op systeem niveau, en van de BVH core, en op de aggregatie van de beoordelingen van de afzonderlijke satellieten.

Waarneming	Beoordeling	Motivatie
High level architectuur	Zeer slecht	- Veelvoud van technologieën. - Verouderde technologie in de kern van de applicatie. - Apart database management systeem ZoZ - Aantal generatoren met onduidelijke status - Screenscraping (GUI en AVI/ITX)
Modulariteit	Slecht	- Datalaag Java satellieten niet gedeeld AVI/Dagrap - Referential integrity van Xpol database: in Accell, in triggers, en in satellieten
Separation of Concerns	Slecht	- GSH: wijzigingen in Database en in User interfaces - Formulieren: AVI afhankelijk van vraag/antwoord teksten
Omvang	Zeer slecht	- Totale omvang zeer groot; schatting rebuild value: 267 manjaar, exclusief formulieren - Voornaamste delen: Xpol Accell (169 manjaar), Xpol SQL (21 manjaar), en AVI (23 manjaar)

Complexiteit	Zeer slecht	- Hoge code complexiteit op een aantal plaatsen (5% tot 10% hoog complex) - Repareerbaar, score gaat na herstel naar goed bij het verwijderen van de hoog complexe code uit de satellieten.
Methode lengte	Slecht	- Hoge methode lengte door de C# en Java satellieten heen
Duplicatie	Zeer slecht	- Zeer hoge redundantie in Xpol (36%) - Hoge redundantie in nieuw gebouwde satelliet Dag-Rap (10%)
Exception Handling	Slecht	- Hoge aantallen verkeerde fout afhandelingen (231 in AVI)
Unit-test	Zeer Slecht	- Unit testing is onvoldoende toegepast voor nieuwe code. Unit Testing is echter voor het hele systeem nauwelijks toegepast.
Proces	Slecht	- Geen dagelijkse builds op integratie omgeving - Geen gedeelde coding standards bewaakt door tool - Geen unieke code repository, versioning system

Tabel 6: Beoordeling BVH systeem.

Duplicatie betreft het voorkomen van dezelfde code fragmente op verschillende plekken in de broncode van het systeem. Een bepaalde mate van duplicatie komt in alle systemen voor, maar hoge percentages redundante code duidt op slordig programmeren en/of structurele problemen. Exception handeling betreft de in het systeem aanwezige logica die actief wordt op het moment dat afwijkende situaties optreden. In de appendix wordt de beoordeling op systeemniveau in detail behandeld van deze en andere aspecten genoemd in Tabel 6. In de hierna volgende secties worden de beoordelingen op het niveau van systeemonderdelen kort behandeld.

3.1.2 Beoordeling BVH Core

Het BVH systeem is gebaseerd op de bestaande handhavingsapplicatie Xpol (nu ook BVH Core genoemd). Xpol is ontwikkeld in de vierde-generatie programmeertaal Accell en maakt gebruik van Sybase database technologie. Er wordt ook gebruik gemaakt van shell scripts, en C, met name voor het realiseren van koppelingen met externe systemen. Xpol biedt een karakter-georiënteerde user interface (niet-grafische schermen).

Naast de database voor de Xpol gegevens zelf kunnen twee secondaire databases worden onderscheiden. De formulieren database wordt gebruikt voor het opslaan van formulieren. De logging database wordt gebruikt om het raadplegen en muteren van de database te loggen. De Xpol database zelf heeft de grootste complexiteit (382 tables, 568 triggers, 43 procedures, 43 views), gevolgd door de logging database (68 tables, 21 procedures, 37 views).

Er zijn geen declaraties van foreign keys of andere referentiele integriteitsconstraints in de database schema's opgenomen. Voor het bewaken van de referentiele integriteit worden twee mechanismen gebruikt. De keuze tussen deze mechanismen wordt ge-

maakt door een procedure genaamd “enforce_ri”, die test of de aanroepende applicatie “Xpol” heet. Als dit het geval is wordt aangenomen dat de applicatie zelf de referentiele integriteit bewaakt. Als dit niet het geval is, d.w.z. als een satelliet applicatie de database benaderd, dan worden database triggers geactiveerd om de referentiele integriteit te bewaken.

Zoals al aangegeven in Figuur 3 kent het BVH systeem een groot aantal koppelingen met externe systemen. Deze koppelingen zijn gerealiseerd in de BVH core middels een veelvoud aan technologieën. Meerdere koppelingen zijn gerealiseerd in de vorm van Accell programma’s die informatie uitwisselen met Unix shell scripts. Ook wordt gebruik gemaakt van routines en programma’s die in C of Java zijn geïmplementeerd. De data uitwisselingsformaten en protocollen die door de verschillende koppelingen worden gehanteerd omvatten onder andere SOAP, FTP, e-mail, en in een enkel geval het uitprinten van een papieren formulier. De broncode van Xerces, een open source pakket voor het lezen en schrijven van XML bestanden, is in de broncode van BVH core opgenomen.

Tussen de twee aangeleverde versies van de BVH Core bestaan weinig verschillen. De enige wijzigingen die zijn aangebracht betreffen aanpassingen in de Accell code vanwege wijzigingen in de gegevensset handhaving (GSH) en vanwege de (halfweg bevroren) introductie van een tabellenserve (XTBS). Verder is er een in C geschreven programma geïntroduceerd dat gebruikt is om Accell code te converteren ten behoeven van het toevoegen van de GUI. Deze conversie komt in de volgende sectie aan bod.

De omvang van de BVH core is bijzonder groot. De geschatte rebuild value van de Accell code (169 manjaar) en de Sybase SQL code (21 manjaar) zijn bijzonder hoog. De hoeveelheid duplicatie in de Accell code ligt ook hoog (36% redundante code regels).

Waarneming	Beoordeling	Motivatie
Omvang	Zeer slecht	Geschatte rebuild value meer dan 160 man jaar
Complexiteit	Neutraal	Weining hoog complexe code
Duplicatie	Zeer slecht	36 % redundantie.
Exception handling	Neutraal	Normaal gebruik van taal constructies voor het afhandelen van onverwachte situaties
Unit testing	Zeer slecht	Geen unit-test framework voor Accell of C.

Tabel 7, Waarnemingen BVH Core

Kortom, de BVH Core is van bijzonder grote omvang en van lage kwaliteit. In overeenstemming me het projectplan zijn gedurende het BVH project zeer weinig wijzigingen aangebracht.

3.1.3 Beoordeling GUI

In het BVH systeem is aan Xpol een grafische user interface (GUI) toegevoegd, gebruikmakend van zogenaamde *screen-scraping* technologie van Seagull. Als onderdeel van de GUI is een op MS-Word georiënteerde tekstverwerker opgenomen (ITX4Word). De landelijke standaard formulierenset is compatible gemaakt met deze tekstverwerker.

Om de GUI te realiseren is gebruik gemaakt van code conversie en generatie. Ten eerste zijn de Accell form scripts geconverteerd middels een in C geschreven conversie programma. Deze conversie bestaat in het automatisch toevoegen van extra informatie aan de karaktergebaseerde schermen om het screen-scrapen te vereenvoudigen. Ten tweede is een reeks in XSLT geschreven transformatie scripts toegepast om uit form scripts grafische schermen te genereren. Uiteindelijk is het mogelijk om met de combinatie van conversie en generatie van code het overgrote deel van de 1195 grafische schermen volautomatisch te genereren, terwijl voor 122 resterende schermen handmatig werk vereist was. De code in Seagull technologie benodigd hiervoor bestaat uit 168 panels, 164 screens, en 53 scripts.

De screen-scraping gebaseerde koppeling van de GUI met de onderliggende Xpol applicatie kent niet de synchronisatie problematiek zoals die bekend is uit het BVO project. In het BVO project is middels screen-scraping een koppeling gemaakt met een SmartStar applicatie. Hierbij moest veelvuldig gebruik gemaakt worden van zogenaamde wait statements om synchronisatie van de grafische schermen met de onderliggende terminal schermen te realiseren. Dit vanwege het ontbreken van een signaal vanuit de terminal dat aangeeft dat schermopbouw afgerond is. In tegenstelling tot het BVO project is er in het BVH project sprake van een onderliggende terminal interface die wel een signaal afgeeft bij het afronden van schermopbouw. Hierdoor is er geen noodzaak tot het veelvuldig gebruik van wait statements met daaraan verbonden synchronisatie problemen. Toch worden in de GUI van BVH enkele (10) wait statements gebruikt. Dit vanwege de mogelijkheid om de Xpol applicatie in zogenaamde “find” mode te gebruiken, waarbij opnieuw geen signalering van het afronden van schermopbouw plaatsvindt. Er zijn gemiddeld 11 velden per Xpol scherm die in “find” mode gebruikt kunnen worden, maar hoeveel gebruikers hier daadwerkelijk gebruik van maken is onbekend.

Waarneming	Beoordeling	Motivatie
Omvang	Zeer goed	Slechts 2k regels C code voor de conversie van de schermen, 6k regels XSLT code voor schermgeneratie, en 1k Visual Basic scripts. Een compact systeem.
Complexiteit	Goed	Gebruik van XML patronen.
Methode Lengte	Goed	Verdeling van generatie stappen over XSLT scripts van beperkte omvang
Duplicatie	Slecht	15% duplicatie in C. 34% in de VB scripts. Duplicatie is echter niet een issue door de geringe omvang.

Tabel 8, Waarnemingen GUI.

Naast de koppeling van de GUI met de onderliggende Xpol applicatie zelf, middels screen scraping, is er ook een koppeling van de GUI aanwezig met de ITX4Word tekst-verwerkingsapplicatie. Deze koppeling vindt plaats middels een telnet sessie.

Kortom, de GUI die in het BVH project is gerealiseerd in samenwerking met Seagull is een bijzonder compact systeem. Het kent weliswaar hoge code redundantie maar door de geringe omvang van de code is dit geen issue.

3.1.4 Beoordeling AVI

Het satelliet systeem AVI (Aangifte via Intranet) is ontwikkeld in Java (110KLOC), JSP (20KLOC), en JavaScript (5KLOC). Hiermee is AVI veruit de grootste van de satellieten. De geschatte rebuild value van AVI bedraagt 23 manjaar.

AVI is gekoppeld met de database van BVH Core middels JDBC connecties en maakt gebruik van Enterprise Java Beans (EJBs).

AVI kent tevens een koppeling met ITX (de C kern van ITX4Word) voor het aanmaken van formulieren. Deze koppeling is gerealiseerd middels een telnet verbinding en screen-scraping, maar maakt geen gebruik van Seagull technologie. De werking van de koppeling komt neer op het simuleren van een interactieve ITX gebruiker door de AVI applicatie waarbij formuliervragen door de AVI applicatie moeten worden geïnterpreteerd om de juiste informatie in het formulier in te vullen. Om deze koppeling te kunnen realiseren zijn wijzigingen aangebracht in de bewoordingen van formulieren. Hiermee zijn AVI en ITX dus sterk gekoppelde applicaties, waarbij wijzigingen in de één vaak wijzigingen in de ander met zich mee zullen brengen.

