



B O N C O D E

Code analyse SPOT Rapportage

Customer: GVB

Project: SPOT

21 april 2021

J. Meetsma

j.meetsma@boncode.nl

Gaining objective insights

TO CREATE BALANCE

- Objective insights are vital to long-term success
- Without all relevant facts on the table, it's hard to make meaningful choices
- Create balance between short-term and long-term priorities in an objective way



Topics

Onderzoeksvragen

Management samenvatting

Context

Impressie

Bevindingen en conclusies

Risico's

Aanbevelingen



Onderzoeksvragen

- “Onderwerp de SPOT-applicatie aan een Code Review en voorzie GVB van bevindingen, aanbevelingen en risico's aangaande de codekwaliteit en doorgevoerde software architectuur.”
- Afgeleide vragen voor dit onderzoek met het oog op de komende aanbesteding:
 - Wat is de **onderhoudbaarheid*** van de huidige codebase?
 - Is de codebase (inclusief documentatie) kwalitatief goed genoeg om het beheer en onderhoud (inclusief doorontwikkeling) van SPOT door de huidige leverancier of door een andere partij voor een periode van tenminste 5 jaar te kunnen laten uitvoeren?
 - Focus op **analyseerbaarheid** en **wijzigbaarheid**.
 - Welke verbeteringen dienen eventueel te worden doorgevoerd om dit verantwoord te laten doen?

**) De kwaliteitseigenschap 'Onderhoudbaarheid' van de ISO 25010 norm luidt: “De mate waarin een product of systeem effectief en efficiënt gewijzigd kan worden door de aangewezen beheerders.”*



Management samenvatting 1 / 2

De code van SPOT is op zichzelf redelijk onderhoudbaar door ingevoerde ontwikkelaars; rating: **68**/100; predicaat: **Adequate**.

Pluspunten:

- De code is consistent en weinig complex in termen van aantallen doorlooppaden (compl. rating **95**).
- De code is goed verdeeld over componenten, taken en verantwoordelijkheden zijn goed gelokaliseerd (dependency span rating **66** en **geen** cyclische afhankelijkheden).
- Met name documentatie (SPOT-HLOB) t.b.v. beheer oogt verzorgd.

Minpunten:

- Code is low-level uitgeschreven, (patroonduplicatie **46%**) Met vermenging van domeinlogica (veel jargon en specifieke coderingen) en technische logica, dit geldt voor:
 1. Berichtafhandeling; Impliceert diep vereiste kennis van berichtformaten en ontwerpbeslissingen aangaande conversie.
 2. Assemblage en configuratie van run-time componenten: rol en samenhang van componenten, alsmede de gevolgde weg van datastromen is lastig te analyseren.
- Concept “keten” of datastroom is onvoldoende uitgewerkt in architectuur, code en test suite.



Management samenvatting 2 / 2

Belangrijkste conclusies:

- Het domein is complex. Dit betreft met name:
Mapping van verschillende berichtformaten o.b.v. internationale, nationale en GVB-specifieke standaarden met soms semantisch afwijkende onderdelen.
- Concepten en gehanteerde frameworks en 3rd party componenten zijn passend voor het domein maar divers en niet geheel gangbaar: dit vergt inleertijd voor de gemiddelde Java ontwikkelaar.
- Beheer, onderhoud en doorontwikkeling van SPOT is **lastig overdraagbaar**.

Grootste risico:

- Grootste risico is dat na overdracht de onderhoudbaarheid snel verslechtert doordat het systeem lastig te analyseren is, dat leidt tot ad hoc aanpassingen naar eigen inzicht, waardoor de analyseerbaarheid en productiviteit weer verder afnemen en het systeem ongewenst gedrag gaat vertonen.
- Risico op een zichzelf versterkend negatief effect.

Belangrijkste aanbevelingen:

- Verbeter de analyseerbaarheid door verschillende datastromen of ketens conceptueel uit te werken en te illustreren in documentatie.
- Verbeter de wijzigbaarheid met risk-based, ketengerichte regressietests met een korte feedback cyclus. Dit dient tevens als low-level documentatie.



Context

Scope, bronnen, onderzoeksstappen



Scope & deliverables

In scope:

- Java broncode: geautomatiseerde analyse en handmatige inspectie.
 - Test code volgens Maven conventie apart doorgemeten.
 - Test (utility) code in productiecode als test code beschouwd.
 - Code aangeduid met “tools” buiten scope gelaten.
 - Code in entity packages beschouwd als gegenereerd en buiten scope gelaten.
- Inventarisatie overige technologieën.
- Aangeleverde documentatie.

Deliverables:

- Een eindpresentatie in PowerPoint met daarin opgenomen de doelstelling van de opdracht, de onderzoeksvragen, bevindingen, aanbevelingen en mogelijke risico's.
- Overzicht architectuur c.q. topologie van het systeem.
- Dashboard op basis waarvan ontwikkelaars verbeteringen door kunnen voeren.

