Doc. nr.	Datum	Titel doc.	Toelichting
1	21-1-2016	RAM - Technische Bevindingen SIG	
2	1-2-2016	RAM - Einrapportage SIG	

Software Improvement Group

Belastingdienst – RAM

Validatiesessie technische componenten

Persoonsgegevens

21 januari 2016

Agenda

1 Onderzoeksvragen en aanpak
2 Technische componenten RAM
3 Volumebepaling
4 Onderhoudbaarheid
5 Ontwikkelproces
6 Samenvatting bevindingen

Het doel van deze sessie is het valideren van de bevindingen die SIG tijdens de analyse van de technische componenten van RAM heeft gedaan.

Onderzoeksvragen

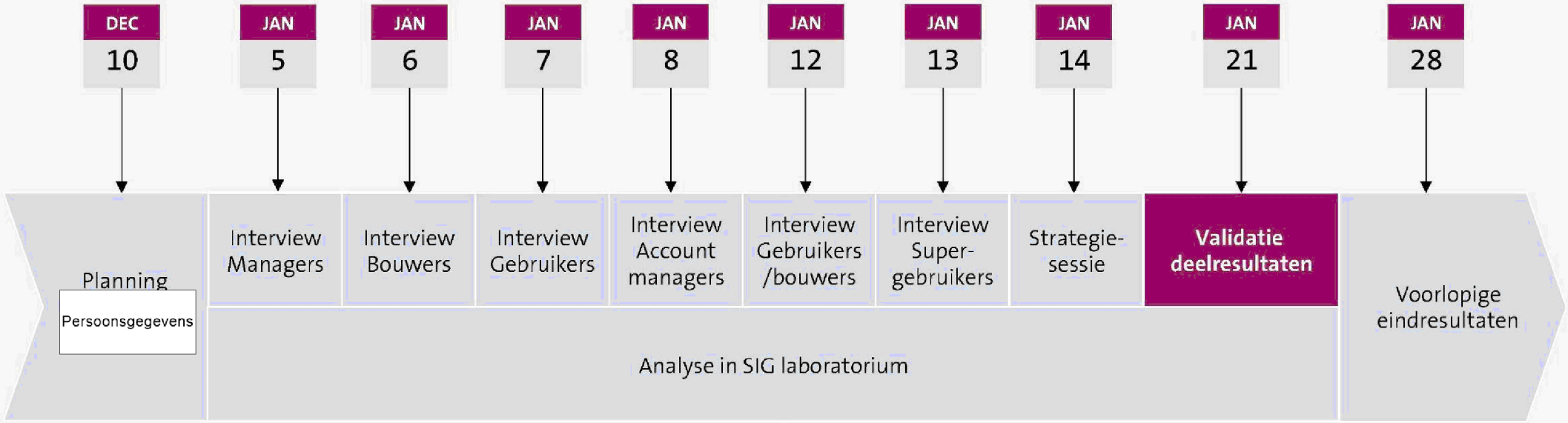
Voer een inventarisatie uit van RAM en geef hierbij antwoord op de volgende vragen:

1. Welke functionaliteit biedt RAM (bestandsopbouw en RAM tooling incl. flexviewer) zijn gebruikers en wie zijn die gebruikers?

2. Uit welke technische componenten bestaat RAM en wat is de onderhoudbaarheid van die componenten (waar mogelijk te bepalen op basis van een broncodeanalyse)?

3. Welke risico's zijn verbonden aan de huidige situatie, zowel bezien vanuit het perspectief van continuering als migratie?

Planning onderzoek



Agenda

1 Onderzoeksvragen en aanpak
2 Technische componenten RAM
3 Volumebepaling
4 Onderhoudbaarheid
5 Ontwikkelproces
6 Samenvatting bevindingen

Technische componenten RAM

Op codeniveau bestaat RAM uit 4 componenten en 1 database

Componenten RAM:

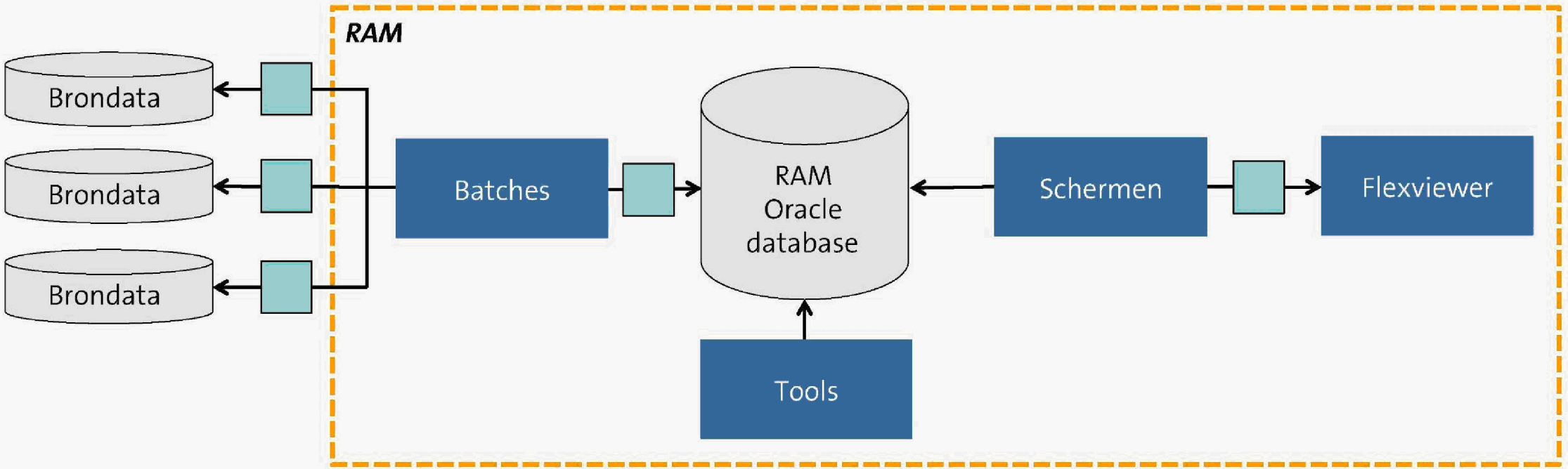
- **Batches:** lezen data-exports uit bronsystemen, bewerken deze data, en schrijven het resultaat vervolgens weg naar de RAM database
- **Schermen:** worden door supergebruikers gebruikt voor het samenstellen en uitvoeren van queries op de RAM database
- **Flexviewer:** wordt gebruikt voor interpretatie van door geautoriseerde gebruikers voorbereidde data
- **Tools:** ondersteunende functionaliteit voor beheer van RAM

Databases:

- **De RAM Oracle database:** slaat de samengevoegde brondata op in een formaat dat aansluit bij het soort bevragingen dat door de gebruikers van RAM wordt gedaan