Ten opzichte van de versie bij aanvang van het BVH project zijn in AVI wijzigingen doorgevoerd waaronder aanpassingen wegens GSH, overschakelen naar de Oracle Application Server, en het vereenvoudigen van de datalaag die toegang geeft tot de database.

Waarneming	Beoordeling	Motivatie
Omvang	goed	109k regels nu (was 103k) schatting van 23 man jaar rebuild value,
Complexiteit	Zeer slecht	9% van de code hoog complex Repareerbaar, score gaat na herstel naar goed bij het verwijderen van de hoog complexe code.
Methode lengte	Slecht	18 % Hoge methode lengte
Duplicatie	Slecht	12 % duplicatie (was 13,7 %)
Exception Handling	Slecht	Hoge aantallen verkeerde fout afhandelingen (231-was 252)
Unit-test	Neutraal	Unit testing is ingevoerd voor nieuwe code, maar ontbreekt voor de aangeleverde versie aan de start van het project

Tabel 9, Waarnemingen AVI

Voor AVI is een regressietest infrastructuur aanwezig op basis van het tool Rational Robot dat automatische functionele tests ondersteund.

Kortom, de AVI satelliet heeft gedurende het BVH project belangrijke wijzigingen ondergaan. Hierbij is het volume licht toegenomen en is de algehele kwaliteit enigszins verbeterd.

3.1.5 Beoordeling DagRap

Dagrapportage is een bevragingssysteem voor de applicatie Xpol in de vorm van een webapplicatie. De gebruiker heeft dus een intranetomgeving voor het benaderen van de gegevens vanuit Xpol en een evt. bijbehorende formulierendatabase.

Deze bevragingen kunnen onder andere het volgende inhouden:

- Overzichten incidenten of acties in een bepaalde tijdsperiode.
- Opvragen specifiek persoon, bedrijf, voertuig, registratie, locatie en communicatiemiddel.
- Zoeken op combinatie. Hierbij kan men denken aan zoeken naar alle registraties van een persoon in een bepaalde tijdsperiode etc.
- Opstellen van overzichten: personen in een buurt, etc...

Alle bevragingen worden gelogd in Xpol zodat het altijd inzichtelijk is wie bevragingen heeft uitgevoerd. Tevens is het mogelijk om opgestelde formulieren op te vragen en te bekijken via dagrapportage.

DagRap is opnieuw gebouwd in Java. De mapping met de Xpol database is verzorgd via Toplink. Tevens is DagRap verbonden met de FormServer.

Waarneming	Beoordeling	Motivatie
Omvang	Zeer goed	16K regels Java code
Complexiteit	Slecht	10% van de code complex, waarvan 5% van de code hoog complex. Neigend naar zeer slecht.
Methode lengte	Slecht	15% van de code te lange methodes
Duplicatie	Neutraal	7% redundantie. Deze dient te worden bewaakt want bij een hogere duplicatie gaat de satelliet naar een slechte score.
Exception Handling	Slecht	31 verkeerde foutafhandelingen.
Unit Testing	Slecht	Zeer weinig Unit Testing code

Tabel 10, Waarnemingen DagRap.

Kortom, de DagRap satelliet, die tijdens het BVH project nieuw is ontwikkeld, kent niettemin een aantal kwaliteitsproblemen. Door de beperkte omvang van deze satelliet kunnen deze problemen met een gerichte inspanning op korte termijn teruggedrongen worden.

3.1.6 Beoordeling AoL

Afspraak op Locatie (AoL) wordt gebruikt om collega's te informeren over diverse zaken die op een bepaalde locatie spelen. De belangrijkste zijn die AoL's die van invloed zijn op de veiligheid van de collega's.

Waarneming	Beoordeling	Motivatie
Omvang	Zeer goed	Slechts 4K regels code, een compact systeem.
Complexiteit	Zeer goed	Geen complexe code in de satelliet.
Methode lengte	Zeer goed	Geen lange methodes in de satelliet.
Duplicatie	Zeer slecht	29 % redundantie.
Exception Handling	Slecht	12 verkeerde foutafhandelingen. Maar, voor dit kleine systeem is dit te repareren

Tabel 11, Waarnemingen AoL.

De AoL satelliet is, zoals blijkt uit Tabel 11, van geringe omvang en vrij van hoog complexe code. Het voornaamste kwaliteitsprobleem is relatief hoge code redundantie.

3.1.7 Beoordeling XPCT

Het doel van het Xpol Correctie Tool (XPCT) is om eenvoudig, sneller en volgens de juiste procedures verbeteringen te kunnen doorvoeren in de Xpol database. Tevens kunnen middels dit tool bijvoorbeeld dubbel ingevoerde gegevens uit de Xpol database verwijderd worden. XPCT is meer een tool voor de gevorderde Xpol database beheerder dan voor de normale gebruiker.

Waarneming	Beoordeling	Motivatie
Omvang	Zeer goed	Slechts 11K regels code, een compact systeem.
Complexiteit	Goed	Nauwelijks te complexe code in de satelliet, en te verbeteren naar zeer goed.
Methode lengte	Goed	Nauwelijks te lange methodes in de satelliet, maar stijgend. Dit dient bewaakt te worden.
Duplicatie	Zeer slecht	21 % redundante regels. (was 17%.)
Exception Handling	Slecht	60 verkeerde foutafhandelingen (was 54). Maar, voor dit kleine systeem is dit te repareren

Tabel 12, Waarnemingen XPCT

De XPTC satelliet is compact en kent zeer weinig complex code. Het voornaamste kwaliteitsprobleem is het relatief hoge niveau van code redundantie. Ook op de kwaliteitsvlakken waar XPTC goed scoort is een lichte verslechtering te constateren ten opzichte van de situatie bij aanvang van het BVH project.

3.1.8 Beoordeling ITX4Word

ITX4Word is een component dat door meerdere applicaties gebruikt wordt. Applicaties kunnen m.b.v. deze component Xpol formulieren uit de formulierendatabase ophalen en converteren naar andere formaten.

Waarneming	Beoordeling	Motivatie
Omvang	Zeer goed	Slechts 29K regels code, een compact systeem.
Complexiteit	Slecht	12% complexe code, waarvan 2% zeer complex.
Methode lengte	Slecht	19% van de code in lange methodes, waarvan 8% in zeer lange methodes.
Duplicatie	Zeer Goed	5 % redundantie.
Exception Handling	Neutraal	59 verkeerde foutafhandelingen. Maar, voor dit kleine systeem is dit te verbeteren tot goed.

Tabel 13, Waarnemingen ITX4Word

De kwaliteit van de nieuw ontwikkelde ITX4Word satelliet is hoog. Op het gebied van foutafhandeling kan nog een verbeteringsslag gemaakt worden.

3.1.9 Beoordeling ZIF

De satelliet ZIF heeft tot doel om met steekwoorden formulieren te vinden waarin de steekwoorden voorkomen en de gevonden formulieren te tonen.

Waarneming	Beoordeling	Motivatie
Omvang	Zeer goed	Slechts 7k regels code (was 1,5k), een compact systeem.
Complexiteit	Goed	5% complexe code (was 0%), maar (nog) geen zeer complexe code in de satelliet..
Methode lengte	Slecht	6% van de code in zeer lange methodes (was 0%). Het gaat om een zeer klein aantal methodes..
Duplicatie	Zeer slecht	27 % redundante code (was 7%).
Exception Handling	Slecht	15 verkeerde foutafhandelingen (was 8). Herstelbaar.

Tabel 14, Waarnemingen ZIF

De ZIF satelliet is sterk gegroeid tijdens het BVH project, maar is nog steeds van beperkte omvang. De kwaliteit is gedurende deze groei significant gedaald op alle bovenstaande vlakken. Gezien de beperkte omvang is deze kwaliteitsdaling nog eenvoudig omkeerbaar.

3.1.10 Beoordeling ZoZ

De Zicht Op Zaken satelliet (ZoZ) biedt overzichtsrapportages over lopende zaken. ZoZ is gerealiseerd in de script talen PHP (24KLOC) en JavaScript (7KLOC), en niet in de meer gangbare talen C# en Java die door de overige satellieten worden gebruikt. Er wordt gebruik gemaakt van een aparte database, gerealiseerd middels MySQL, waarin een selectie wordt opgeslagen van gegevens uit de Sybase database van de BVH Core.

ZoZ wordt gevoed door middel van periodieke exports uit de Xpol database. De geëxporteerde gegevens worden via e-mail verstuurd van de Xpol beheerder naar de ZoZ beheerder. Vervolgens worden de gegevens ingelezen en in de MySQL database van ZoZ opgeslagen.

In de PHP code van de ZoZ applicatie is onvoldoende scheiding aangebracht tussen presentatie functionaliteit en business logica. Ook wordt te weinig gebruik van functies voor structurering en intern code hergebruik. Ook wordt te weinig gebruik gemaakt van de object-orientatie features van PHP. Dit uit zich ook in een de hoge mate van code duplicatie (45% redundante coderegels).

Het bevragen van de onderliggende database is gerealiseerd door middel van SQL queries die verweven zijn met de business logic en presentatie code. Eventuele fouten die optreden bij deze bevraging worden niet op gestructureerde wijze afgehandeld.

SIG heeft een bug geconstateerd in de authenticatie functionaliteit. Verder is de robuustheid van de applicatie op het gebied van security sterk afhankelijk van server configuratie.

Waarneming	Beoordeling	Motivatie
Omvang	Zeer goed	Slechts 24k regels PHP code, een compact systeem.
Complexiteit	Slecht	Control-flow logica en SQL queries zijn verweven met de presentatie code
Methode lengte	Slecht	Monolitische scripts door weinig gebruik van functies en/of object-orientatie features van PHP.
Duplicatie	Zeer slecht	45 % redundantie.
Exception handling	Slecht	Eventueel optredende fouten, bijvoorbeeld bij bevraging database, worden niet netjes afgehandeld.

Tabel 15, Waarnemingen ZoZ

Kortom, de ZoZ satelliet is een redelijk klein systeem, maar ontwikkeld in afwijkende technologie. De code kwaliteit is laag.

3.1.11 Beoordeling FormServer

De formulieren server (FormServer) is op zichzelf geen satelliet, maar een component die door meerdere satellieten wordt gebruikt, namelijk door ZiF en DagRap (zie ook

Figuur 2). De taak van de formulieren server is het aanbieden van functionaliteit op de formulieren database aan deze satelliet applicaties. Verder is de formulieren server gekoppeld aan ITX4Word voor het converteren van formulieren. In onderstaande tabel worden de waarnemingen op de FormServer samengevat.