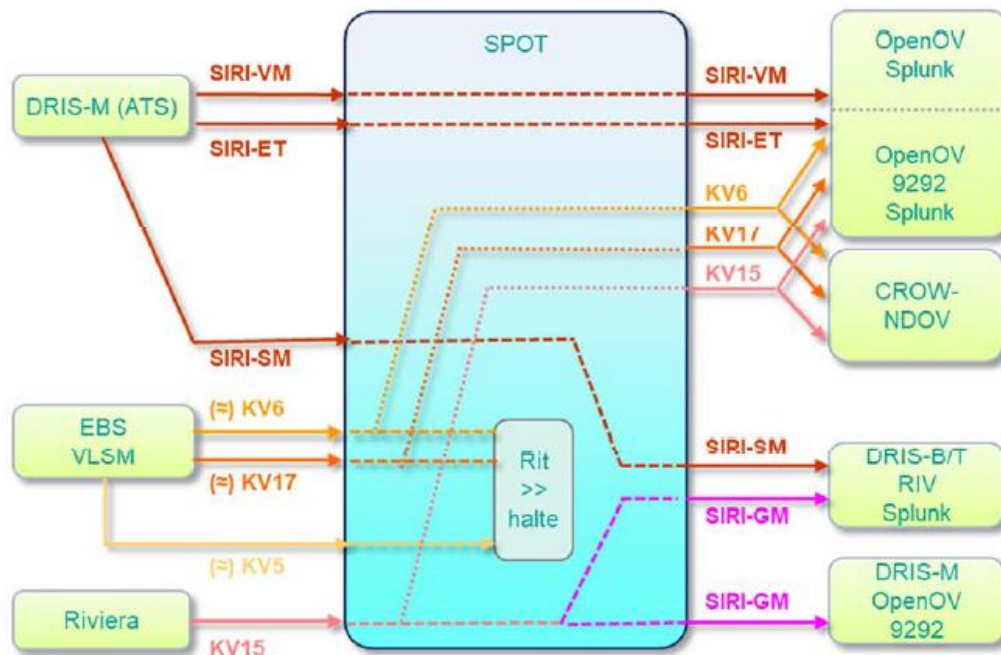


Sources

Source	Description	Date	Involved
SPOT IRS SIRI-client_services 1.4.3.docx		27-2-2021	Jetro Veenstra
SPOT SDD V2.2.1.docx			” ”
SPOT-DBDD_v24.docx			” ”
spot.zip	Main sources	23-3-2021	Erwin van Dijk
spotcommon.zip	Additional (re)sources		” ”
HLOB SPOT v1.8 definitief.pdf			Erwin van Dijk
SDD SPOT V2.4 4.pdf			” ”
SPOT IRS SIRI-client_services 1.4.3.pdf			” ”
SPOT SIRI use cases_v11.pdf			” ”
SPOT-DBDD_v25.pdf			” ”
SPOT-SVD_4.0.0.pdf			” ”

Achtergrond

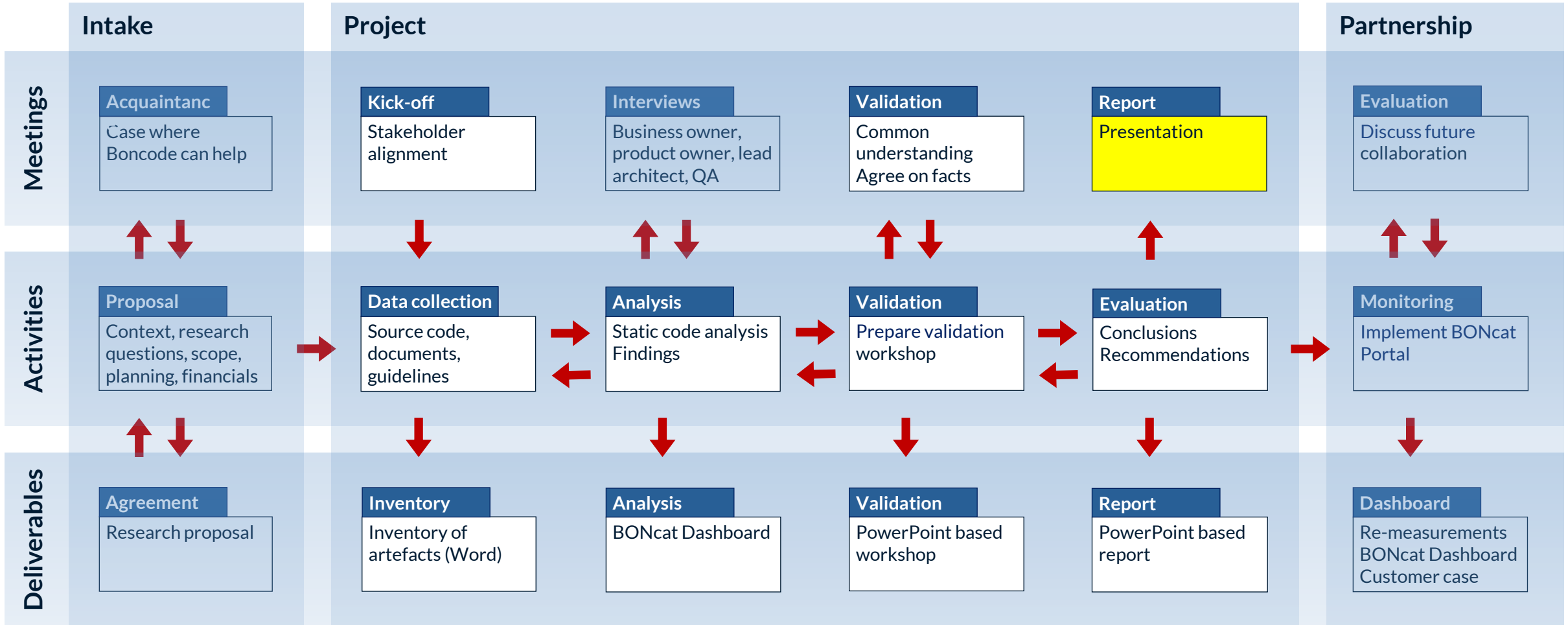
- “SPOT is een software pakket ontwikkeld voor het GVB, waarmee alle datastromen die het GVB naar afnemers verstuurt, via één systeem verstuurd en gemonitord worden. Dit systeem is opgedeeld in verschillende deel-systemen, die elk hun eigen datastromen verwerken.”