Technische componenten RAM

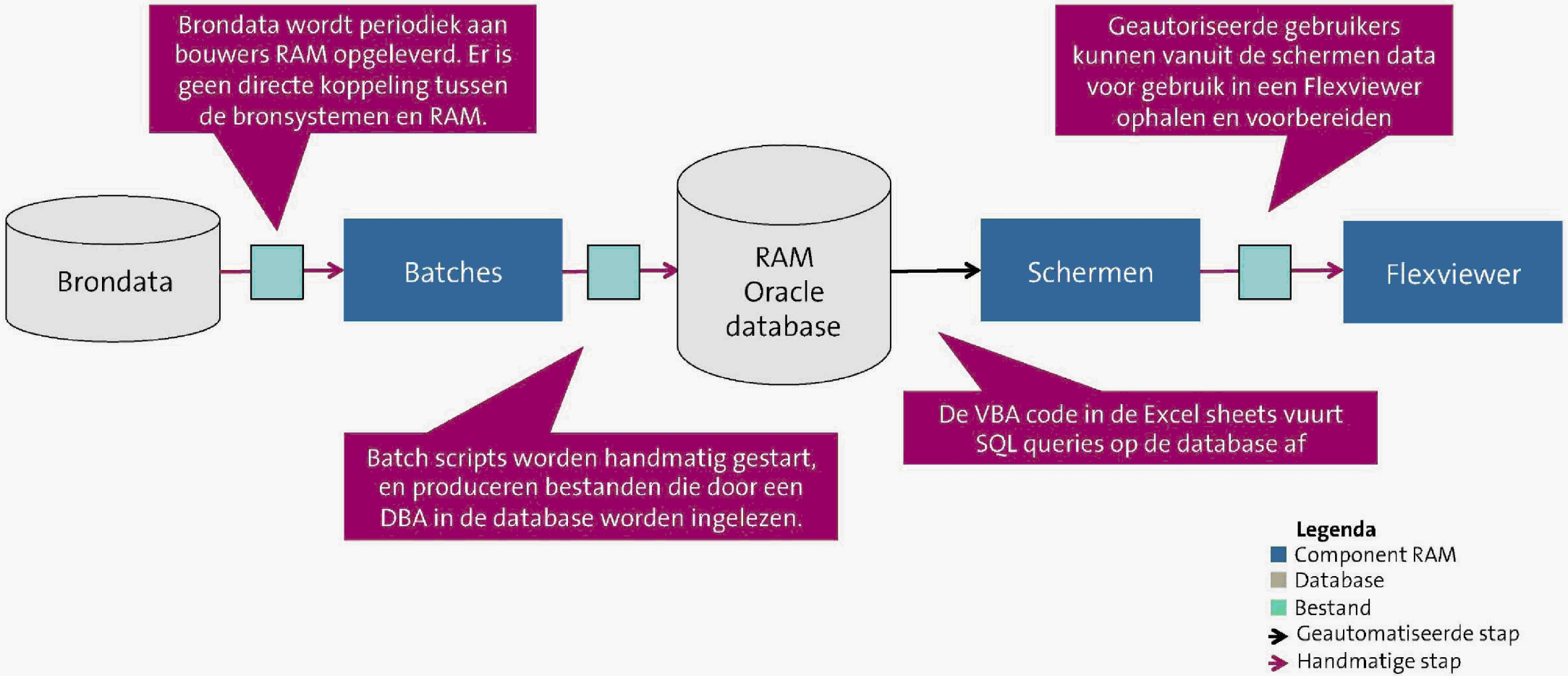
Componenten en interfaces



Legenda
- Component RAM
- Database
- Bestand
- Interface

Technische componenten RAM

Interfaces tussen componenten zijn grotendeels handmatig



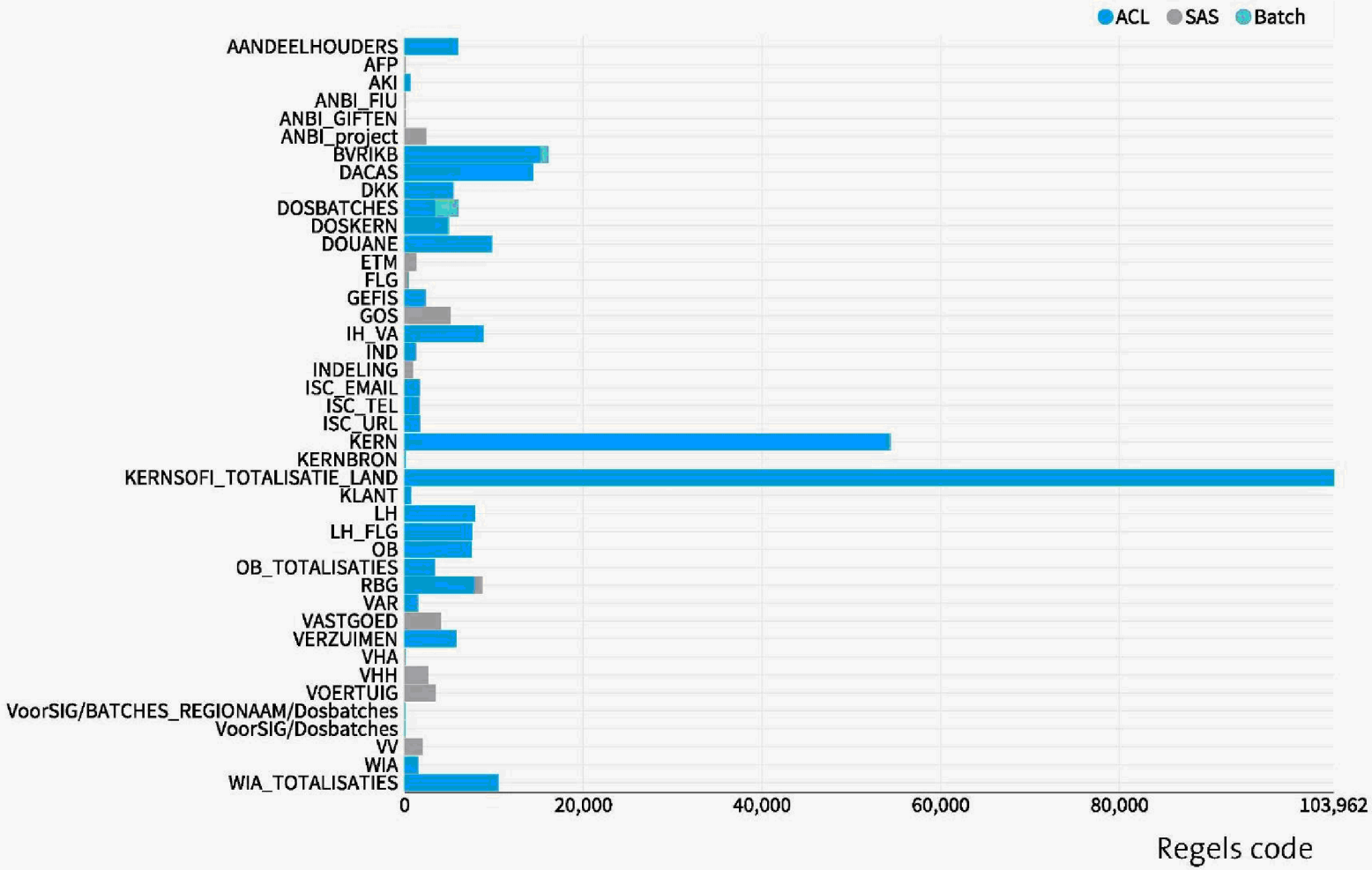
Agenda

1 Onderzoeksvragen en aanpak
2 Technische componenten RAM
3 Volumebepaling
4 Onderhoudbaarheid
5 Ontwikkelproces
6 Samenvatting bevindingen

Volumebepaling

De batches bestaan uit 315.000 regels code, waarvan 90% in ACL is geïmplementeerd

De scripts zijn langs functionele grenzen gegroepeerd



Volumebepaling

Bestandsformaat bronbestanden is gebaseerd op velden met vaste lengte

1) De bronbestanden bevatten de data als platte tekst. Velden worden niet gescheiden d.m.v. een delimiter (zoals bij CSV), maar hebben een vaste lengte.

2) De ACL scripts bevatten de lokaties van de verschillende velden binnen het bronbestand.

^LAYOUT	KERN_SUMM_914_2014		
RECORD_DELETED	ASCII	1	1
JAAR	ASCII	254	4
SOFIGELDIG	ASCII	258	1
REGIOCODE	ASCII	259	3
REGIONAAM	ASCII	262	18
KANTOOR	ASCII	280	10
KANTID	ASCII	290	3
KANTNAAM	ASCII	293	10
TEAMNW	ASCII	303	2
TEAM_INDEL	ASCII	305	22
CODE	ASCII	327	29

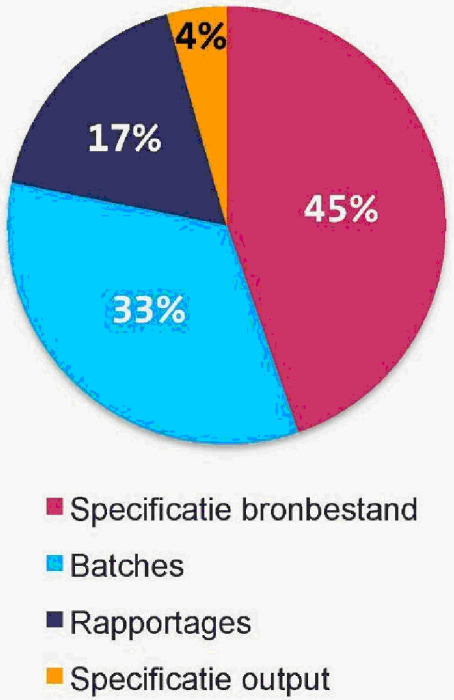
Voorbeeld: het veld "JAAR" begint op positie 254 en heeft een lengte van 1 karakter.