Waarneming	Beoordeling	Motivatie
Omvang	Zeer goed	Slechts 9k regels code. Een compacte component.
Complexiteit	Slecht	7% complexe code, waarvan 3% hoog complex.
Methode lengte	Slecht	17% code in lange methodes
Duplicatie	Slecht	15% redundante code regels
Exception handling	Slecht	50 verkeerde foutafhankelingen

Tabel 16, Waarnemingen Form Server.

De formulieren server is nieuw ontwikkeld tijdens het BVH project. Het is een compacte component, maar de code kwaliteit is op meerdere vlakken onvoldoende.

3.1.12 Beoordeling DFS

De DBI Foundation Suite (DFS) is een component die door in C# ontwikkelde satelliet-tapplicaties gebruikt kan worden voor toegang tot de Xpol database. DFS wordt op dit moment gebruikt door XPCT, ZiF, en de formulieren server, maar niet door AoL of ITX4Word.

De DFS biedt onder meer een generator die het mogelijk maakt om code te genereren uit PowerDesigner datamodellen. Deze gegenereerde code vormt dan, gezamenlijk met ondersteunende code van de DFS component, de datalaag voor een C# satelliet. De PowerDesigner modellen zijn echter niet bijgewerkt nadat het SQL datamodel van Xpol is aangepast. In onderstaande tabel worden de waarnemingen op DFS samengevat.

Waarneming	Beoordeling	Motivatie
Omvang	Neutraal	23k regels code., redelijk groot voor een bibliotheek
Complexiteit	Goed	2% complexe code
Methode lengte	Neutraal	10% code in lange methodes
Duplicatie	Goed	5% redundante code regels
Exception handling	Slecht	106 verkeerde foutafhankelingen

Tabel 17, Waarnemingen DFS.

De DFS component bestond al bij aanvang van het BVH project en heeft nagenoeg geen veranderingen ondergaan. De kwaliteit van de component is redelijk. Het niet bijwerken van PowerDesigner modellen waarvan DFS afhankelijk is vormt een belangrijk onderhoudsrisico.

3.1.13 Beoordeling GLog

De Gebruikers Logging (GLog) component kan door C# satelliet applicaties gebruikt worden om, aanvullend aan de logging van database acties door de BVH core zelf, informatie vast te leggen over gebruikersactiviteit. Onderstaande tabel geeft een overzicht van de waarnemingen op GLog.

Waarneming	Beoordeling	Motivatie
Omvang	Zeer goed	5k regels code, compacte bibliotheek
Complexiteit	Zeer goed	Geen complexe code
Methode lengte	Zeer slecht	33% code in lange methodes
Duplicatie	Zeer goed	Slechts 4% redundante code regels
Exception handling	Slecht	32 verkeerde foutafhankelingen

Tabel 18, Waarnemingen GLog.

De GLog component is klein en niet complex. Er is bijzonder weinig code duplicatie. Op het vlak van lengtes van methodes en foutafhandelingen is de kwaliteit laag.

3.2 Onderhoudbaarheid op langere termijn

De beoordeling uit de vorige paragraaf is toegepast op het begrip onderhoudbaarheid uit het ISO 9126 kwaliteitsmodel voor software.

> De onderhoudbaarheid van het BVH systeem is zeer laag.

Deze conclusie volgt uit het generieke model in onderstaande Tabel 19. Het model legt een relatie is tussen de waarnemingsgebieden en de kwaliteitskarakteristieken uit het ISO 9126 model. De kwaliteitskarakteristieken staan in de linkerkolom, aan de bovenzijde staan de waarnemingsgebieden. Waar een waarnemingsgebied een significante bijdrage levert aan een kwaliteitskarakteristiek staat een kruisje in de tabel.

	High level architectuur	Modularisation	Separation of concerns	Complexity	Method length	Duplication	Volume	Exception handling	Unit test quality	Process	Beoordeling
Beoordeling	--	-	-	--	-	--	--	-	--	-	
Analysability		x	x		x	x	x				-
Changeability	x		x	x		x					--
Stability								x	x	x	-
Testability			x	x	x				x		--

Tabel 19: Relatie waarnemingsgebieden en kwaliteitsmetrieken.

Onderstaande paragrafen geven een korte onderbouwing van de onderhoudbaarheid op hoofdlijnen op basis van de ISO 9126 kwaliteitskarakteristieken gerelateerd aan de observaties van het vorige hoofdstuk .

3.2.1 Analysability

Met 'analysability' wordt de mate aangeduid waarin de applicatie begrijpelijk is vanuit de code voor het doen van aanpassingen. De analyseerbaarheid is slecht. Dit wordt bepaald door een aantal negatieve factoren. Dit wordt ondermeer bepaalt door het gebrek van delen van de datalaag tussen AVI en DagRap. De meervoudige implementatie van de refential integriteit van de database in Accell, database triggers, en in de satellieten benadeelt de analysability van het systeem verder. Wijzigingen in de GSH zorgen voor vergelijkbare maar toch weer iets andere veranderingen in databases, user interfaces, en formulieren. De lange methods, hoge duplicatie, en het grote volume van het systeem zijn verdere negatiever factoren voor de analysability.

3.2.2 Changeability

De changeability of veranderbaarheid is gedefinieerd als de inspanning benodigd om te modificeren, corrigeren of het toevoegen van nieuwe functionaliteit. De veranderbaarheid is van het systeem is zeer slecht. Dit is voornamelijk veroorzaakt door de veelvoud van technologieën in het systeem, en de verouderde technologie in BVH core. Verder zorgen wijzigingen in de GSH voor vergelijkbare wijzigingen in verschillende delen van het systeem. De complexe code en hoge mate van code duplicatie maken het verder moeilijk om code veranderingen door te voeren.

3.2.3 Stability

Met 'stability' wordt de maat aangeduid waarin de applicatie ongevoelig is voor het optreden van fouten bij het doen van aanpassingen. De stabiliteit van de broncode is slecht. Dit wordt enerzijds veroorzaakt door de slechte exception handling met name voor AVI. Anderzijds ondersteunen de oudere en omvangrijke technologieën zoals Accell niet moderne foutafhandelingsfunctionaliteit, wat de stabiliteit negatief beinvloed. Het ontbreken van voldoende unit-tests bemoeilijkt verder het opsporen van fouten in wijzigingen, en dus de stabiliteit. Door het ontbreken van dagelijkse builds op een integratie omgeving, en het ontbreken van een centrale code repository kunnen onderdelen van het BVH systeem met elkaar uit de pas gaan lopen. Op het moment dat de on-

derdelen geïntegreerd worden tot één geheel is vervolgens extra inspanning nodig om de discrepanties op te lossen.

3.2.4 Testability

De testability is een maat voor de hoeveelheid inspanning die benodigd is voor validatie van aanpassingen aan de software. De testbaarheid is zeer slecht. Wijzigingen in de GSH zorgen voor meerdere vergelijkbare veranderingen in de delen van het systeem die elk moeten worden gecontroleerd op correctheid en onderlinge consistentie. Wijzigingen in de formulieren hebben invloed op een groot aantal schermen van de diverse satellieten. De hoge complexiteit van de code en de lange methodes maken het testen van wijzigingen moeilijk. Verder zorgt het ontbreken van voldoende unit-tests een negatieve bijdrage aan testability op systeem niveau.

3.3 Risico's

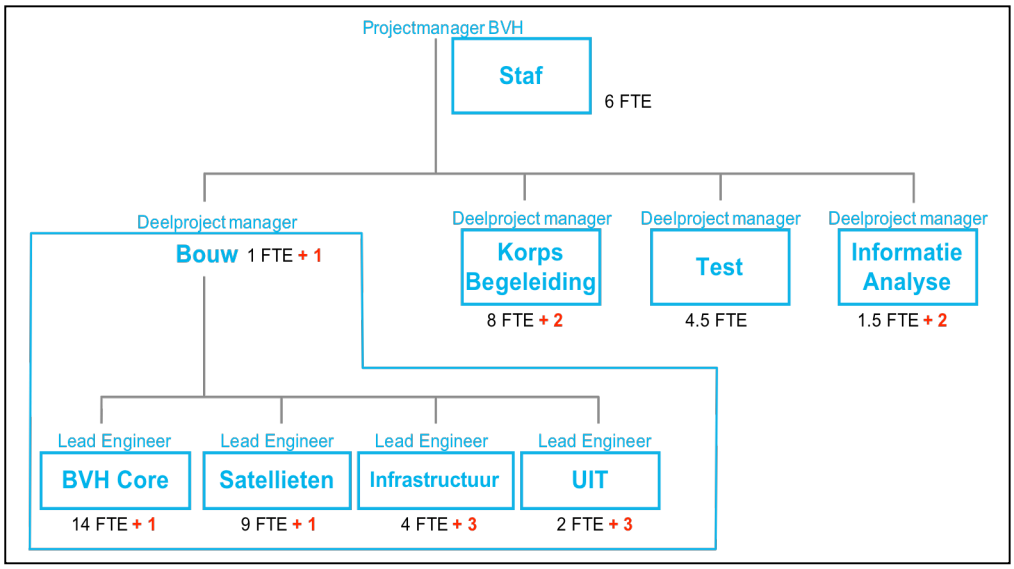
Op basis van de waarnemingen en beoordelingen van het BVH systeem en de verschillende systeemonderdelen kunnen een aantal belangrijke risico's worden geïdentificeerd voor het BVH project.

- Niet-herhaalbare uitrol. Het uitrollen van BVH is op dit moment nog een zeer tijdrovende, foutgevoelige activiteit die diepe systeemkennis vereist. Het herhaalbaar maken van de installatieprocedures is een belangrijke uitdaging.
- Problematisch beheer. Het beheer van het BVH systeem na uitrol zal problematisch zijn, in die zin dat bij uitval of performanceproblemen wederom behoefte zal zijn aan diepe systeemkennis om effectief te kunnen troubleshooten. Het terug in bedrijf brengen na uitval zal bij geval lang kunnen duren.
- Kostbaar onderhoud. De diversiteit van technologieën, de lage code kwaliteit, en de complexe software architectuur zal leiden tot hoge onderhoudskosten.
- Functioneel niet aanpasbaar. In het geval van sterke behoefte aan functionele wijzigingen zal het bijzonder moeilijk zijn het BVH systeem aan te passen. Naast mogelijke gebruikerswensen moet hierbij gedacht worden aan veranderingen in gekoppelde externe systemen die veranderingen in het BVH systeem oproepen.

In hoofdstuk 6 zullen aanbevelingen worden geformuleerd die deze risico's tot op zekere hoogte kunnen verkleinen.