Taken SPOT:

- Transformeren,
- verrijken,
- bufferen,
- en routeren van berichten m.b.t. reizigersinformatie.
- Faciliteren van monitoring en besturing van berichtenverkeer / datastromen middels GUI.

Huidige stap in onderzoeksproces



Doelstellingen gezamenlijke processtappen

Kick-off

- Alle betrokkenen informeren over de aanpak voor het onderzoek, de input die van eenieder verwacht wordt en een korte introductie in het vakgebied van software metriecken.

Interview(s)

- Inzage krijgen in projectverloop, software development processen (en zo mogelijk businessplannen en strategie).

Validatiesessie

- Gedetailleerde uitleg statische code analyse.
- Draagvlak / gezamenlijk beeld realiseren bij de betrokkenen voor de uitkomsten van het onderzoek.
- Presenteren en valideren bevindingen, onder andere aan de hand van voorbeelden.
- Vaststellen eventuele *false positives*.

Rapportage

- Tussentijdse impressie van stand van zaken.
- Eindrapportage met beantwoording onderzoeksvragen, belangrijkste bevindingen, conclusies en aanbevelingen.



Bevindingen en conclusies

Static code analysis



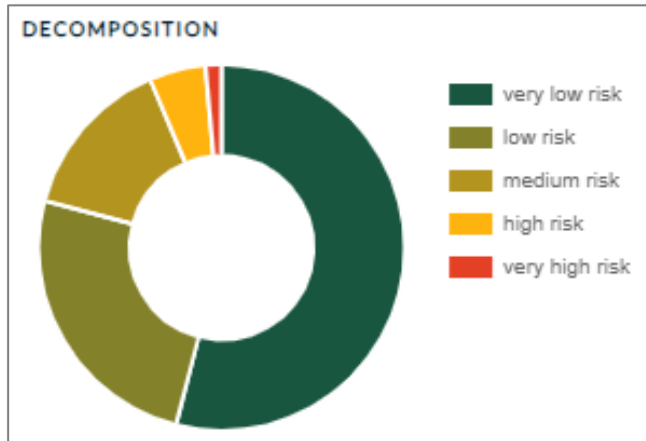
Impressie

- De functionaliteit van SPOT houdt het midden tussen analogie van een postkantoor en een distributiecentrum en heeft kenmerken van een ETL-oplossing*.
- De afhankelijkheid op gegenereerde code o.b.v. elders bepaalde berichtformaten (SIRI en BISON) beperkt de vrijheidsgraden van de oplossing. Dit leidt tot repeterende logica in berichtafhandeling, gemeten als patroonduplicatie.
- BonCode mist beschrijvende code die high-level concepten koppelt aan (delegeert naar) low-level uitwerking; dit impliceert dat ontwikkelaars én veel domeinkennis moeten hebben én het ontwerp doorgrond moeten hebben voordat ze de werking kunnen begrijpen.
- (Een refactorslag om code op een hoger niveau begrijpelijk te maken is blijkbaar achterwege gebleven.)
- De code bevat op talloze plekken indicaties van “onaf werk”.
- Documentatie is niet af en loopt uit de pas met implementatie.
- SPOT is onder hoge projectdruk gerealiseerd.