Voorbeeld uit _KERNSOFI_SUMM_BVR.ACL

Volumebepaling

Inrichting koppeling met bronbestanden zorgt voor grote hoeveelheden code in RAM

Verdeling volume ACL scripts



45% van de ACL code bestaat uit het specificeren van de lokaties van velden binnen de bronbestanden

- Daarnaast bestaat nog eens 4% van de code uit specificatie van de *output*bestanden
- Met een ander soort koppelingen (bijvoorbeeld CSV-, XML-, JSON-bestanden; rechtstreekse koppeling op de database; koppeling via webservice) zou de onderhoudslast van de ACL code dus met 50% kunnen worden verlaagd

Volumebepaling

Correcties op broncode zijn in code geïmplementeerd

```
"2009-01-01" IF DOSNR = "002941314"
"2003-06-28" IF DOSNR = "009589016"
"1994-01-01" IF DOSNR = "037220329"
"1995-01-01" IF DOSNR = "042569199"
"1995-01-01" IF DOSNR = "050468662"
"2006-11-25" IF DOSNR = "075920396"
"2006-09-18" IF DOSNR = "107073432"
"2002-01-01" IF DOSNR = "113857226"
"2001-04-13" IF DOSNR = "118603292"
"2006-09-30" IF DOSNR = "154193331"
"2007-11-12" IF DOSNR = "161638831"
"1994-12-21" IF DOSNR = "163678492"
"2008-04-11" IF DOSNR = "243069650"
"1992-01-01" IF DOSNR = "800221175"
"2002-01-01" IF DOSNR = "801222102"
"1999-01-01" IF DOSNR = "803077932"
"1999-01-01" IF DOSNR = "803078158"
"1999-01-01" IF DOSNR = "803078213"
"1995-01-01" IF DOSNR = "803524225"
"2003-07-26" IF DOSNR = "804267339"
"1997-01-01" IF DOSNR = "804413757"
"2000-01-01" IF DOSNR = "805367147"
```

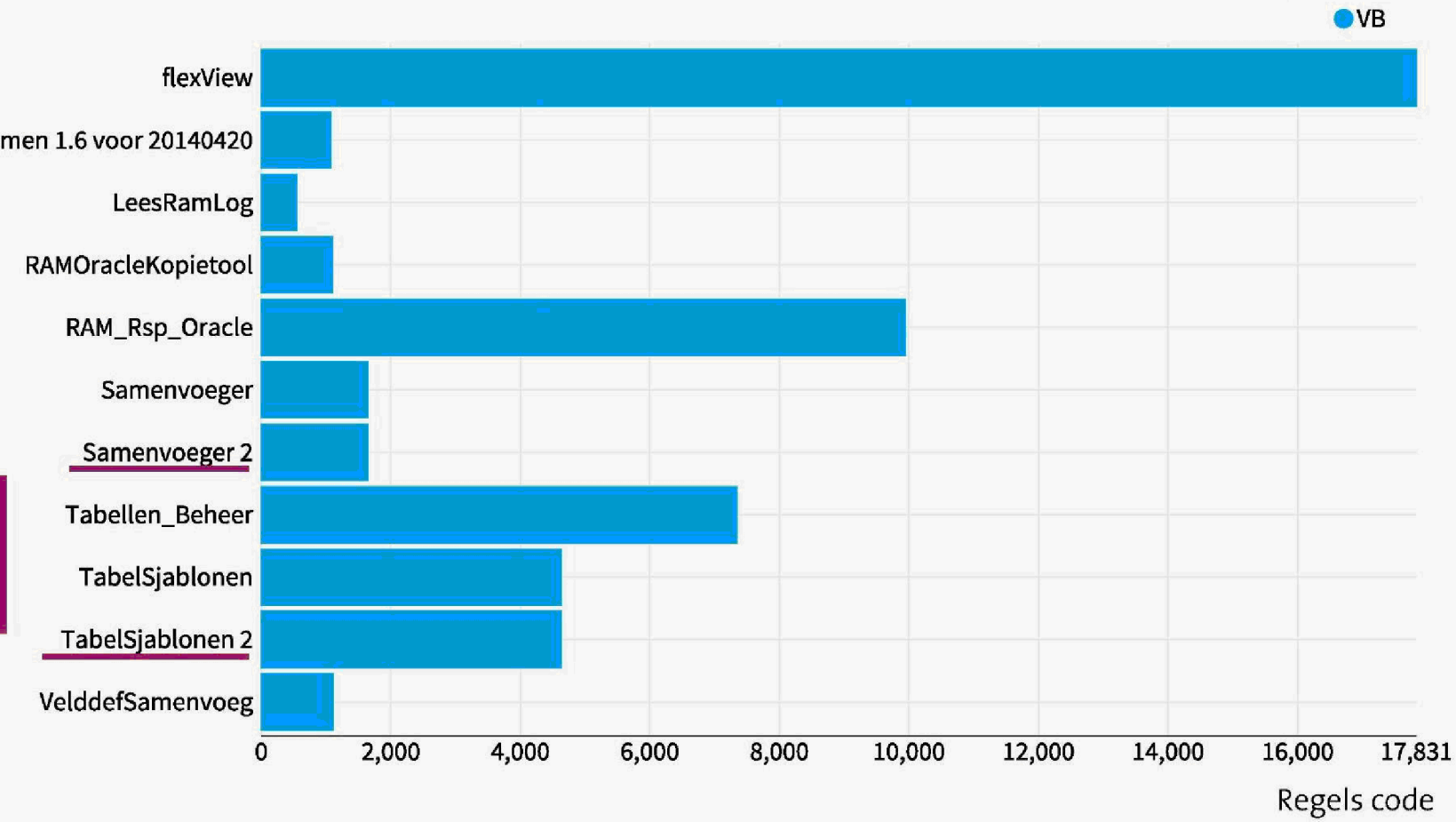
Voorbeeld uit BATCHES_LAND/BVRIKB/_BVRIKB.ACL

Naast het importeren en converteren van brondata voeren de ACL scripts ook correcties op de data uit

- Deze correcties zijn niet gericht op patronen (bijvoorbeeld: conversie van oude rekeningnummers naar IBAN), maar gericht op specifieke fouten in de brondata
- De import is een éénrichtingsproces: deze correcties worden na detectie niet in de brondata gemaakt, maar worden door het script bij elke nieuwe import opnieuw uitgevoerd

Volumebepaling

De RAM Excel sheets bevatten in totaal 51.000 regels VBA code

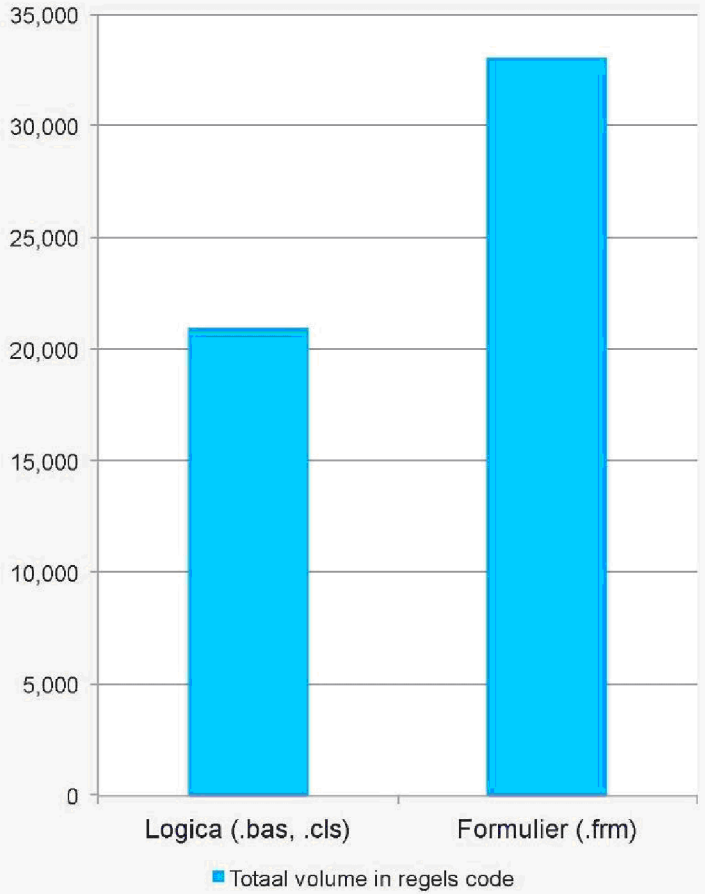


2 bestanden komen dubbel voor. Het verwijderen van de loze kopieën scheelt 6.000 regels code.

Volumebepaling

VBA scripts bevatten naast presentatie ook logica voor bewerkingen van data

Codevolume VBA scripts, per type script



De VBA scripts bestaan voor een significant deel (21.000 regels code, 40%) uit logica

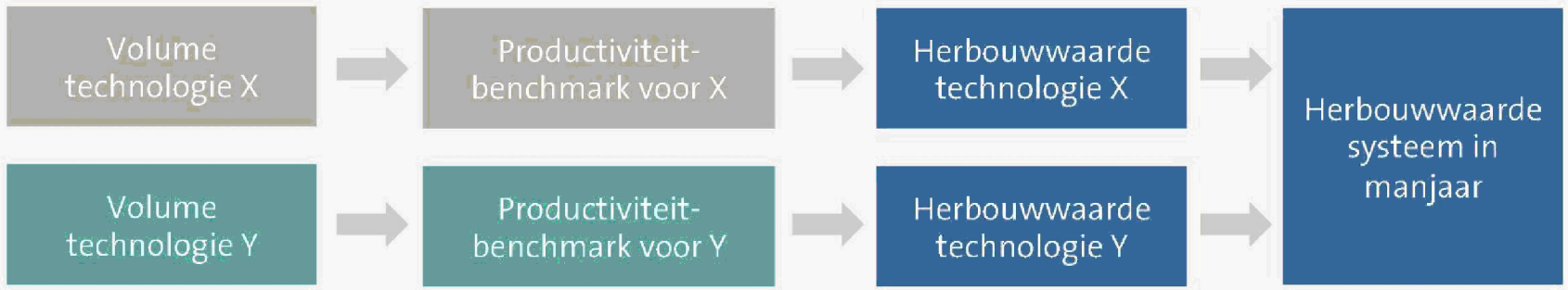
- Deze logica wordt gebruikt voor onder andere:
  - Dataconversie van oude naar nieuwe formaten
  - Samenvoegen/totaliseren van data
  - Het exporteren van data naar andere (bestands)formaten
- In technische termen: de schermen en de Flexviewer zijn niet alleen de “front-end” van RAM