4 Organisatie en bemensing van BVH

De huidige organisatie structuur voor het BVH project is weergegeven in Figuur 4. De organisatie bestaat uit vier teams, waarvan Bouw veruit de grootste is. Het huidig aantal FTEs per onderdeel wordt getoond in zwart, en de gewenste extra FTEs in rood.



Figuur 4, Organisatie structuur BVH project

Het aantal openstaande vacatures worden op dit moment niet als zeer knellend ervaren. Er wordt echter veel “geleend” tussen teams, met namen voor testen en uitrol. Dit wordt als verstorend ervaren op de overige activiteiten, wellicht omdat in de planning van deze activiteiten en in de aansturing van de uitvoerenden onvoldoende geanticipeerd wordt op deze taakwisselingen.

Het functioneel ontwerp van het BVH systeem is op dit moment bij het BVH Core team ingeschoven. Opvallend is de afwezigheid van een software architect in de organisatie. De rol van software architect behelst het bewaken van de algehele software architectuur. In het geval van BVH betreft dit de samenhang van de verschillende systeemonderdelen, maar ook het hoog niveau ontwerp van de deployment processen. Een natuurlijke plaatst voor de software architect in bovenstaand organogram zou zijn naast de deelprojectmanager van het Bouw team.

Het UIT team is sinds mid-december actief in het documenteren, vereenvoudigen, en uitvoeren van het installatieproces, en niet slechts in het testen van de integratie van deelsystemen. De naam van het team (UIT = Unit Integration Testing) is dus misleidend. Het UIT team voert deze activiteiten uit in een centrale omgeving, met het oog op uitrol in het eerste verzorgingsgebied. Wijzigingen en uitbreidingen die het UIT team ontwikkelt worden aan de rest van het bouwteam doorgegeven. Op welke wijze deze ontwikkelingen hun weerslag vinden in de broncode en documentatie van de ontwik-

kelversie van BVH was op het moment van informatievergaring voor deze assessment niet formeel vastgelegd.

Het “lenen” van mensen tussen teams, en de rol van het UIT team illustreert dat de organisatiestructuur van het project voornamelijk is ingericht voor bouw van het systeem en de verschillende systeemonderdelen, en niet voor het ondersteunen van herhaalbare deployment. De bouwfase van het project nadert echter afronding en de komende periode zal in het teken staan van (herhaalde) deployment van het systeem en wellicht van verbetetering van de technische kwaliteit van het systeem (zie ook de aanbevelingen in hoofdstuk 6). De organisatiestructuur en de bemensing van het project zal in dat licht heroverwogen moeten worden.

In de personele bezetting van het project zijn enkele risico’s aanwezig met betrekking tot het delen en vastleggen van kennis. Met name de kennis over de Accell technologie en de legacy component BVH Core zijn sterk geconcentreerd bij enkele individuen. Kennis over de GUI component, ontwikkeld in samenwerking met en op basis van technologie van Seagull, is voor een groot gedeelte aanwezig bij externe projectmedewerkers. Ten slotte is ook de kennis van ITX en de ITX4Word componenten niet breed gedeeld binnen de organisatie. Maatregelen kunnen getroffen worden om deze kennis vast te leggen en breder te delen.

5 Onderzoeksvragen

Zoals in sectie 1.1 is gememoreerd, heeft vtsPN aan SIG een oordeel gevraagd over de kwaliteit van de software ontwikkeling BVH, waarbij vier subvragen te onderscheiden zijn:

1. Wat is de kwaliteit van de ontwerpdocumentatie en de software van de verschillende onderdelen waaruit de BVH bestaat?
2. Op welke wijze is het realiseren van BVH organisatorisch opgezet om te borgen dat de verschillende onderdelen in samenhang functioneren?
3. Is per BVH onderdeel qua bezetting de juiste kwaliteit en capaciteit aanwezig om kwalitatief hoogwaardige software op te leveren?
4. Zijn de condities aanwezig om het software ontwikkelingsproces kwalitatief juist uit te kunnen voeren?

In het voorgaande is een antwoord geformuleerd op de hoofdvraag. In dit hoofdstuk worden de vier subvragen beantwoord, op basis van de waarnemingen uit de appendix en de beoordelingen uit het vorige hoofdstuk.

5.1 Wat is de kwaliteit van de ontwerpdocumentatie en de software van de verschillende onderdelen waaruit de BVH bestaat?

De kwaliteit en onderhoudbaarheid van BVH Core is laag, met name vanwege de grote omvang, het gebruik van een legacy programmeertaal, en hoge code duplicatie. Deze lage kwaliteit was bij aanvang van het BVH project al een feit, en de weinige veranderingen in dit deelsysteem hebben op de kwaliteit geen significante invloed gehad.

De kwaliteit van de satellieten van BVH is redelijk tot goed, met enkele uitzonderingen. De grootste satelliet, AVI, is van lage kwaliteit, maar gedurende het BVH project is deze kwaliteit op een aantal punten licht verbeterd. De nieuw ontwikkelde satelliet DagRap is slechts van lage kwaliteit. De ZiF satelliet is uitgebreid, waarbij de kwaliteit significant omlaag is gegaan. De GUI en ITX4Word zijn van goede kwaliteit. De AoL satelliet, de formserver en de bibliotheekcomponenten DFS en Glog zijn van redelijk tot goede kwaliteit. De ZoZ satelliet is klein van omvang, maar is gebouwd in afwijkende technologie (PHP) en met zeer matige code kwaliteit.

De kwaliteit van het BVH systeem als geheel is laag, met name vanwege de complexe software architectuur en de hoge diversiteit aan technologieën.

De ontwerp documentatie is uitgebreid, doch versnipperd. De kwaliteit van elk van de geleverde documenten samengevat in Tabel 20 is tenminste redelijk. Uitzondering is het ontbrekende functioneel ontwerp van de BVH Core (er zijn wel andere vormen van documentatie aanwezig), en het gebrek aan overzichtelijkheid van de AVI documentatie. De beheerscans geven een goed en bondig beeld van de systeem onderdelen. De documentatie loopt ook mee met de doorgevoerde wijzigingen in de systeem onderdelen. Wat echter ontbreekt is documentatie op niveau van het gehele systeem. De algehele software architectuur van het systeem, en hoe de delen onderling communiceren

en van elkaar afhankelijk zijn, is versnipperd over afzonderlijke documenten en overzichtsdocumentatie ontbreekt.

Systeem onderdeel	Ontwerpdocumentatie
BVH core	FO niet beschikbaar
Xpol Correctie Tool	FO v0.2: voldoende
AVI	FO v1.0: zeer uitgebreid (366 pagina's), veel kruisverwijzingen, niet overzichtelijk, beschrijving van processen en schermen, geen use cases.
Zoeken in Formulieren	FO v1.1: voldoende
Zicht op Zaken	FO v1.2: voldoende
Afspraak op Locatie	FO v1.0: voldoende
Dag Rapportage	FO v1.1: uitgebreid
ITX4Word	FO v0.3: voldoende
Xpol GUI	Systeem documentatie v0.1, d.d. 14-03-2008

Tabel 20, Waarnemingen ontwerpdocumentatie.

5.2 Op welke wijze is het realiseren van BVH organisatorisch opgezet om te borgen dat de verschillende onderdelen in samenhang functioneren?

In hoofdstuk 4 is de organisatorische opzet van BVH aan bod gekomen. In deze opzet is de rol van Software Architect niet formeel benoemd en onvoldoende ingevuld. Hiermee ontbreekt overzicht en bewaking van de samenhang van de verschillende systeemonderdelen. Ondanks dat de Lead Engineers van de verscheidene deelteams overleggen over deze samenhang is het van belang om eindverantwoordelijkheid voor de algehele software architectuur op één plaats te beleggen.

De huidige organisatorische opzet is vooral gericht op het bouwen van de verscheidene systeemonderdelen. Echter, naarmate het project vordert kenmerkt het project zich in toenemende mate door de problematiek van integreren van de delen en het herhaald operationeel maken van het geheel. Het UIT deelteam dat in december is samengesteld heeft deze nadruk op deployment tot uitdrukking gebracht, maar de overige dealteams van het bouwteam zullen ook nauwer betrokken moeten worden bij het uitrolbaar maken van de door hen geproduceerde software. Ook voor de software architect zou er een taak liggen om het hoog niveau ontwerp van de deployment processen te bewaken.

5.3 Is per BVH onderdeel qua bezetting de juiste kwaliteit en capaciteit aanwezig om kwalitatief hoogwaardige software op te leveren?

Per onderdeel zijn voldoende kwaliteit en capaciteit aanwezig om voor elk onderdeel technische goede software te bouwen. Echter, de inspanningen op de verschillende onderdelen behoeven betere coördinatie om de kwaliteit van het systeem te waarborgen.

Deze coördinerende rol zou op inhoudelijk vlak ingevuld moeten worden door de software architect.

5.4 Zijn de condities aanwezig om het software ontwikkelingsproces kwalitatief juist uit te kunnen voeren?

De condities zijn grotendeels aanwezig om het software ontwikkelingsproces kwalitatief juist uit te voeren. Uitzondering hierop is ten eerste wederom de afwezigheid van een software architect die de kwaliteit en hoog niveau ontwerp van het systeem bewaakt en richting geeft. Daarnaast ontbreken enkele technische zaken:

- Er worden geen dagelijkse (of nachtelijke) automatische builds en regressie tests uitgevoerd op een centrale integratie omgeving, met uitzondering van een functionele regressietest suite voor AVI.
- Er zijn geen gedeelde coding standaarden of andere kwaliteitsafspraken die centraal bewaakt worden middels daarvoor geschikte tools
- Er is geen unieke code repository en/of versioning systeem die door het gehele team wordt gedeeld.

Ook de condities bij aanvang van het project, zoals het ontbreken van documentatie, de lage code kwaliteit, en ontbrekende tests, hebben de uitvoer van het project bemoeilijkt.

6 Conclusies en aanbevelingen

De vorige hoofdstukken geven de beoordeling van het BVH systeem en deelsystemen, en beantwoording van de onderzoeksvragen. Dit hoofdstuk vat de beoordeling samen in de conclusies en geeft aanbevelingen ter verbetering van BVH.

6.1 Conclusies

SIG heeft vastgesteld dat het BVH systeem zeer omvangrijk is en een hoge diversiteit kent in gebruikte technologieën. Met name de koppelingen tussen systeemonderdelen en met externe systemen zijn talrijk en op vele verschillende wijzen gerealiseerd. De kwaliteit van de systeemonderdelen, wanneer los beschouwd, is met een enkele uitzondering redelijk tot goed. Knelpunten zijn de relatief hoge code duplicatie en het gebrek aan regressietestsuites. Echter, het systeem als geheel is van lage kwaliteit en slecht onderhoudbaar. De belangrijkste zwakte is de algehele architectuur met verouderde technologie in het centrum en hoge diversiteit in satellietonderdelen en koppelingen. Dit komt ook tot uitdrukking in de documentatie van het systeem, die per systeemonderdeel is uitgewerkt, terwijl overzichtsdocumentatie over het gehele systeem ontbreekt.