*) https://nl.wikipedia.org/wiki/Extraction,_Transformation_and_Load



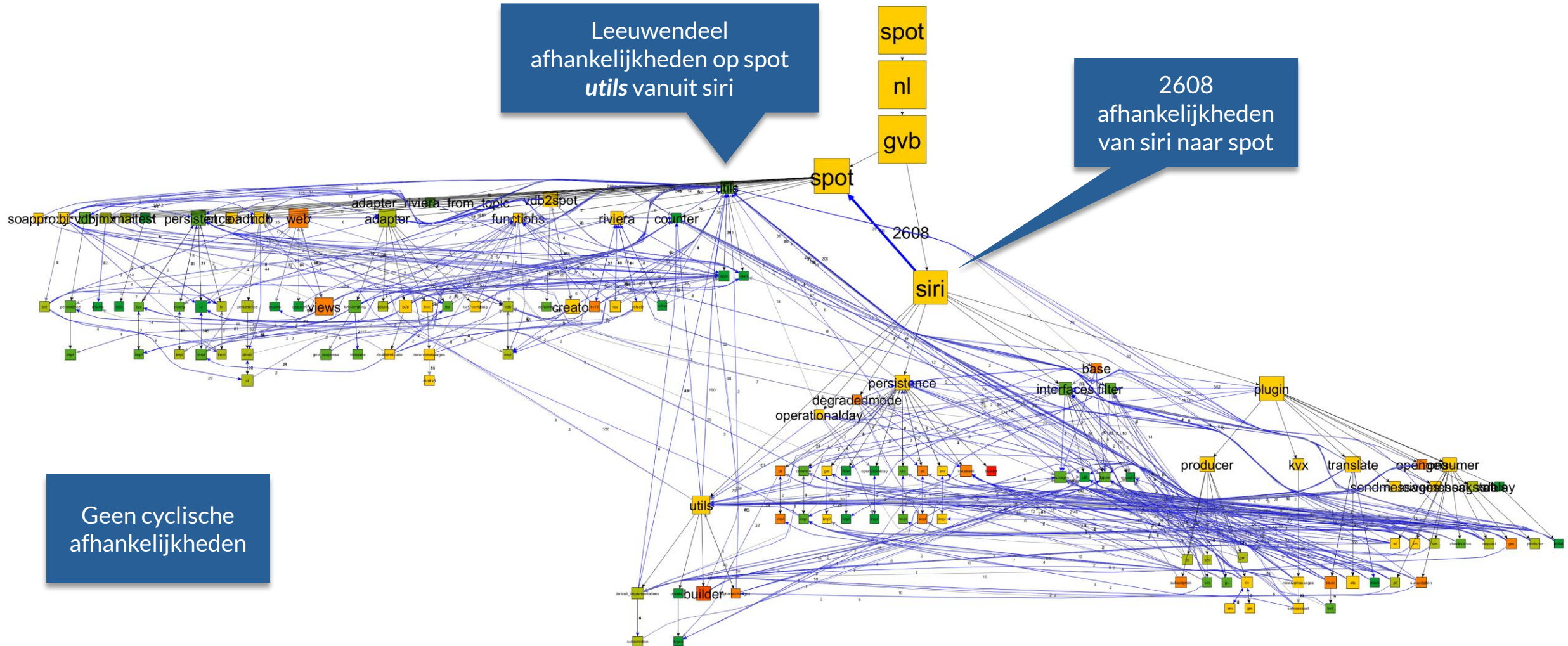
SPOT Decomposition rating is 68 /100



DECOMPOSITION	
Weighted Decomposition Rating	67.97
Function Parameter Rating	92.49
Function Size Rating	46.37
Function Complexity Rating	94.98
Dependency Span Rating	65.52
Dependency Volume Rating	54.78

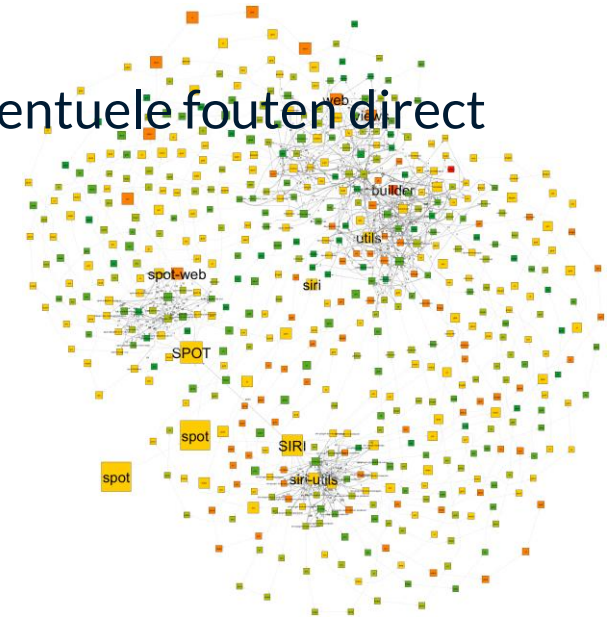
- Overall rating kwalificeert onderhoudbaarheid als **adequate**
- Subratings geven indicatie dat
 1. code zich bevindt in relatief:
 - grote classes
 - en lange methodes
 2. code niet complex is c.q. uitgeschreven in reeksen van op zichzelf eenvoudige statements
- 94 % van de code is aangemerkt als laag tot medium risicovol
 - 1,4 % als zeer hoog risicovol

SPOT dependency graph



Samenvatting bevindingen afhankelijkheden

- Modules hebben in het algemeen afhankelijkheden naar specifieke onderdelen binnen de code base of libraries.
 - De code base is niet sterk verweven maar juist goed gemodulariseerd.
 - Dit betekent dat wijzigingen doorgaans:
 - of “lokaal” impact hebben.
 - ofwel (bijvoorbeeld in utilities) overal impact hebben, waarbij eventuele fouten direct zichtbaar worden.
- 