Volumebepaling

SIG drukt volume uit in herbouwwaarde

De *herbouwwaarde* van een systeem beschrijft hoe lang het bij marktgemiddelde productiviteit zou duren om het systeem te herbouwen

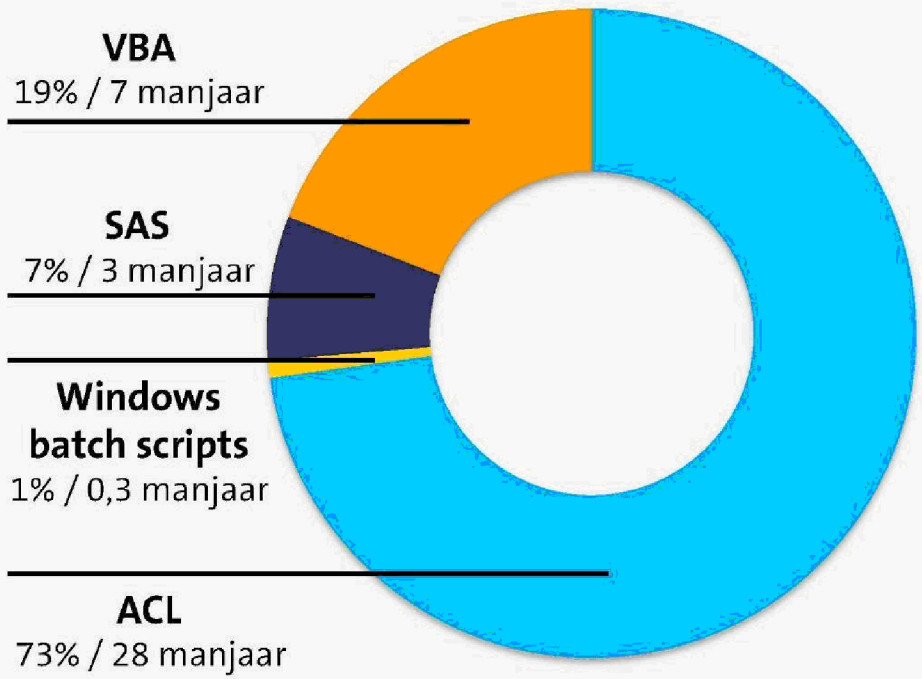
- De herbouwwaarde wordt berekend door het volume in regels code te vermenigvuldigen met de marktgemiddelde productiviteit in de gebruikte technologie
- Door herbouwwaarde te gebruiken kan het volume van systemen in verschillende technologieën met elkaar vergeleken worden
- Het aantal manjaar is een bepaling van het codevolume, en geeft niet weer hoeveel tijd er daadwerkelijk is geïnvesteerd



Volumebepaling

De herbouwwaarde van de RAM-code is 38 manjaar

Herbouwwaarde RAM per technologie



80% van de RAM-code (31 manjaar aan code) bestaat uit de batches voor het importeren van data uit bronbestanden

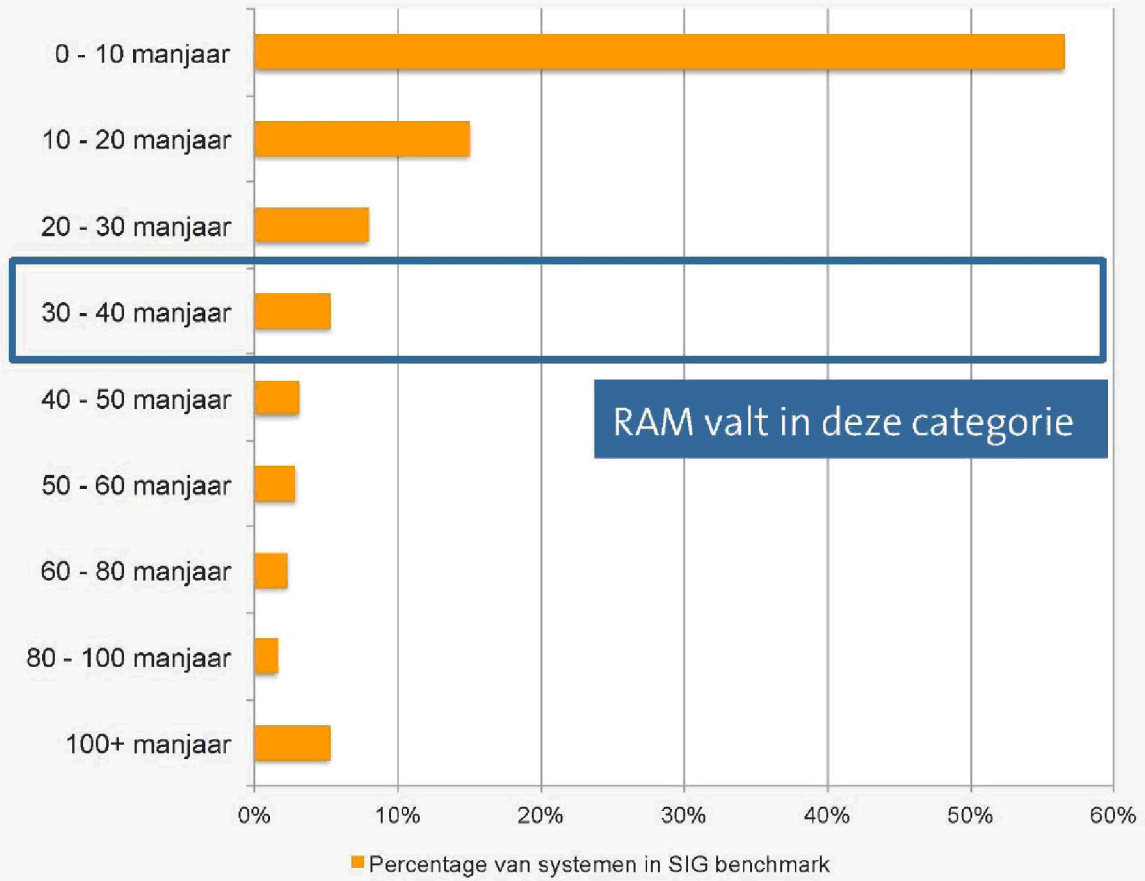
- Het aandeel van de VBA code in de schermen en de Flexviewer op het totale volume is betrekkelijk klein

De herbouwwaarde betreft alleen de code van RAM, niet de documentatie

Volumebepaling

Met 38 manjaar is RAM vergeleken met de SIG benchmark een systeem van gemiddelde grootte

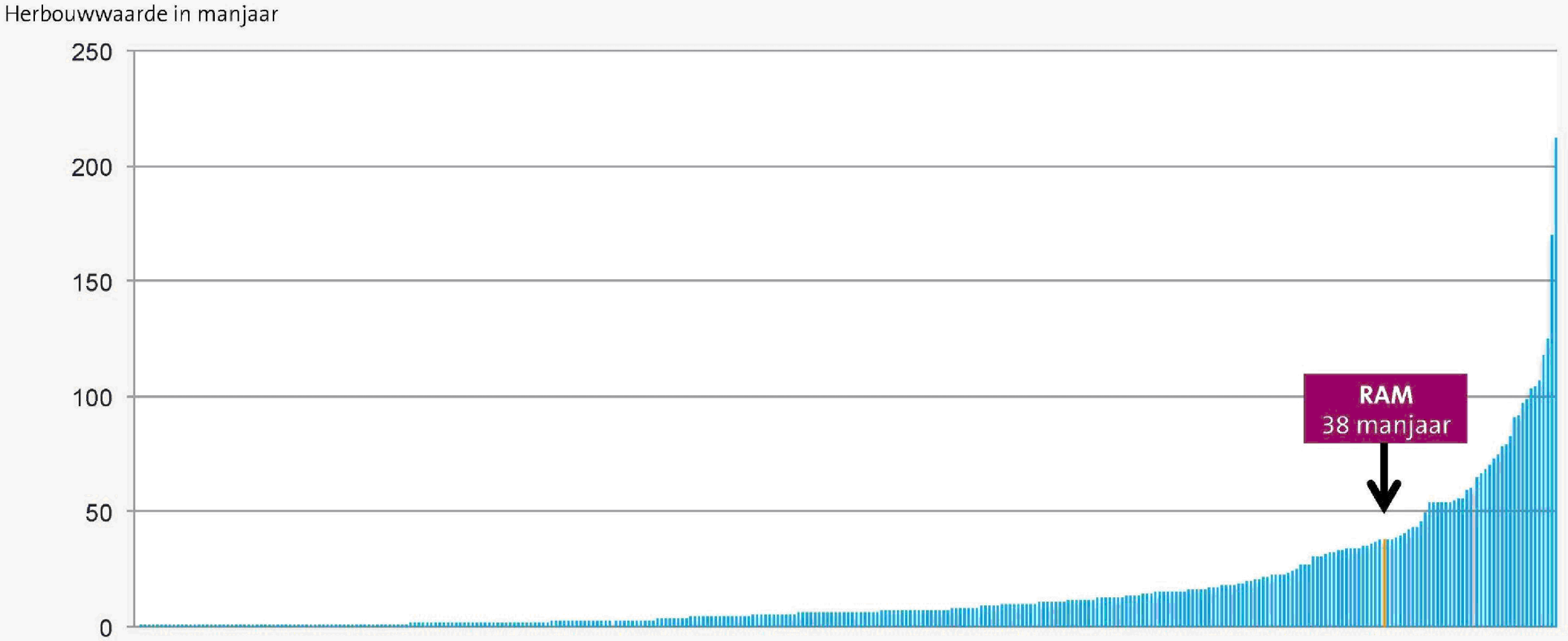
Vergelijking herbouwwaarde met de 1.889 systemen in de SIG benchmark