Het BVH project kenmerkt zich in sterke mate door de problematiek van het op elkaar afstemmen van de verschillende systeemonderdelen en het herhaald operationeel maken van het gehele systeem. Echter, de huidige organisatiestructuur van het BVH project is voornamelijk gericht op ontwikkeling van de verscheidene deelsystemen. De taak van het bewaken van de algehele software architectuur, alsmede van het ontwikkelen, harmoniseren, en testen van de installatieprocedures zijn onvoldoende belegd.

6.2 Aanbevelingen

De aanbevelingen zijn uitgesplitst naar korte termijn (kan onmiddellijk mee worden begonnen), middellange termijn (kan binnen een jaar mee gestart worden), en langere termijn (pas na afronding van de uitrol).

6.2.1 Korte termijn

Op korte termijn kan het software voortbrengingsproces van BVH op enkele belangrijke punten worden verbeterd.

- Stroomlijn het software versie beheer door de broncode van de verschillende deelsystemen in één enkel versiebeheersysteem onder te brengen. Zorg ook dat de status verhelderd wordt van de verschillende generatoren en van open source bibliotheken waarvan de broncode in de BVH broncode is geïncorporeerd.
- Maak een centrale integratieomgeving beschikbaar voor ontwikkelaars zodat wijzigingen eenvoudig in samenhang getest kunnen worden. Hierbij is het van belang dat dergelijke unit integratie tests niet worden uitgesteld tot een testfase na de ontwikkelfase, maar dat al tijdens de ontwikkeling door de ontwikkelaars zelf gecontroleerd kan worden of wijzigingen in één unit negatieve effecten heeft op de samenhang met andere units.

- Verbeter het gebruik van regressietesting, met name door middel van *white box unit testing*. Met name voor de in C# en Java ontwikkelde systeemonderdelen dient gebruik gemaakt te worden van instrumenten voor het dagelijks (of nachtelijks) draaien van automatische unit test suites. Ook dient de dekkingsgraad van deze unit test suites centraal bewaakt te worden. Voor functionele (black-box) tests voor AVI is hiermee al een begin gemaakt.
- Verbeter de bewaking van code kwaliteit, bijvoorbeeld door het inzetten van code style checking tools en/of door het monitoren van de toename van (complexe) code omvang, code duplicatie, etc.
- Borg de kennisdeling over bepaalde componenten tussen teamleden. Hierbij moet vooral gedacht worden aan de BVH Core, ITX4Word en GUI componenten, die het risico kennen van kennisconcentratie bij enkele individuen of bij externe medewerkers.

Op het gebied van de organisatiestructuur van het BVH project verdient het aanbeveling om het toezicht op de algehele software architectuur te versterken, bijvoorbeeld door het toevoegen van een software architect naast de deelprojectmanager van het bouw team. Ook dienen de activiteiten van het UIT subteam beter verankerd te worden in de organisatie en het voortbrengingsproces. Het ontwikkelen en debuggen van installatiescripts wordt bijvoorbeeld bij voorkeur niet gedaan door een apart deelteam, maar wordt belegd bij de ontwikkelaars zelf.

De documentatie van het BVH systeem kan voor enkele deelsystemen verder worden verbeterd. Maar met name verdient het aanbeveling om overzichtsdocumentatie aan te leggen die het systeem als geheel beschrijft. Hier kan gedacht worden aan een (klein aantal) architectuurplaten die in één oogopslag de samenhang van de verschillende deelsystemen met elkaar en met externe systemen weergeeft.

Sinds mid-december 2007 is er een inspanning gaande voor het documenteren en vereenvoudigen van het installatieproces. Deze inspanning is bijzonder belangrijk en dient structureler belegd te worden dan nu het geval is. Ook is het van belang dat het installatieproces aandacht krijgt op het niveau van de algehele software architectuur. Een eventueel aan te stellen software architect zal dus tot taak moeten krijgen om toezicht te houden op de keuzes en wijzigingen die worden gedreven door verbeteringen in het installatieproces.

6.2.2 Middellange termijn

Op middellange termijn verdient het aanbeveling om specifieke inspanningen te verrichten om de diversiteit en complexiteit van het BVH systeem te reduceren. Hierbij kan aan de volgende maatregelen worden gedacht.

- Heroverweeg de koppeling tussen AVI en ITX die op dit moment plaatsvindt middels een zelfontwikkelde screenscraping methode. Deze koppeling veroorzaakt nu grote afhankelijkheid tussen deze twee componenten, waardoor wijzigingen in de één met zich meebrengen dat er tevens wijzigingen in de ander gepleegd moeten worden.

- Overweeg het vervangen van de ZoZ satelliet door een Java of C# implementatie, mogelijkerwijs als uitbreiding van één van de bestaande Java of C# satellieten.
- Verminder het gebruik van Accell tezamen met Unix shell scripts voor het realiseren van externe koppelingen. Overweeg in plaats hiervan het gebruik van Java of C#.

Tevens verdient het aanbeveling om een onderzoek te starten naar de mogelijkheden voor het opheffen van de afhankelijkheid van het BVH systeem van verouderde hardware en met name naar de impact hiervan op de software. In de systeem architectuur van BVH zijn HP Alpha Servers opgenomen, model ES45. HP accepteert sinds April 2007 geen bestellingen van nieuwe Alpha systemen meer. Eigenaren van Alphas konden tot April 2008 uitbreidingen bestellen. Op dit moment is ondersteuning beperkt tot “factory-refurbished” systemen en uitbreidingen, waarschijnlijk tot 2012. Hiermee is het opheffen van de afhankelijkheid van deze systemen geen acute noodzaak, maar kan op termijn wel belangrijk worden.

6.2.3 Lange termijn

Op langere termijn zal het van belang zijn om een strategie te bepalen voor vervanging en/of renovatie van het BVH systeem. Het huidige systeem leent zich niet goed voor functionele uitbreidingen of platform veranderingen.

Een belangrijke eerste stap in een dergelijke strategiebepaling is het definiëren van de essentiële services die het BVH systeem moet leveren.

Globaal gesproken dient het aanbeveling drie verschillende veranderingsstrategieën te onderzoeken.

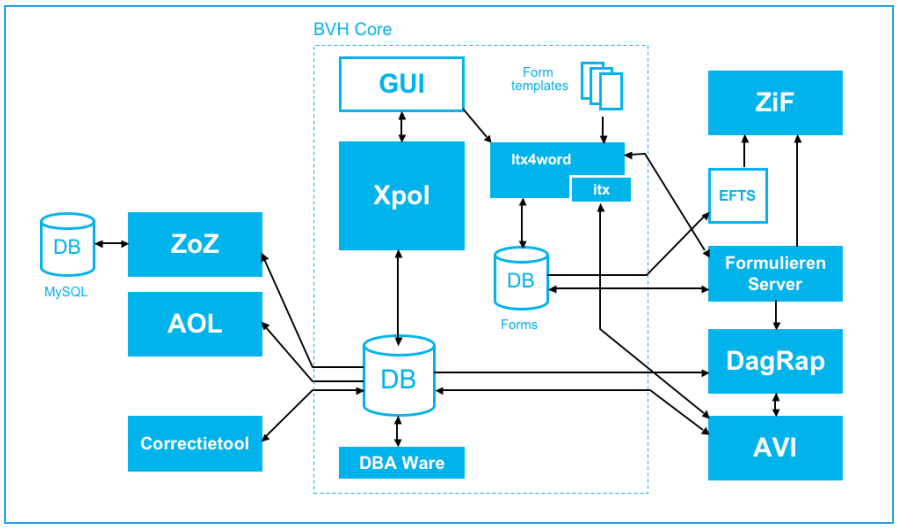
- Gradueel: verplaats functionaliteit stap voor stap vanuit de BVH core naar de satellieten, waarna het mogelijk wordt om de BVH database naar een nieuw platform te migreren en de Unify/Accell delen overbodig te maken.
- Geheel: selecteer een extern pakket en maak het geschikt voor het leveren van de BVH services. Dit vergt het onderzoeken van de mogelijkheden van bestaande pakketten, uitmondend in een gap analyse en een inschatting van de benodigde inspanning voor het geschikt maken. Hierbij dient niet onderschat te worden wat de hoeveelheid benodigd maatwerk zal zijn.
- Geheel nieuw in modernere technologie: ontwikkel het BVH systeem geheel opnieuw, maar gebruikmakend van een zeer beperkt aantal, moderne technologieën. De basis van deze strategie dient te zijn dat er heldere functionele eisen van het BVH systeem voorhanden zijn.

A. Appendix details van de waarnemingen

Deze appendix beschrijft de waarnemingen van het BVH systeem die zijn gedaan op basis van de analyse van de broncode, documentatie en interviews. De waarnemingen zijn onderverdeeld naar verschillende aspecten en abstractieniveaus en systeem onderdelen.

A.1 Systeem architectuur

De interne architectuur van het BVH systeem bestaat uit een aantal componenten.



Figuur 5: Interne architectuur van het BVH systeem.

Het BVH systeem is gebaseerd op de bestaande handhavingsapplicatie Xpol (nu ook BVH-core genoemd). Xpol is ontwikkeld in de vierde-generatie programmeertaal Accell en maakt gebruik van Sybase database technologie. Xpol biedt een karaktergeoriënteerde user interface (niet-grafische schermen).

In het BVH systeem is aan Xpol een grafische user interface (GUI) toegevoegd, gebruikmakend van zogenaamde *screen-scraping* technologie van Seagul. Er bestaat een redelijk directe correspondentie tussen de nieuwe grafische schermen en de bestaande niet-grafische schermen van de onderliggende Xpol applicatie. Als onderdeel van de GUI is een op MS-Word georiënteerde tekstverwerker opgenomen (ITX4Word). De landelijke standaard formulierenset is compatible gemaakt met deze tekstverwerker.

Bij Xpol horen een aantal satellietapplicaties:

- Aangifte Via Intranet (AVI): Een Java applicatie ontwikkeld met WebSphere. Aangepast om te werken met Oracle Application Server. Configureerbaar gemaakt om voor verschillende korpsen te worden ingezet.
- Dagrapportage (DagRap): Een VB6 applicatie opnieuw ontwikkeld in Java.
- Zoeken in Formulieren (ZiF): Een C# applicatie.
- Afspraak op Locatie (AoL): Een C# applicatie.
- Zicht op Zaken (ZoZ): Een PHP applicatie met MySQL database.