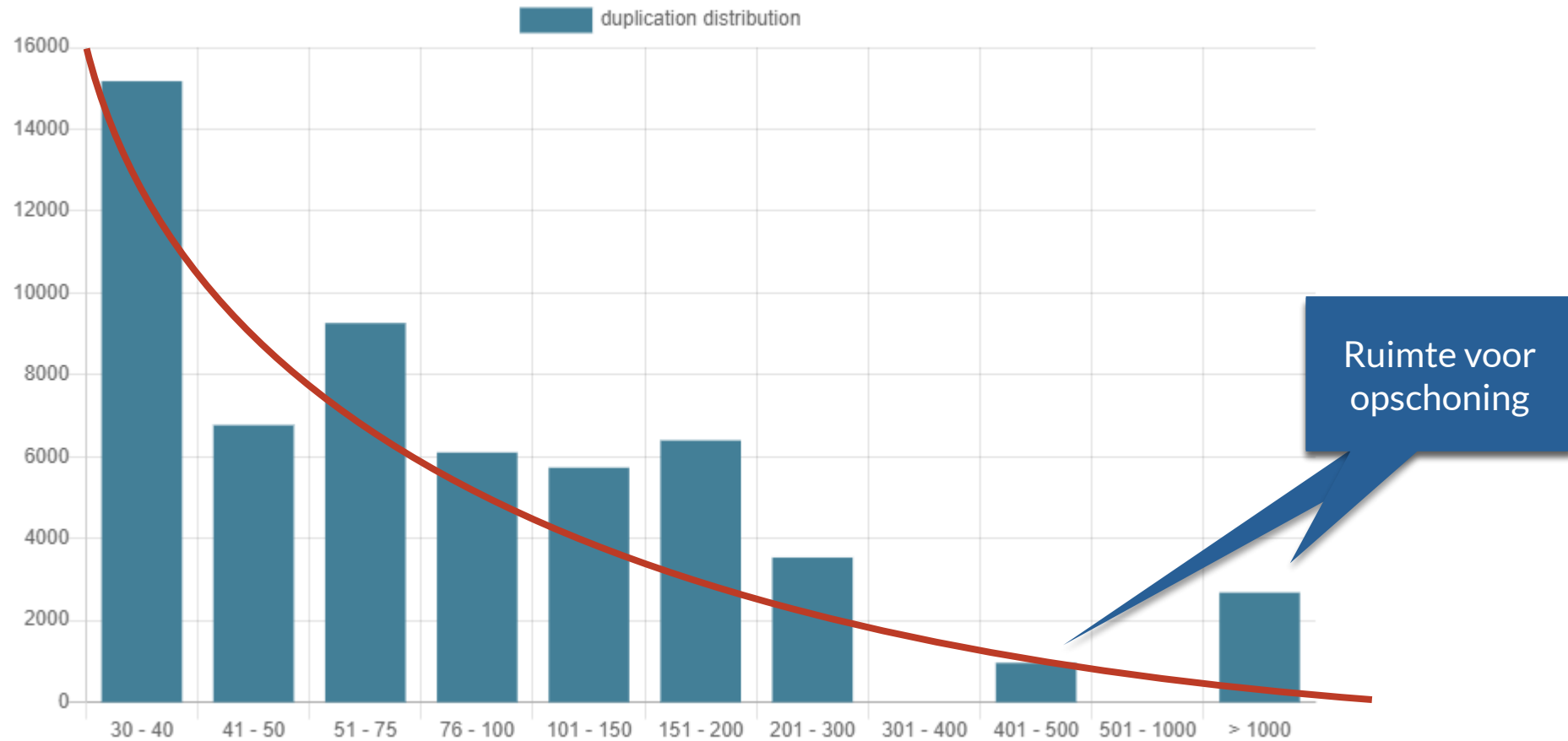
SPOT volume & duplicatie

- Volume 50 kLOC, kwalificatie klein tot middelgrote applicatie
- 17 % van code aangemerkt als duplicaat
- 45 % van de code aangemerkt als onderdeel patroonduplicatie
 - **Code bevat veel lang uitgeschreven repeterende statements / structuren**
- Aantal doorlooppaden door methods 7.640
- 17.839 LOC test code (1 regel test code op op 2,8 regels productie code)
- 1.558 afgeteste situaties (1 assert op 5 doorlooppaden)
- 280 bestanden
- Handmatig regressietesten vereist.