Volumebepaling

RAM hoort bij de grootste 20% door SIG geanalyseerde Belastingdienst-systemen

Vergelijking herbouwwaarde met de 343 eerder door SIG geanalyseerde Belastingdienst-systemen



Agenda

1 Onderzoeksvragen en aanpak
2 Technische componenten RAM
3 Volumebepaling
4 Onderhoudbaarheid
5 Ontwikkelproces
6 Samenvatting bevindingen

Onderhoudbaarheid

De ISO 25010 standaard voor softwarekwaliteit

ISO 25010 definieert onderhoudbaarheid als 1 van de 8 karakteristieken van softwarekwaliteit

- Onderhoudbaarheid wordt gedefinieerd als *"de mate waarin een product of systeem effectief en efficiënt gewijzigd kan worden"*
- Onderhoudbaarheid heeft een sterke invloed op de snelheid waarmee zowel de functionaliteit en de overige kwaliteitskarakteristieken kunnen worden aangepast
- De onderhoudsfase beslaat 80% van de kosten van een systeem

Onderhoudbaarheid

Beperkingen van ISO 25010

ISO 25010 geeft alleen definities van onderhoudbaarheid en de subkarakteristieken daarvan

- Om systemen te kunnen beoordelen is naast de definitie het volgende nodig:

1. Metrieken

   Beschrijf hoe de subkarakteristieken gemeten kunnen worden.

2. Eenheden

   Beschrijf eenheden om meetresultaten in uit te drukken.

3. Normen

   Zorg ervoor dat meetresultaten met elkaar vergeleken kunnen worden.

Onderhoudbaarheid

Doelen van een model om onderhoudbaarheid van software te kunnen meten

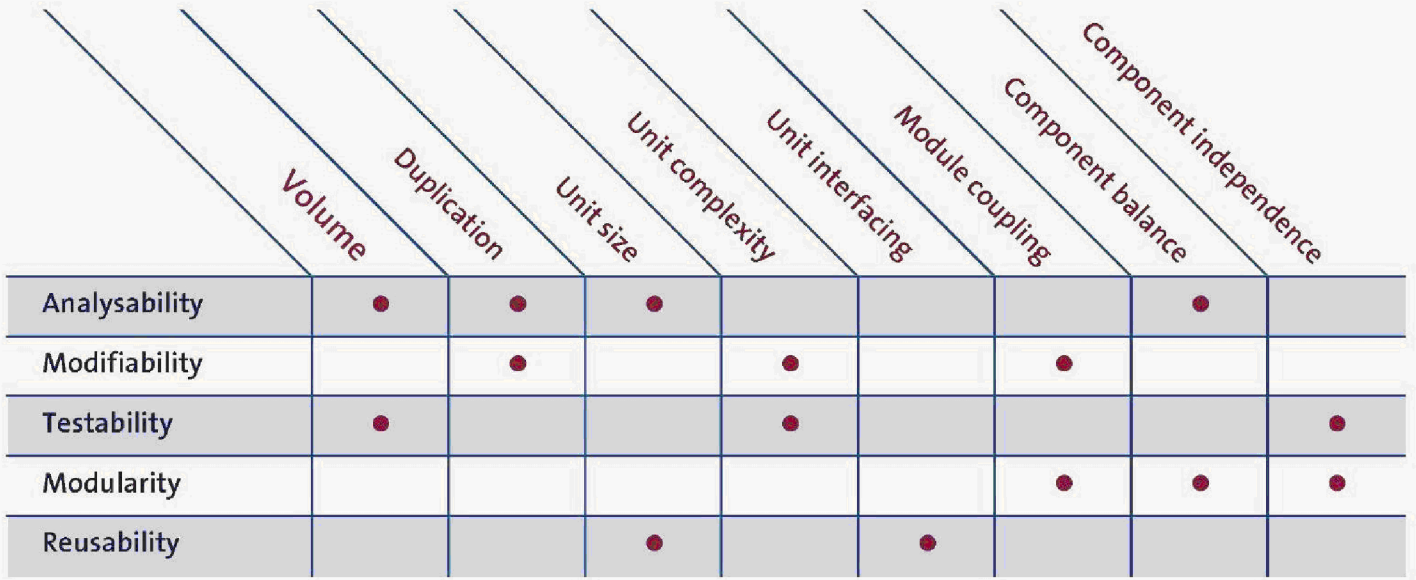
Makkelijk te begrijpen	Iedereen met een achtergrond in software engineering moet de metingen en de meetresultaten kunnen begrijpen.
Objectief	De meetresultaten moeten niet afhankelijk zijn van de persoon die de metingen uitvoert.
Inzicht in oorzaken	De metingen moeten inzicht geven in welke aspecten het systeem beter of minder onderhoudbaar maken.
Maakt systemen vergelijkbaar	De meetresultaten moeten tussen systemen vergeleken kunnen worden, zodat duidelijk wordt welke aspecten van een systeem relatief beter/minder zijn.
Technologie-onafhankelijk	De metingen moeten toepasbaar zijn onafhankelijk van de technologie(ën) waarin het systeem is geïmplementeerd.

Onderhoudbaarheid

Het SIG/TÜViT kwaliteitsmodel voor onderhoudbaarheid

Het SIG kwaliteitsmodel gebruikt broncodemetingen voor een evaluatie van de verschillende subkarakteristieken van onderhoudbaarheid in ISO 25010

- Deze metingen worden vervolgens vergeleken met de andere systemen in de SIG benchmark, wat resulteert in een score tussen de 1 en 5 sterren



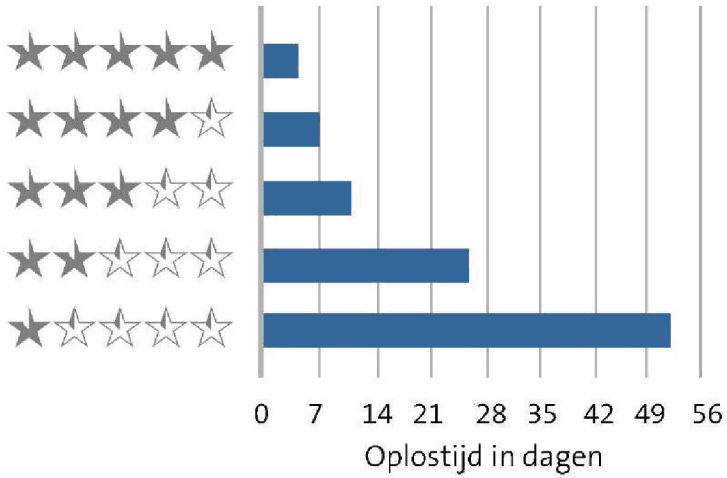
Onderhoudbaarheid

Er bestaat een correlatie tussen de scores in het SIG model en onderhoud in de praktijk

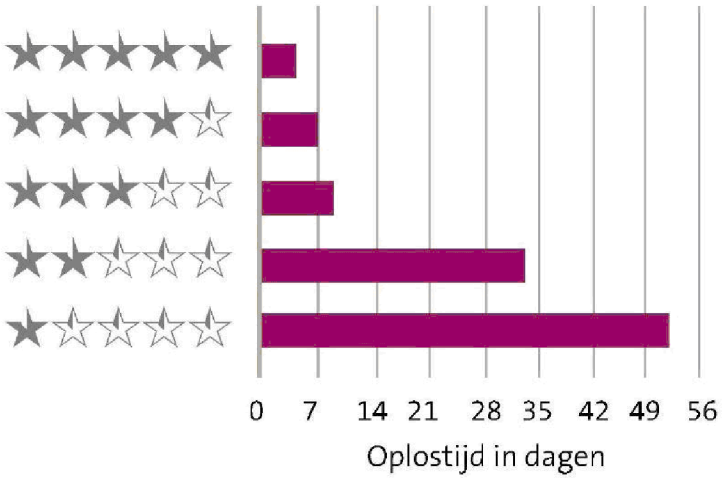
In goed onderhoudbare systemen kunnen wijzigingen sneller worden doorgevoerd