- Xpol Correctietool (XPCT): Een ASP.Net applicatie.

Deze satellietapplicaties werken direct op de Sybase database van Xpol. Daarnaast bestaan er enkele additionele koppelingen tussen de satellieten en de Xpol applicaties zelf. Door middel van database triggers wordt de *referential integrity* van de Sybase database bewaakt. Validaties (logica) die uitgevoerd worden door de Xpol applicatie zelf zijn in de satellietapplicaties gerepliceerd.

Met betrekking tot de beoordeling van de architectuur zijn er een aantal bevindingen die de onderhoudbaarbheid van de code negatief beïnvloeden.

- Veelvoud van technologieën.
- Verouderde technologie in de kern van de applicatie.
- Eigen database ZoZ
- Aantal generatoren met onduidelijke status
- Screenscraping (GUI/Xpol en AVI/ITX)

Het systeem kenmerkt zich door een groot aantal verschillende technologieën. Figuur 6 geeft een overzicht van het aantal regels code per gebruikte technologie. Naast dat het systeem erg omvangrijk is, vergt onderhoud aan het systeem een grote diversiteit aan kennis van bijvoorbeeld programmeertalen.

De kern van de applicatie, de Xpol core, is ontwikkeld in in de vierde generatie programmeertaal Accell. Accell is een verouderde ontwikkelomgeving en kent weinig actieve gebruikers. Ervaren programmeurs zijn erg moeilijk te vinden voor deze niet maintstream taal.

De satelliet AVI en de Xpol GUI gebruiken beide screenscraping. Bij screenscraping wordt de uitvoer van het beeldscherm ontleed om zo de data uit een ander systeem te halen. De Xpol GUI vertaalt zo de Accell omgeving naar een grafische omgeving en AVI ontleedt zo de data die op het beeldscherm door een user in formulieren ingevuld zou worden. Dit soort screenscraping is echter een risico omdat hierdoor een kwetsbare koppeling tussen delen van afzonderlijke programma's wordt ingebouwd. Voor de Xpol GUI moet nauwkeurig nagegaan worden in hoeverre wijzigingen in de Accell schermen ook daadwerkelijk de nodige veranderingen in de grafische schermen teweegbregen. AVI is kwetsbaar voor veranderingen in de gebruikte formulieren aangezien een kleine wijziging van een vraag kan leiden tot het haperen van de communicatie tussen AVI en de ITX formulierengenerator.

Voor de Xpol core met screen scraping bestaat een extra risico doordat de generatoren voor de vertaalslag van de Accell naar de GUI schermen niet actief worden onderhouden. De verantwoordelijkheid voor toekomstige wijzigingen aan deze generatoren is onduidelijk.

Het datamodel voor de koppeling met de C# satellieten is gegenereerd vanuit Powerdesigner data model via de Data Foundation Services (DFS) generatoren. Deze generato-

ren worden echter niet meer actief beheerd, en zijn momenteel niet direct inzetbaar voor het opnieuw genereren van de datalaag bij wijzigingen in het datamodel.

Een verder issue voor de onderhoudbaarheid van de Xpol applicatie is de eigen database van de ZoZ satelliet. De Xpol database wordt periodiek via mail geëxporteerd van de Xpol beheerder naar de ZoZ beheerder. De ZoZ satelliet werkt dus met een kopie van (een deel van) de de Xpol Database, en met een ad-hoc connectie in plaats van gebruik te maken van bestaande connecties naar de database. Hierdoor wordt onderhoud van de ZoZ satelliet moeilijk door het creeëren van een aparte database waarvan de overeenkomst van de structuur met de originele Xpol database bewaakt dient te worden, en moet er nog een additioneel connectie protocol met de Xpol database ondersteund worden.

A.1.1 Modularisatie

Bij modularisatie van het systeem wordt onderzocht of verschillende delen van het systeem verschillende functionaliteit hebben. Modularisatie wordt doordroken indien bijvoorbeeld verschillende delen van het systeem te sterk hetzelfde doen. Dit probleem komt naar voren bij de koppeling van de Java satellieten met de Xpol core, en in het bewaken van de referential integrity en de logische regels in de Xpol database.

De twee Java satellieten zijn verschillend gekoppeld met de Xpol database. AVI koppelt naar de de Xpol database via JDBC en EJB's terwijl Dagrap is gekoppeld via Toplink. Wijzigingen in de Xpol database structuur, bijvoorbeeld ten behoeve van de GSH, moeten daardoor op twee plekken en op verschillende manieren doorgevoerd worden.

De data in de Xpol database moet worden bewaakt op een groot aantal criteria, ook opde referential integrity van de database. Referential integrity houdt kortweg in dat gekoppelde tabellen in de database consistent en bruikbaar blijven. Het bewaken van de referential integrity en de logische regels is geïmplementeerd in de Accell code en aangevuld in de Xpol database zelf door middel van triggers. Tevens is het nodig om deze bewaking van de consistentie van de database gedeeltelijk in te bouwen in de satellieten. Dit maakt het correct aanpassen van de applicatie als geheel bij een wijziging van de database een moeilijke operatie die op meerdere plaatsen herhaald en gecontroleerd moet worden.

A.1.2 Separation of concerns

Het principe van 'Separation of Concerns' houdt in dat onderdelen in een systeem een specifieke taak uitvoeren. Het is een ontwerpprincipe dat stelt dat er geen overlap van functionaliteit is, maar dat er een heldere splitsing waar te nemen is. Het principe bevestigt dat we maar met één ding tegelijk bezig kunnen zijn en dat, om complexiteit in bedwang te houden, er een duidelijke plek voor alles is.

Binnen het BVH systeem is geconstateerd dat wijzigingen, bijvoorbeeld in de GSH of in de formulierenset, systeem-breed impact kunnen hebben. Veranderingen in de GSH en in de (layout van) de gebruikte formulieren kunnen op verschillende delen van het systeem tegelijkertijd wijzigingen afdwingen die allen nodig zijn om de applicatie als geheel te laten functioneren. Dit is met name een issue voor de satelliet AVI.

AVI zit in het spanningsveld tussen de Xpol database en de gebruikte formulieren vanuit de ITX4Word. Wijzigingen in de GSH versterken dit spanningsveld aan beide kanten. Door GSH kan enerzijds de Xpol database veranderen, waardoor de AVI satelliet moet worden aangepast. Tegelijkertijd kan een wijziging in de GSH zorgen voor een verandering in de vragen in de gebruikte formulieren, wat weer een moeilijk te voorspellen wijziging in de ITX4Word interactie voor AVI kan betekenen. Deze laatste wijziging kan moeilijk te doorgronden zijn door de screen-scraping/telnet interactie tussen AVI en ITX4Word. Door dit alles tezamen kan een simpele verandering van een veld zorgen voor een niet werkend systeem en dat de oorzaak moeilijk op te sporen en te repareren is.

A.2 Code level design

In deze sectie wordt het code level design besproken. Achtereenvolgens komen aanbod: de omvang van de code, de complexiteit van de code, duplicatie in de code, lengte van de methodes. Deze metingen zijn toegespitst op de Java en C# satellieten.

A.2.1 Omvang van de broncode en gebruikte technologieën

De omvang van de broncode is gemeten naar het aantal regels code in de broncode bestanden met en zonder meeneming van lege regels en commentaar. In Figuur 6 staat een overzicht van het aantal regels code per broncode type.

	Xpol	XPTC	GUI	itx4wd	FrmSv	AVI	DagRap	ZiF	ZoZ	AoL	DFS	GLog
Accell	866K											
C	57K		2K	31K								
shell	*38K*											
VBA			1K				2k					
xslt			6K									
Java						110K	16K					
JSP						*20K*	*7K*					
C#		11K		29K	9K			7K		4K	23K	5K
SQL	*153K*											
PHP									*24K*			
JS						*5K*	*11K*		*7K*			
ITX.fo	75K											

Figuur 6: Aantal regels code per broncode type.

Het grootste deel van het systeem, gemeten in regels broncode, bestaat uit de Xpol core met de Accell code. Java, C#, en SQL zijn ook sterk vertegenwoordigd.

A.2.2 Complexiteit

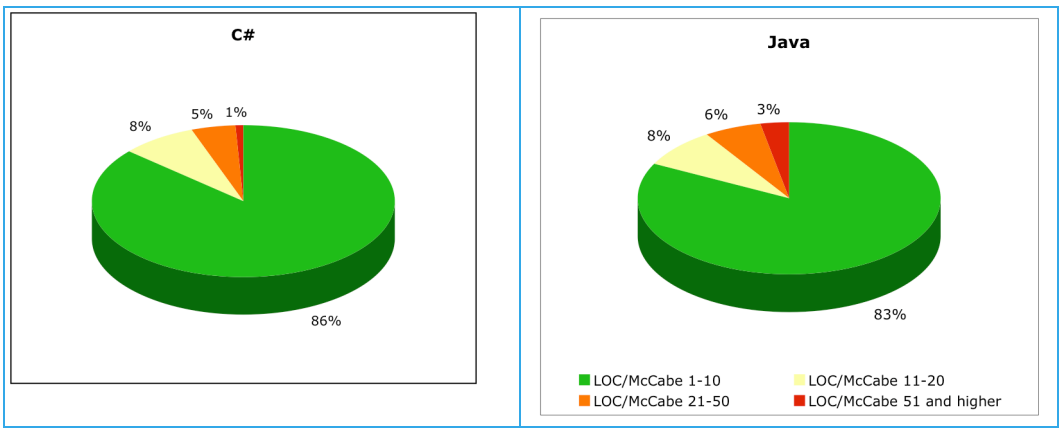
Voor het meten van de complexiteit van de broncode van een systeem wordt gebruik gemaakt van de zogenaamde McCabe-index. De McCabe-index wordt berekend voor elk kleinste onderdeel van een systeem dat nog als eenheid kan worden uitgevoerd en getest. Voor VB Java en C# is dit een procedure. De McCabe-index is een maat voor de complexiteit van een systeem op het niveau van de broncode: hoe hoger de McCabe-index van een programma of procedure, hoe meer executiepaden er mogelijk zijn.