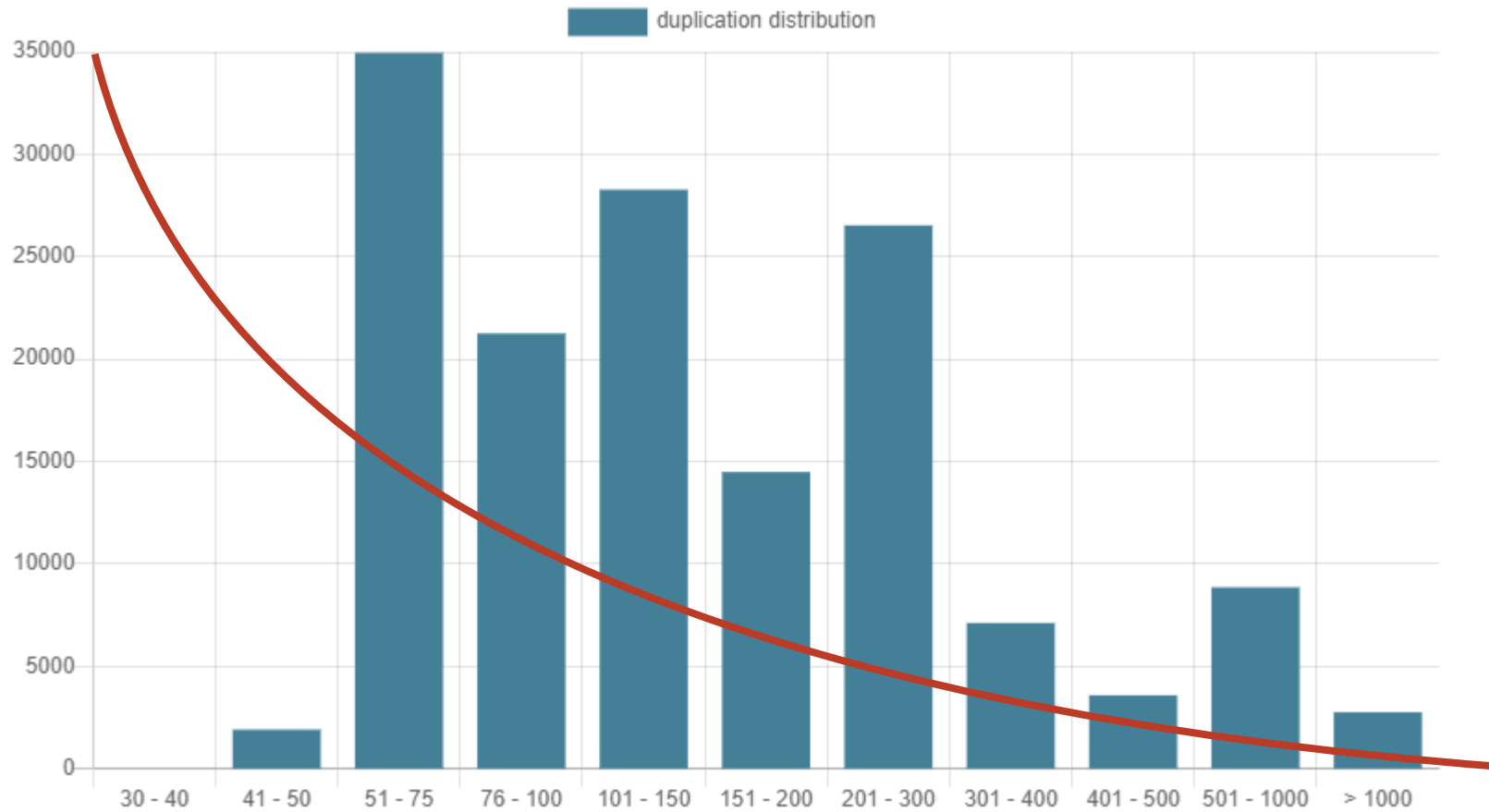
VOLUME	
Volume In Lines Of Code	49470
Volume In Lines	62384
Lines Of Comment	5003
Volume In Tokens	398274
Duplicate Tokens Percentage	17.2 %
Duplicate Pattern Tokens Percentage	45.5 %



17% duplicatie, deels in grote blokken



45% patroonduplicatie, veel in grote blokken



Word cloud naamgeving (identifiers)



Naamgeving
illustreert
meer techniek
dan domein



Overige bevindingen

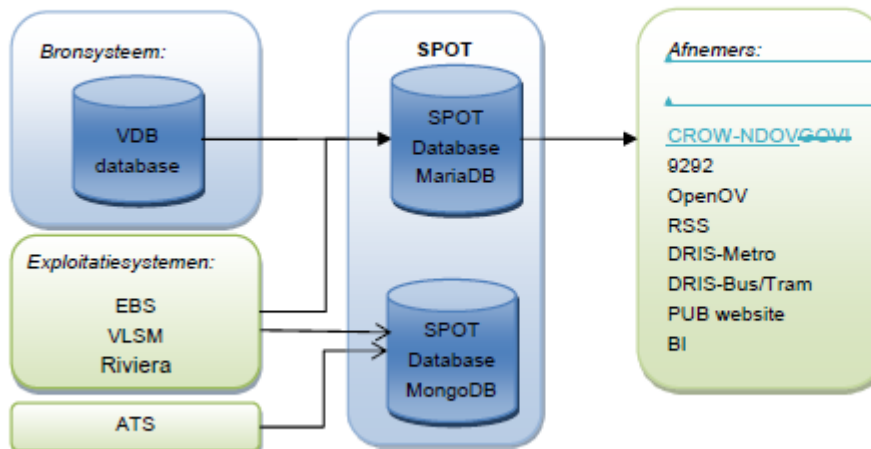
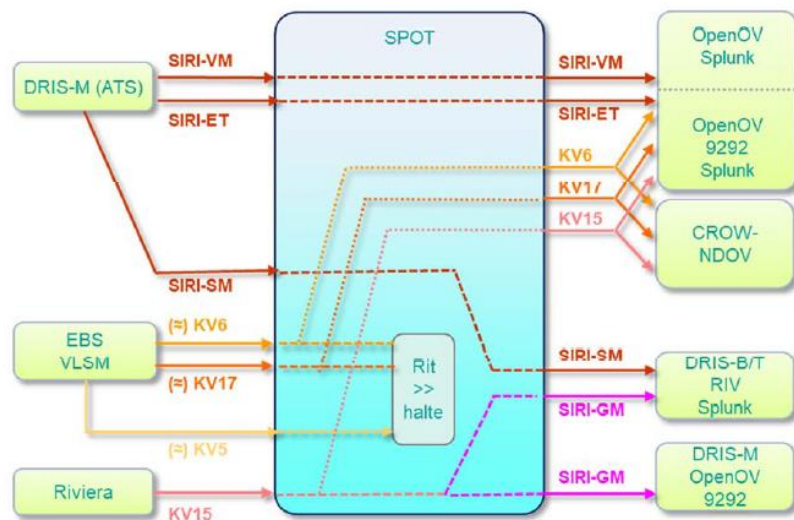
- Wachtwoorden m.b.t. infrastructuur en databases zijn plain text opgeslagen (en raadbaar).
- Database wordt handmatig bijgewerkt; handmatig deployment proces.
- Code bevat enkele hardcoded URLs.
- Logging en foutafhandeling is in het algemeen consistent en afdoende.
 - M.b.t. uniformiteit logging: er is in de loop der tijd een tussenschakel toegevoegd (geweest). Dit kan vragen oproepen bij onderhoud of bij toekomstige integratie van libraries met een alternatieve vorm van logging.
 - M.b.t. foutafhandeling: in enkele gevallen worden uitzonderingssituaties niet gelogd of niet (juist) afgehandeld. Dit kan leiden tot onopgemerkte fouten bij wijzigingen.