- Dit geldt voor zowel het oplossen van bugs als het implementeren van nieuwe functionaliteit

Tijd nodig voor implementeren nieuwe functionaliteit



Tijd nodig voor oplostijd bugs



Source: "Faster issue resolution with higher technical quality of software", Software Quality Journal, 2011

Onderhoudbaarheid

RAM is beneden marktgemiddeld onderhoudbaar (★★☆☆☆)

	Batches	Schermen	Flexviewer	Tools
Volume		★★★☆☆		
Duplication	★☆☆☆☆	★★★☆☆	★★☆☆☆	★☆☆☆☆
Unit size	★☆☆☆☆	★★☆☆☆	★★☆☆☆	★★☆☆☆
Unit complexity	★★☆☆☆	★★☆☆☆	★★★☆☆	★★★☆☆
Unit interfacing	★★★★★	★★★☆☆	★★★☆☆	★★★★☆
Module coupling	★★★★☆	★★★★☆	★★★★★	★★★★★
Component balance		★★☆☆☆		
Component independence		★★★★★		
Maintainability		★★☆☆☆		

Onderhoudbaarheid

RAM is beneden marktgemiddeld onderhoudbaar (★★☆☆☆)

Over het algemeen is de onderhoudbaarheid van de VBA-code (★★★☆☆) hoger dan de ACL (★☆☆☆☆)

- Dit komt gedeeltelijk door de karakteristieken van de technologieën
- De manier waarop de bronbestanden moeten worden ingelezen leidt tot grote scripts
- Vastleggen domeinkennis in de batch scripts leidt tot complexiteit in code
- De lage onderhoudbaarheid van de ACL scripts is problematisch aangezien deze 80% van de code beslaat

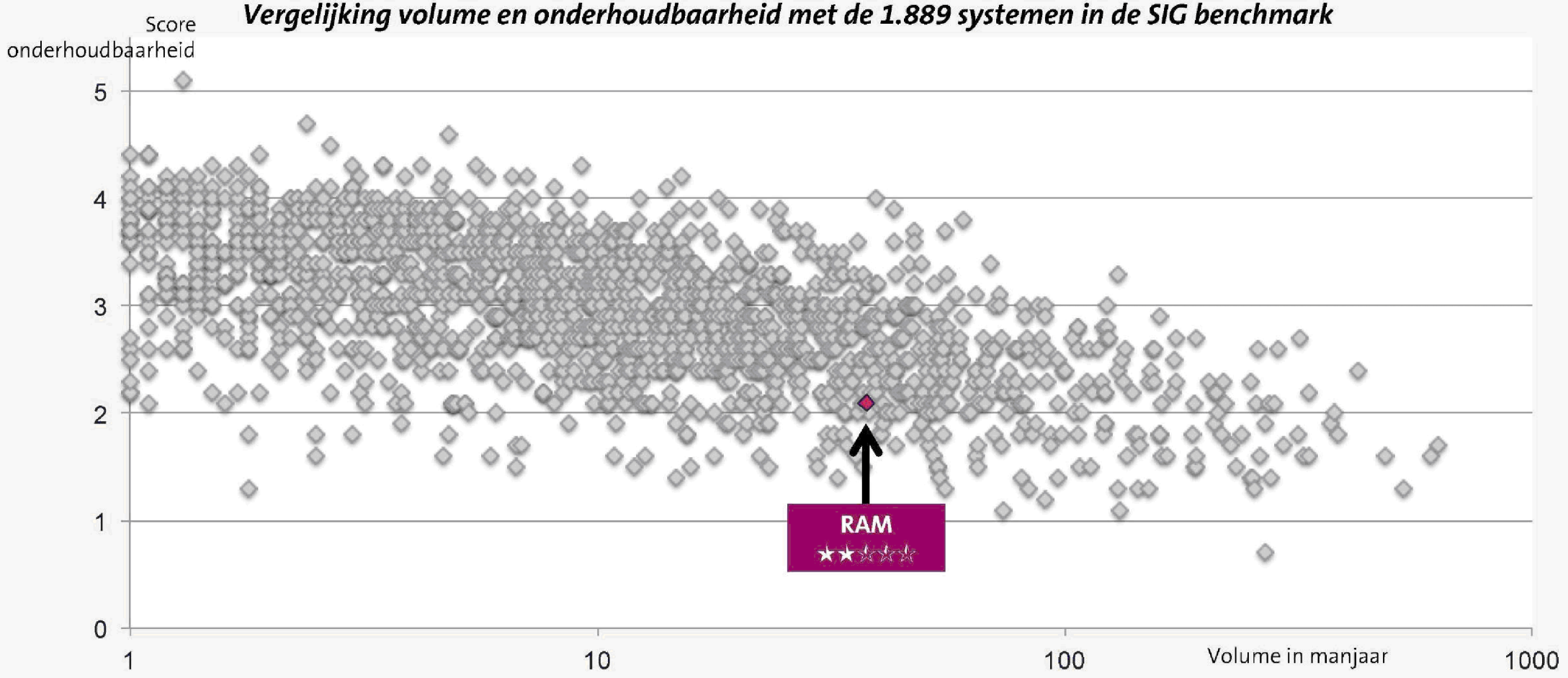
RAM kent weinig verwevenheid

- De database fungeert als interface tussen de componenten: zolang de database-structuur gelijk blijft kunnen de verschillende componenten onafhankelijk van elkaar onderhouden worden

Onderhoudbaarheid

De onderhoudbaarheid van RAM is voor een systeem van 38 manjaar relatief laag

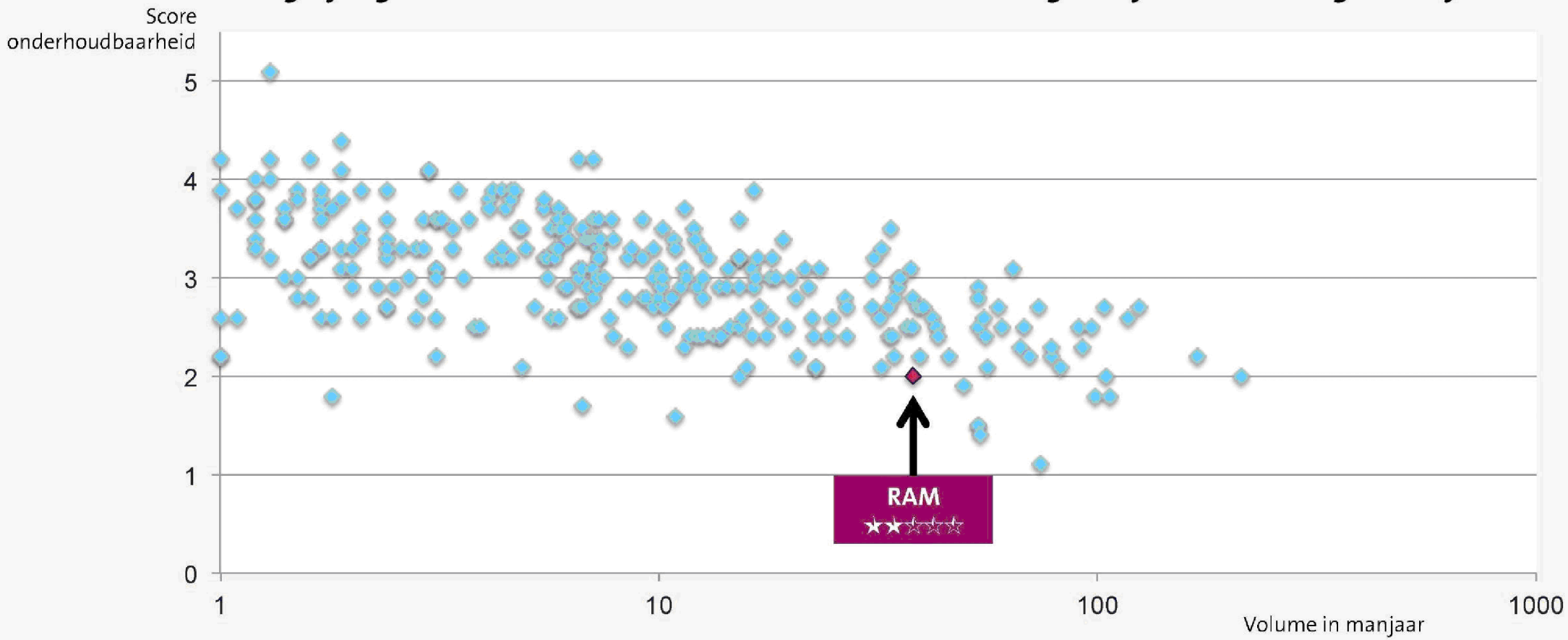
Vergelijking volume en onderhoudbaarheid met de 1.889 systemen in de SIG benchmark