Tabel 21 toont een algemeen geaccepteerde classificatie van de risico's van broncode in relatie tot de McCabe-index (bron: The Software Engineering Institute, SEI):

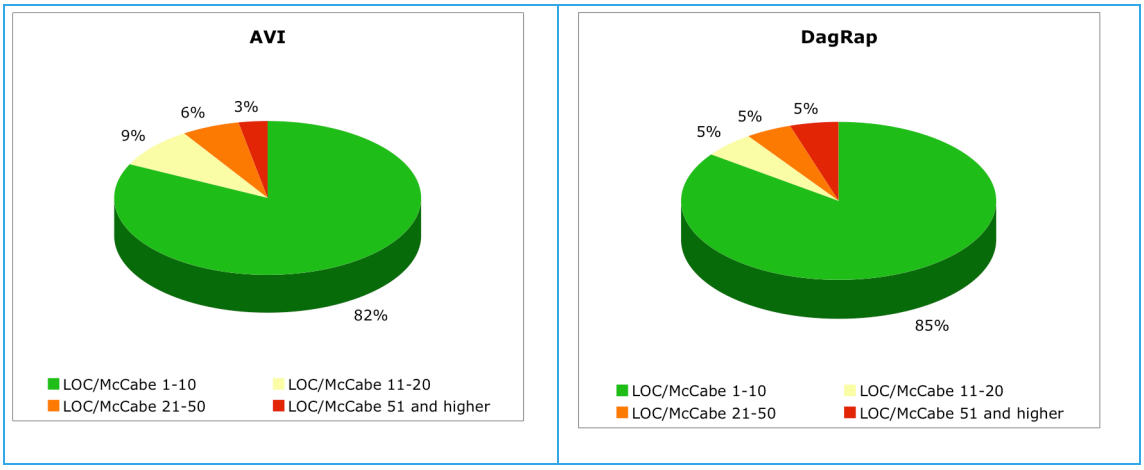
McCabe-index	Risico
1-10	Eenvoudige code, klein risico
11-20	Enigszins complexe code, beperkt risico
21-50	Complexe code, hoog risico
> 50	Ontestbare code, zeer hoog risico

Tabel 21: Risicoclassificatie voor broncode op basis van de McCabe-index.

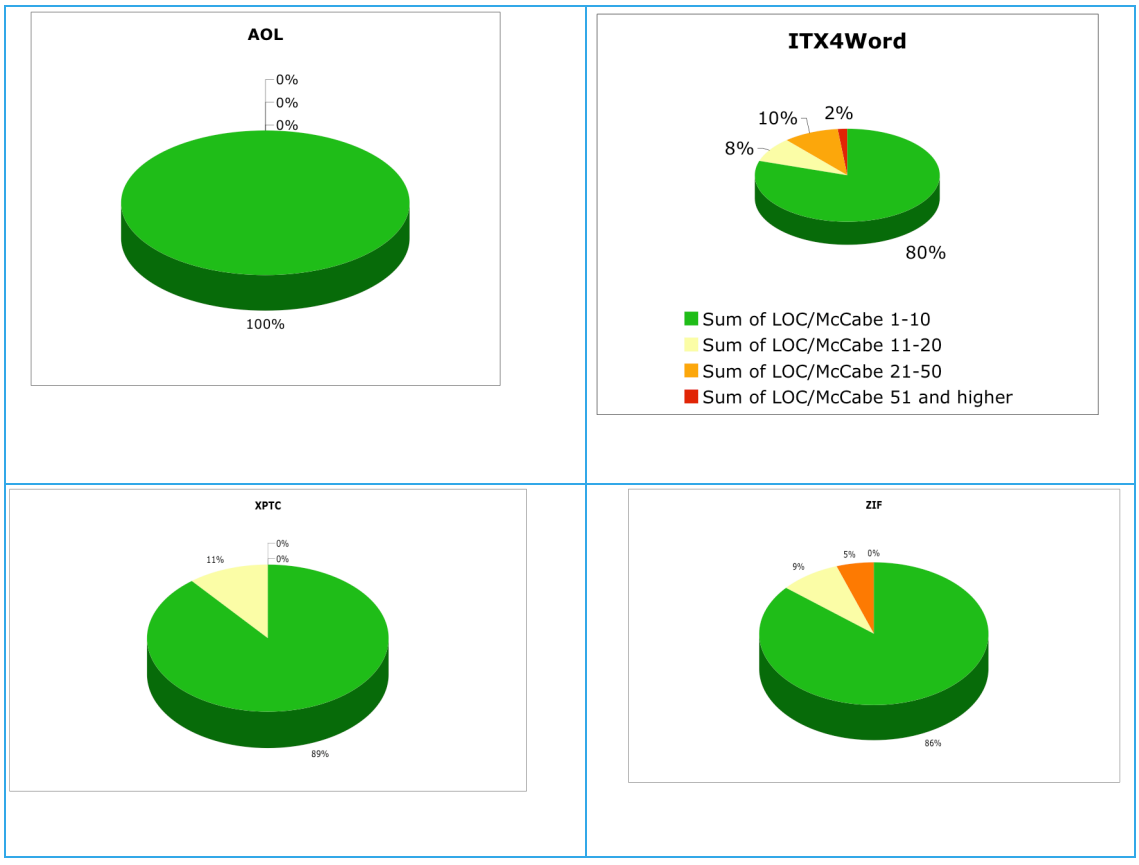
De complexiteit is gemeten voor de verschillende onderdelen van de code, onderstaande Figuur 7 geeft het aandeel complexe regels broncode weer voor de satellieten. Figuur 8 geeft de complexiteit in meer detail voor de afzonderlijke Java satellieten. Figuur 9 geeft de complexiteit weer voor de C# satellieten.

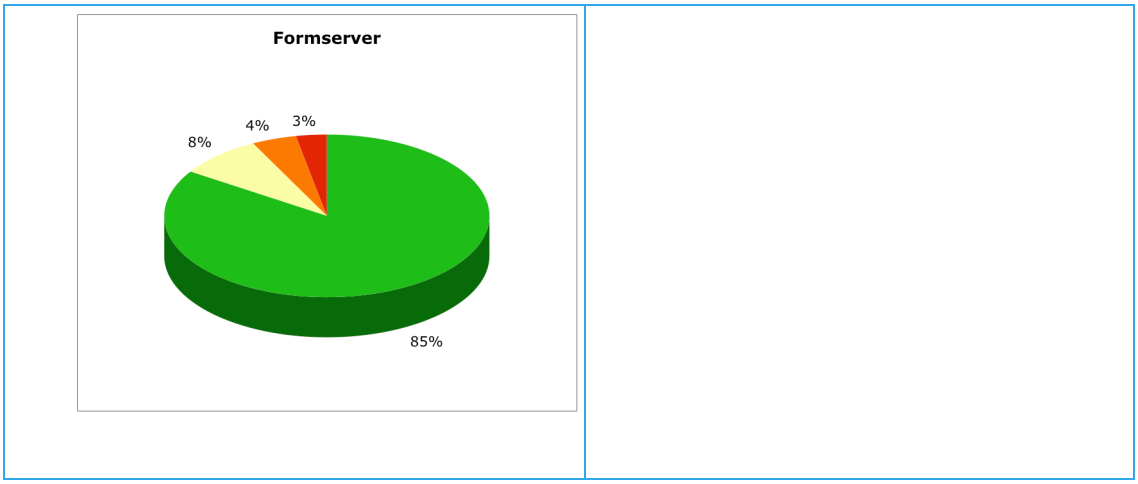


Figuur 7: Complexiteit C# code en de Java code.



Figuur 8, Complexiteit van de Java satellieten



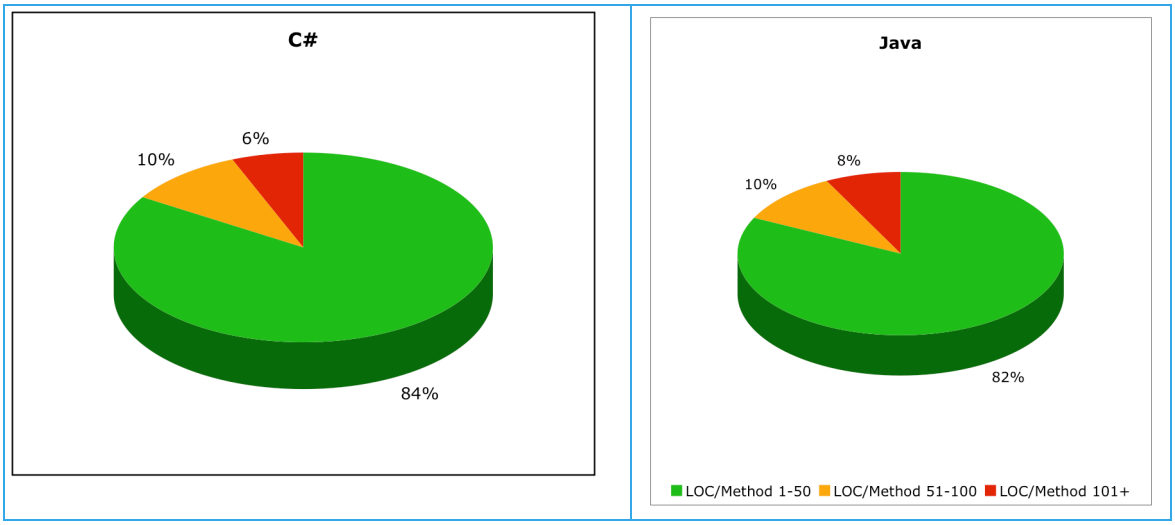


Figuur 9, Complexiteit voor de C# satellieten

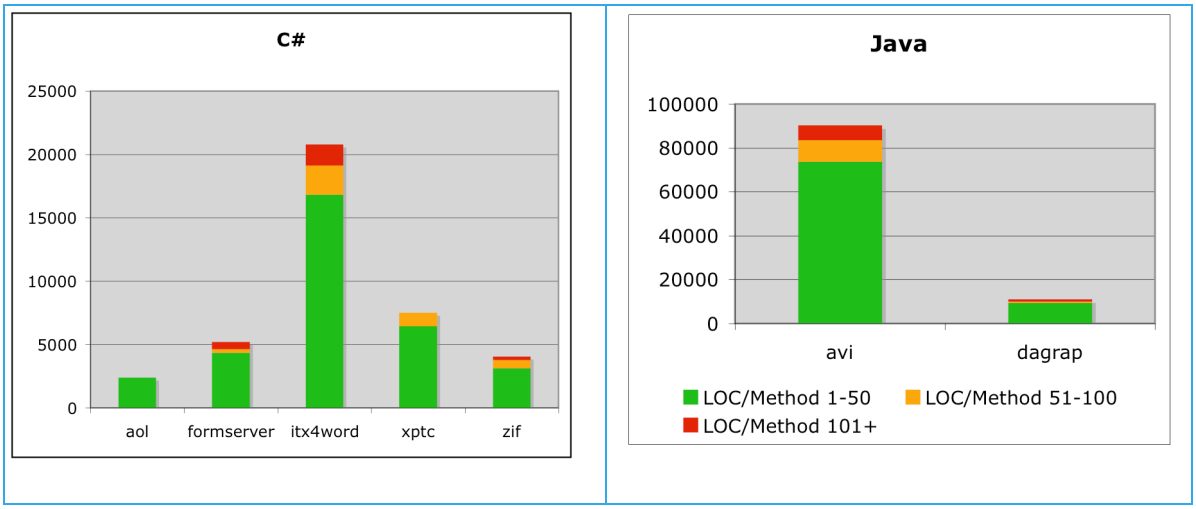
A.2.3 Lengte van procedures

De lengte van de procedures in de broncode wordt gemeten aan de hand van geschreven regels code (die zijn ontdaan van commentaar en lege regels). Lange units zijn slechter analyseerbaar, de lengte maakt het minder eenvoudig te overzien. Er is normaal een significante correlatie tussen het aantal regels in een unit en de complexiteit van de unit.