Topologie 1 / 2

Dit document bevat het ontwerp van de SPOT-database.

1.2 Databaseoverzicht



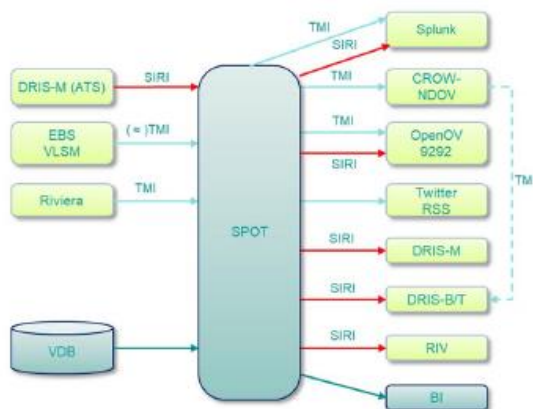
Met opmerkingen [RV1]: Worden de SIRI berichten, die vertaald zijn vanuit de Koppelvlakken, afkomstig uit EBS, VLSM en Riviera niet vastgelegd in MongoDB?

Met opmerkingen [PG(2R1)]: Aangepast

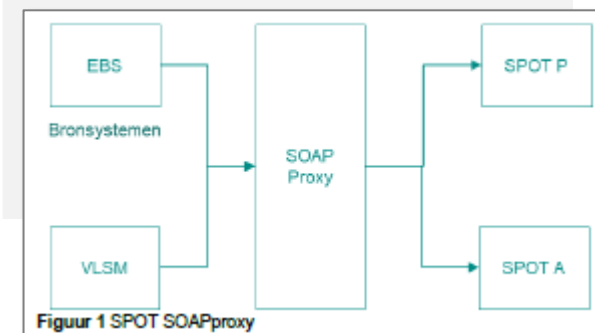
Met opmaak: Engels (Verenigd Koninkrijk)

Met opmaak: Engels (Verenigd Koninkrijk)

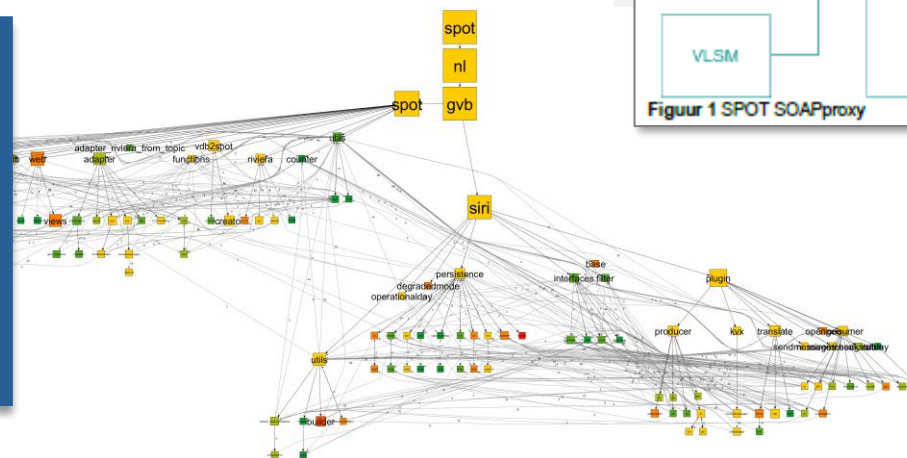
Figuur 1 - Context van de SPOT-database



Topologie is doorgaans te achterhalen uit documentatie, aangevuld met structuur code base