Onderhoudbaarheid

RAM hoort bij de 10% minst onderhoudbare Belastingdienst-systemen

Vergelijking volume en onderhoudbaarheid met de 343 door SIG geanalyseerde Belastingdienst-systemen



Onderhoudbaarheid

Duplication: uitleg van de metriek

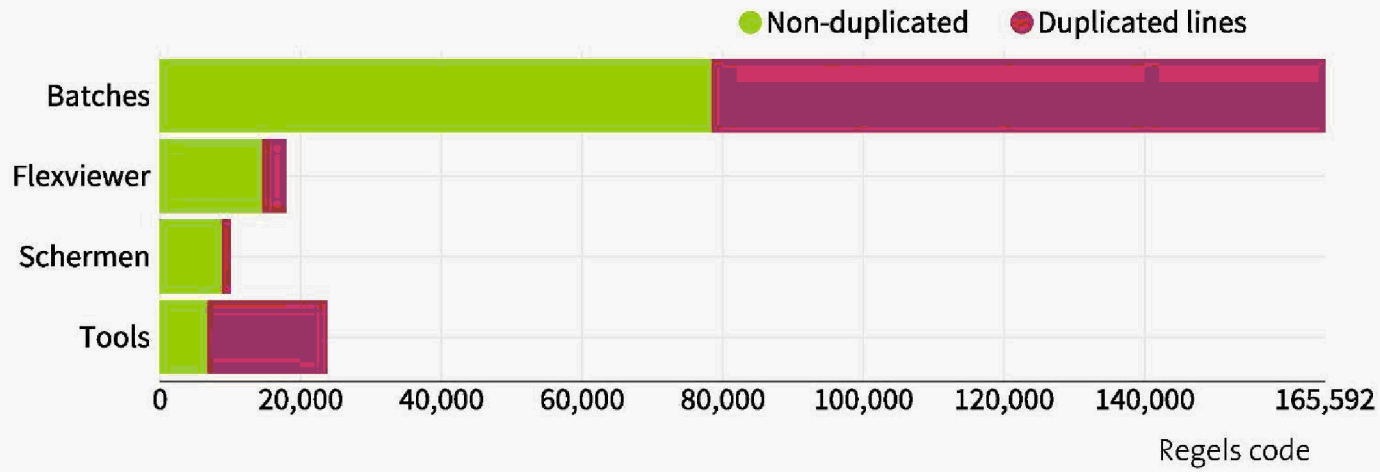
Duplicaten in code verminderen de onderhoudbaarheid

- Veel duplicatie maakt het fixen van bugs moeilijker
- Veel duplicatie maakt het testen moeilijker

Gemeten als het percentage redundante code:

Onderhoudbaarheid

Grote verschillen in de hoeveelheid duplicatie tussen componenten



De batch scripts zijn grotendeels gedupliceerd (53%)

- Dit geldt voor beide technologieën die voor de batch scripts worden gebruikt (ACL en SAS)
- De duplicatie tussen de VBA code in de Excel is in vergelijking relatief beperkt

Onderhoudbaarheid

Grote queries en blokken code worden tussen SAS scripts gekopieerd

```
          TABEL2.STUK_WIJZIGING_WW,
          TABEL2.STUK_WIJZIGING_DATUM_WW,
          TABEL2.VERLIJDENSDATUM_WW,
          TABEL2.SOORT_STUK_WW
      FROM TW_WRK.QUERY_FOR_VASTGOED_OBJ_OBK TABEL1, TI_KAD.KAD_OBJECTADRES TABEL2
      WHERE (TABEL1.OBJE_KAD_GEM_K = TABEL2.KADASTRALE_GEMEENTECODE AND TABEL1.OBJE_SEKTIE = TABEL2.SECTIE AND
TABEL1.PERCEELNR =
           TABEL2.PERCEELNUMMER AND TABEL1.OBJE_OBJ_INDEX_LR = TABEL2.OBJECTINDEXLETTER AND TABEL1.OBJ_INDEX_NR =
TABEL2.OBJECTINDEXNUMMER);
QUIT;
```

Voorbeeld uit VASTGOED.sas

```
          TABEL2.STUK_WIJZIGING_WW,
          TABEL2.STUK_WIJZIGING_DATUM_WW,
          TABEL2.VERLIJDENSDATUM_WW,
          TABEL2.SOORT_STUK_WW
      FROM TW_WRK.QUERY_FOR_VHH_OBJ_OBK TABEL1, TI_KAD.KAD_OBJECTADRES TABEL2
      WHERE (TABEL1.OBJE_KAD_GEM_K = TABEL2.KADASTRALE_GEMEENTECODE AND TABEL1.OBJE_SEKTIE = TABEL2.SECTIE AND
TABEL1.PERCEELNR =
           TABEL2.PERCEELNUMMER AND TABEL1.OBJE_OBJ_INDEX_LR = TABEL2.OBJECTINDEXLETTER AND TABEL1.OBJ_INDEX_NR =
TABEL2.OBJECTINDEXNUMMER);
QUIT;
```

Voorbeeld uit VHH.sas

Dit blok code van 700 regels is in beide scripts op 1 regel na gelijk

- Gevolg: toekomstige verbeteringen nu steeds in beide bestanden moeten worden doorgevoerd
- Het is beter om gedeelde functionaliteit naar één gedeeld bestand te verplaatsen, en deze functionaliteit vervolgens vanuit beide scripts aan te roepen

Onderhoudbaarheid

Ook in de ACL scripts worden grote blokken code gedupliceerd

```
Dagen = CTOD(DATE();"DD-MM-YYYY") - CTOD(ALLTRIM(_Startdatum);"DD-MM-YYYY")

IF Dagen > 0 uren = VALUE(SUBSTR(Eindtijd;1;2);0)+Dagen*24 - VALUE(SUBSTR(_Starttijd;1;2);0)
IF Dagen = 0 uren = VALUE(SUBSTR(Eindtijd;1;2);0)          - VALUE(SUBSTR(_Starttijd;1;2);0)

IF VALUE(SUBSTR(Eindtijd;4;2);0) <  VALUE(SUBSTR(_Starttijd;4;2);0) uren = uren-1
IF VALUE(SUBSTR(Eindtijd;4;2);0) <  VALUE(SUBSTR(_Starttijd;4;2);0) minuten = VALUE(SUBSTR(Eindtijd;4;2);0)+60 -
VALUE(SUBSTR(_Starttijd;4;2);0)
IF VALUE(SUBSTR(Eindtijd;4;2);0) >= VALUE(SUBSTR(_Starttijd;4;2);0) minuten = VALUE(SUBSTR(Eindtijd;4;2);0)    -
VALUE(SUBSTR(_Starttijd;4;2);0)

IF VALUE(SUBSTR(Eindtijd;7;2);0) <  VALUE(SUBSTR(_Starttijd;7;2);0) minuten = minuten-1
IF VALUE(SUBSTR(Eindtijd;7;2);0) <  VALUE(SUBSTR(_Starttijd;7;2);0) seconden = VALUE(SUBSTR(Eindtijd;7;2);0)+60 -
VALUE(SUBSTR(_Starttijd;7;2);0)
IF VALUE(SUBSTR(Eindtijd;7;2);0) >= VALUE(SUBSTR(_Starttijd;7;2);0) seconden = VALUE(SUBSTR(Eindtijd;7;2);0)    -
VALUE(SUBSTR(_Starttijd;7;2);0)

Verstrekentijd = ALLTRIM(ALLTRIM(STR(uren;4))+" uur "+STR(minuten;2)+" minuten "+STR(seconden;2)+" seconden")

COMM Dialoogbox met einde batch en verstreken tijd
DIALOG (DIALOG TITLE "    ISC_URLadressen_batch" WIDTH 459 HEIGHT 180 ) (BUTTONSET TITLE "&OK" AT 180 140 DEFAULT 1 )
(TEXT TITLE "De batch (inclusief export MDB-bestanden) regio %_regio_zonder_underscore% is klaar!" AT 20 35) (TEXT TITLE
"Batch gestart op:  %_Startdatum% %_Starttijd%" AT 20 65) (TEXT TITLE "Batch gestopt op: %Einddatum% %Eindtijd%" AT 20
80) (TEXT TITLE "Verstreken tijd :     %Verstrekentijd%" AT 20 95)
```

Dit duplicaat is in totaal 300 regels code

Voorbeeld uit IND.ACL en ISC_URLadressen.ACL

Ook in de ACL scripts worden grote stukken code gekopieerd

- In tegenstelling tot bij SAS is het bij ACL een stuk moeilijker om gedeelde functionaliteit apart te zetten, de duplicatie is hier deels een gevolg van de gebruikte technologie