Onderstaande grafieken in Figuur 10 geven een overzicht van de lengte van de procedures voor de C# en Java code in de satellieten. Per categorie in de grafiek is het aantal regels code (LOC) geteld waarvan de methode in die categorie valt. Figuur 11geeft een detail beeld per satelliet.



Figuur 10: Lengte van de procedures in C# en Java broncode van het systeem.



Figuur 11, overzicht van de methode lengte per satelliet

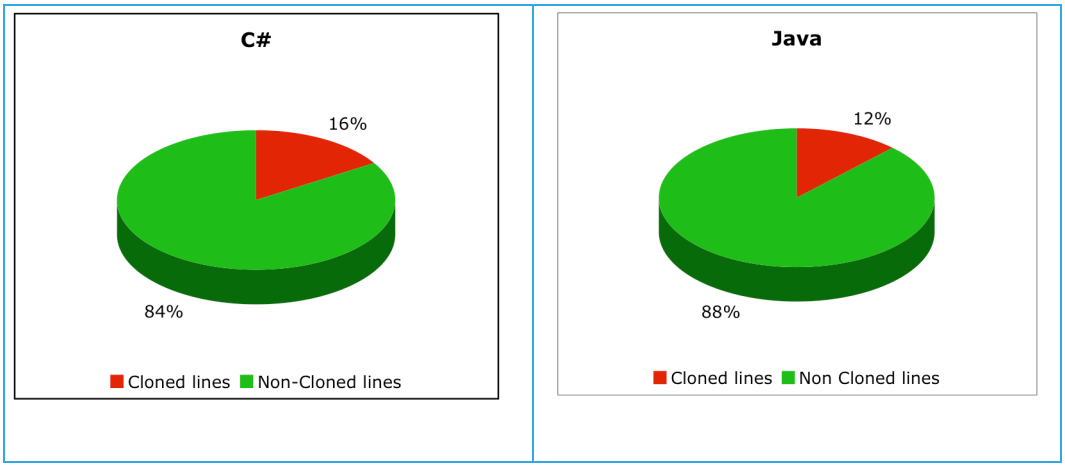
A.2.4 Codeduplicatie

Ieder software systeem bevat een zekere hoeveelheid gedupliceerde code. Code is gedupliceerd wanneer op meerdere plaatsen in het systeem dezelfde regels code voorkomen. Een codeblok wordt als duplicaat geteld indien

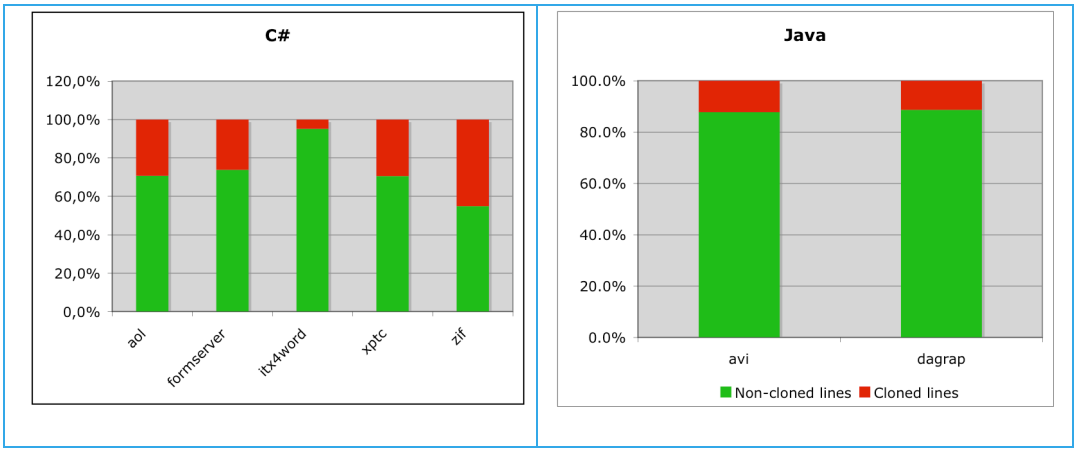
a. het uit minimaal 6 aaneengesloten regels bestaat
b. het codeblok op meerdere plaatsen in het systeem wordt aangetroffen

Gedupliceerde code is niet wenselijk omdat dit de onderhoudbaarheid verslechterd. Aanpassingen in gedupliceerde code moet op meerdere plekken aangepast worden, met het risico dat een of meerdere plekken over het hoofd worden gezien. Het is daarom wenselijk duplicatie zo laag mogelijk te houden.
Bij het vaststellen van het codeduplicatie-percentage zijn alle kopieën, inclusief het originele codeblok als gedupliceerde regels geteld.

Het resultaat van de duplicatiemeting is weergegeven in Figuur 12. De duplicatiepercentages zijn berekend ten opzichte van het totaal aantal regels in de broncode per onderdeel van het systeem. Figuur 13 geeft een detailbeeld per satelliet.



Figuur 12: Broncode duplicatie in de broncode van het BVH systeem.

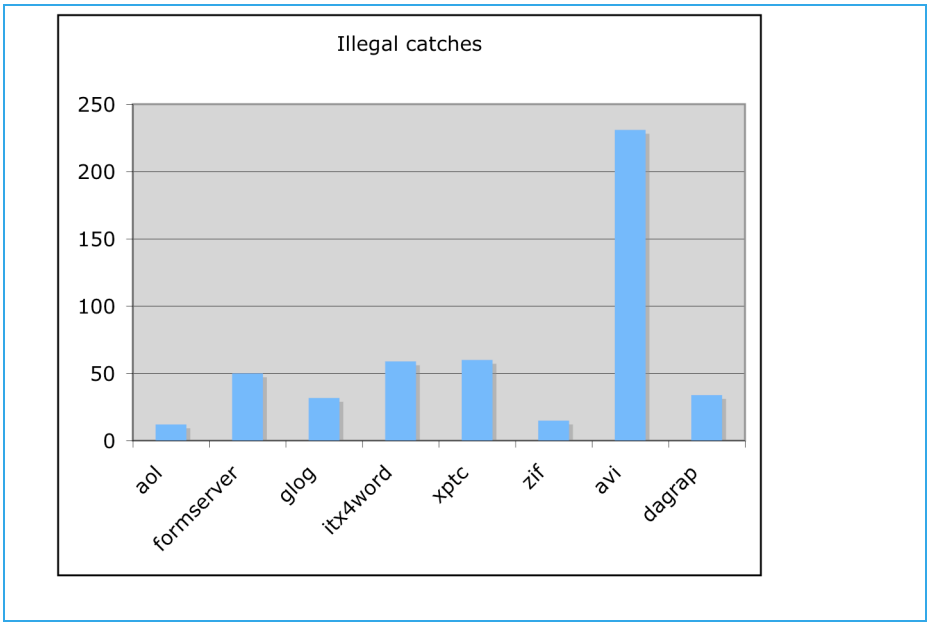


Figuur 13, Overzicht van de duplicatie per satellieten

A.2.5 Exception handling

Exception handling is het tijdens de executie afvangen van fouten en daarop de juiste acties uitvoeren. Excepties kunnen ontstaan door verwachte of onverwachte problemen tijdens het draaien van programma's. Stabiele code heeft als kenmerk dat de laatste categorie minimaal is. Een goed exception handling mechanisme maakt code stabiel en robuust voor aanpassingen in de code. Populair gezegd: als er iets misgaat dan heeft de programmeur voorzien wat er misgaat.

Figuur 14 geeft een overzicht van de verkeerde foutafhandeling in de broncode van het BVH systeem. De "illegal catches" zijn verkeerd gebruikte generieke foutafhandelingen op C# en Java taalniveau, waar specifieke, toepassingsafhankelijk e methoden zouden moeten worden gebruikt.



Figuur 14, overzicht van verkeerde foutafhandeling in BVH

A.2.6 Unit-test

Geautomatiseerd unit testen is tegenwoordig een algemeen geaccepteerd gebruik in de ontwikkeling van software. De ontwikkelaar schrijft daarbij zelf tests voor de ontwikkelde code. Doordat code en test geschreven worden op basis van de interne structuur van de code in plaats van het externe gedrag van de code wordt er ook wel gesproken van whitebox testing. Deze techniek is de tweede in de rij: code review, whitebox testing, blackbox testing en acceptatie tests als succesvolle techniek om fouten te voorkomen. De unit-test techniek heeft een extra documenterende waarde voor programmeurs. Tevens bevordert het gebruik van unit-tests dat de code is opgesplitst in kleine behapbare delen functionaliteit. Dit komt de onderhoudbaarheid ten goede.

Unit tests zijn niet gebruikt tijdens de nieuwe werkzaamheden aan de satellieten, behalve voor AVI en DagRap. Unit Tests zijn gebruikt voor de nieuwe onderdelen van AVI, en minimaal voor de nieuwbouw van DagRap.

A.3 Aangeleverde broncode

In de onderstaande tabel zijn de versienummers aangegeven van systeemonderdelen waarvan broncode is aangeleverd voor dit onderzoek. De versie informatie is apart aangeleverd via e-mail en is daarna aan de hand van de broncode gevalideerd.

Systeem onderdeel	Nulmeting	Huidige verie
BVH core	2004b phase 1	1.0.3
Xpol Correctie Tool	1.0	2008.2.1.9
AVI	2.4.3	1.0.3
Zoeken in Formulieren	1.0	2008.2.1.7
Zicht op Zaken	n.v.t.	2008.1.0.7
Afspraak op Locatie	n.v.t.	2007.1.0.2
Dag Rapportage	n.v.t..	1.0.1
ITX4Word	n.v.t.	1.0.0.17
Xpol GUI	n.v.t.	1.1 build 2097
FormServer	n.v.t.	2008.1.0.4
DFS	2.8.3.15	2007.2.9.2
GLOG	2.0	2007.2.1.3

Tabel 22, Overzicht van aangeleverde broncode.

Als peildatum voor de nulmeting is de project aanvangsdatum (april 2007) aangehouden. Dit impliceert dat wijzigingen die in de voorbereidende fase zijn gemaakt, zoals het verwijderen van afhankelijkheden op meer dan één versie van bepaalde componenten, buiten de scope van de assessment vallen.