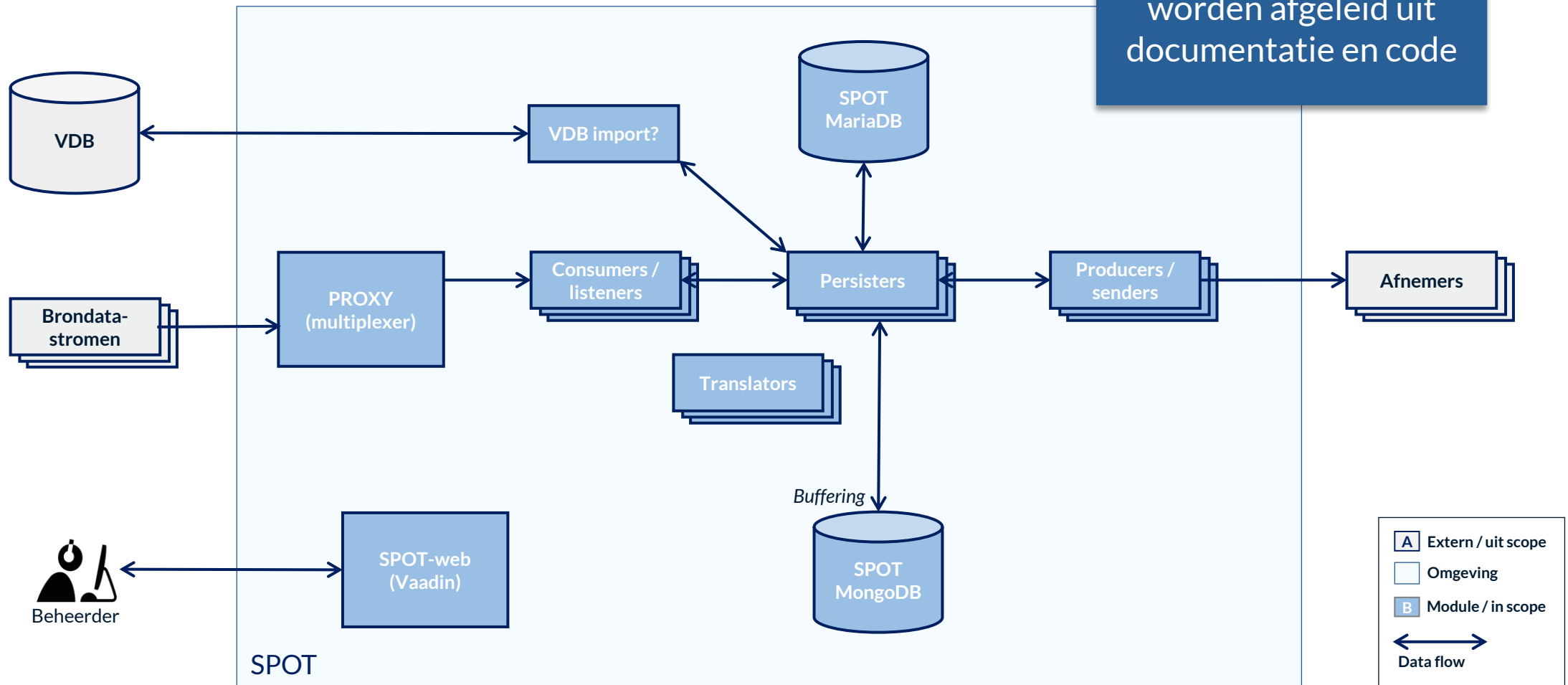


Figuur 1 SPOT SOAPproxy



Topologie 2 / 2

Exacte samenhang en
bewerkingsvolgorde
kan niet eenvoudig
worden afgeleid uit
documentatie en code



Risico's



Risico's

- Ontwikkelaars zijn terughoudend om bestaande code veelzijdiger te maken en zullen bij uitbreidingen bestaande functionaliteit knippen, plakken en enigszins aanpassen, waarbij volume en technische schuld toeneemt.
- Onbegrepen gedrag bij doorontwikkeling door introductie van fouten die niet tijdig worden opgemerkt.
- Onvoldoende productiviteit bij grotere uitbreidingen.
- Het systeem is mogelijk onvoldoende beschermd tegen ongeautoriseerde toegang vanuit het GVB-netwerk, afhankelijk van de standaarden die GVB hiervoor hanteert.

Aanbevelingen



Aanbevelingen algemeen

- Verbeter de analyseerbaarheid door verschillende datastromen of ketens conceptueel uit te werken en te illustreren in documentatie,
 - a.d.h.v. kenmerkende routes door het systeem waarbij alle componenten geraakt worden.
- Verbeter de analyseerbaarheid door de werking van het systeem in high-level code te vangen die delegeert naar low-level afhandeling.
 - Dit kan bijvoorbeeld bij toevoeging van een nieuwe datastroom. Deze code kan vervolgens dienen als **referentie-implementatie** voor vervolgonwikkelingen of refactoring van bestaande functionaliteit bij substantiële wijzigingen.
- Verbeter de wijzigbaarheid met risk-based, ketengerichte regressie(integratie)tests met een korte feedback cyclus. Dit vergroot de wijzigbaarheid en dient tevens als low-level documentatie.
 - Bepaal wenselijkheid a.d.h.v. verwachte hoeveelheid wijzigingen en uitbreiding.
 - Doe een proof-of-concept voor één case om de opzet te valideren.
 - Voeg tests toe bij geconstateerde fouten of geplande wijzigingen. Dit verkleint ook de risico's bij eerder genoemde eventuele refactoring.
- Beperk ontwerpdocumentatie zoveel mogelijk tot de essentie:
 - (Behoud) Belangrijkste ontwerpbeslissingen en rationale.
 - High level concepten en belangrijkste runtime componenten.
 - (Behoud) Beschrijving specifieke aannames bij mapping en belangrijk vergroot herleidbaarheid door in code zelfde terminologie te gebruiken.
 - Verwijs bij details, zoals versienummers (POM), database velden etc. door naar (up-to-date) bron in code base.
 - Vermijd duplicatie in documentatie.



Aanbevelingen technisch

- Maak foutafhandeling waterdicht.
- Adresseer top 10 duplicatie.
- Uniformeer logging, evaluateer nut en noodzaak SpotLog class.
- Voer stricte scheiding test- en productiecode door volgens Maven conventie.
- Uniformeer database interactie, maak alles geparametriseerd, vermijd string concatenatie.
- Verplaats (geduplicateerde) Strings, zoals codes ("PT") naar constanten of enumeraties.
- Verplaats potentieel instelbare gegevens zoals URLs naar configuratie.



Questions, remarks?



J e r o e n M e e t s m a
j . m e e t s m a @ b o n c o d e . n l