Onderhoudbaarheid

Unit size: uitleg van de metriek

Veel lange units in een systeem vermindert de onderhoudbaarheid

- Een unit is een functie, methode of procedure
- Korte units kunnen makkelijk worden hergebruikt
- Korte units zijn doorgaans minder complex
- Korte units zijn makkelijk gedocumenteerd door hun naam

Gemeten als risicoprofiel op basis van lengte van units in regels code:

```
Collection<Package> getChangedPackages() {
   if (changedPackages != null) {
      return changedPackages;
   } else {
      return Collections.emptyList();
   }
}
```

7 LOC in unit

Lengte van de unit (Lines of Code)	Risico-inschatting
1-15	Eenvoudig te begrijpen, geen risico
16-30	Gewone lengte, laag risico
31-60	Lange unit, gemiddeld risico
> 60	Erg lange unit, hoog risico

Onderhoudbaarheid

Herhaling leidt tot grote functies en onnodig veel code

```
'Toelichting toevoegen Als Code = 2 dan Toel Index toevoegen (Naam van de toelichting komt nu in de

If Mtx_Toelichtingen(iMx, 1) = 2 Then _
    Worksheets(strToelichtingen).Cells(R, iMx) = Worksheets(sExtr).Cells(lR, 1)

'Toelichting toevoegen Als Code = 3 dan Toelichting tekst toevoegen
If Mtx_Toelichtingen(iMx, 1) = 3 Then _
    Worksheets(strToelichtingen).Cells(R, iMx) = Worksheets(sExtr).Cells(lR, lRefKol)

'Toelichting toevoegen Als Code = 4 dan USERID en Naam toevoegen
If Mtx_Toelichtingen(iMx, 1) = 4 Then _
        Worksheets(strToelichtingen).Cells(R, iMx) = Environ("UserName") & " | " & Application.UserName

'Toelichting toevoegen Als Code = 5 dan Actuele bestandsnaam
If Mtx_Toelichtingen(iMx, 1) = 5 Then _
        Worksheets(strToelichtingen).Cells(R, iMx) = ActiveWorkbook.Name

'Toelichting toevoegen Als Code = 6 dan Datum en tijd van verzamelen
If Mtx_Toelichtingen(iMx, 1) = 6 Then _
        Worksheets(strToelichtingen).Cells(R, iMx) = Format(Now(), "dd-mm-yy, hh:mm:ss")
```

Voorbeeld uit mod_toelichtingen

Hetzelfde blok code steeds herhalen zorgt ervoor dat functies al snel groot worden

- Toekomstige aanpassingen moeten in elke variant opnieuw worden doorgevoerd
- Door dit soort herhaling anders te implementeren, in dit geval bijvoorbeeld in een select/case, is minder code nodig, en wordt de functie daardoor makkelijker aanpasbaar

Onderhoudbaarheid

Grote functies zijn moeilijk te begrijpen doordat functies niet in deelproblemen worden opgesplitst

```
  Case 2  'per waarde/per tabel in 1 bestand
      'Nieuw klantbeeldbestand.
      'Maak een nieuw Excel Klantbeeld bestand
      Set wkbxlsklantbeeld(1) = Doelbestand_XlsNieuwBestandInGeheugen()
      'Het eerste blad wordt RAM_Info
      Set wksRAMInfo(1) = ActiveSheet
      Call MaakWkbKlantbeeld(sVeld, swaarde, wksRAMInfo(1))
      'JGA20100908: Aanvulling Bloknamen
      'Tweede blad aanmaken: Bloknamen
      Set wksBloknamen(1) = BloknamenBlad(wkbxlsklantbeeld(1), "Bloknamen", False)
      'Project en sjabloonblad toevoegen
      wkbxlsklantbeeld(1).SaveAs filename:=sKlantbeeldXlsDoelDir & sKlantbeeldDoelXls(1) & "_" &
ConflictResolution:=xlOtherSessionChanges
      lRecordteller(1) = 0
  Case 3  'Per tabel een bestand
      'Maak een nieuw Excel Klantbeeld bestand
      Set wkbxlsklantbeeld(1) = Doelbestand_XlsNieuwBestandInGeheugen()
      'Het eerste blad wordt RAM_Info
      Set wksRAMInfo(1) = ActiveSheet
      Call MaakWkbKlantbeeld(sVeld, swaarde, wksRAMInfo(1))
      'JGA20100908: Aanvulling Bloknamen
      'Tweede blad aanmaken: Bloknamen
      Set wksBloknamen(1) = BloknamenBlad(wkbxlsklantbeeld(1), "Bloknamen", False)
      'Project en sjabloonblad toevoegen
      sKlantbeeldDoelXls(1) = cKbTekst & "_" & sTabelnaam
      iVolgnr = 1
      While Dir(Doel.sBeeldOpslaanInMap & sKlantbeeldDoelXls(1) & "_" & iVolgnr & "*") <> ""
          iVolgnr = iVolgnr + 1
      Wend
      sKlantbeeldDoelXls(1) = sKlantbeeldDoelXls(1) & "_" & iVolgnr
      Doel.sBeeldbestandNaam = sKlantbeeldDoelXls(1)
      wkbxlsklantbeeld(1).SaveAs filename:=sKlantbeeldXlsDoelDir & sKlantbeeldDoelXls(1),
```

Voorbeeld uit frmKlantbeeld

Het ophalen van het klantbeeld is als één grote functie geïmplementeerd

- Dit maakt de functie moeilijk leesbaar, waardoor het moeilijker wordt om de gevolgen van aanpassingen op de rest van de functie te voorspellen
- De functie kan worden verkleind door bijvoorbeeld de inhoud van de case statements naar aparte functies te verplaatsen

Onderhoudbaarheid

Mengen van data en logica in hetzelfde script leidt tot grote units

```
SUBSTR("OPTICIENSWERKPLAATSEN (ZONDER DETAILHANDEL)";1;65) IF ECON_BRC = "3832"
SUBSTR("KLOKKEN- EN UURWERKFABRIEKEN";1;65) IF ECON_BRC = "3841"
SUBSTR("DIAMANTSLIJPERIJEN E.D.";1;65) IF ECON_BRC = "3911"
SUBSTR("GOUDEN EN ZILVEREN ARTIKELEN, GOUDEN EN ZILVEREN SIERADENFABR.";1;65) IF ECON_BRC = "3912"
SUBSTR("OVERIGE SIERADENFABRIEKEN, MUNTEN- EN MEDAILLEFABRIEKEN";1;65) IF ECON_BRC = "3913"
SUBSTR("MUZIEKINSTRUMENTENFABRIEKEN (EXCL. ELEKTRONISCHE)";1;65) IF ECON_BRC = "3921"
SUBSTR("FOTO- EN FILMLABORATORIA";1;65) IF ECON_BRC = "3931"
SUBSTR("SPEELGOEDFABRIEKEN";1;65) IF ECON_BRC = "3941"
SUBSTR("SPORTARTIKELENFABRIEKEN (EXCL. KLEDING, SCHOEISEL)";1;65) IF ECON_BRC = "3942"
SUBSTR("SOCIALE WERKPLAATSEN";1;65) IF ECON_BRC = "3951"
SUBSTR("STEMPELFABRIEKEN";1;65) IF ECON_BRC = "3991"
SUBSTR("OVERIGE BE- EN VERWERKENDE INDUSTRIE N.E.G.";1;65) IF ECON_BRC = "3999"
SUBSTR("ELEKTRICITEITSPRODUKTIEBEDRIJVEN";1;65) IF ECON_BRC = "4011"
SUBSTR("ELEKTRICITEITSDISTRIBUTIEBEDRIJVEN";1;65) IF ECON_BRC = "4012"
SUBSTR("GASDISTRIBUTIEBEDRIJVEN";1;65) IF ECON_BRC = "4021"
SUBSTR("WATERWINNINGSBEDRIJVEN";1;65) IF ECON_BRC = "4031"
SUBSTR("WATERDISTRIBUTIEBEDRIJVEN";1;65) IF ECON_BRC = "4032"
SUBSTR("WARMTEVOORZIENINGSBEDRIJVEN (PRODUCTIE)";1;65) IF ECON_BRC = "4041"
SUBSTR("WARMTEVOORZIENINGSBEDRIJVEN (DISTRIBUTIE)";1;65) IF ECON_BRC = "4042"
SUBSTR("GEMENGDE PROD.-DISTR.BEDR. ELEKTR.-, GAS-, WATER, WARMTEVOORZ.";1;65) IF ECON_BRC = "4051"
SUBSTR("GEMENGDE DISTR.BEDR. ELEKTR.-, GAS-, WATER- WARMTEVOORZ.";1;65) IF ECON_BRC = "4052"
SUBSTR("DIRECTEUR/GROOTAANDEELHOUDER";1;65) IF ECON_BRC = "4444"
SUBSTR("AANNEMERSBEDRIJVEN VAN BURGERLIJKE EN UTILITEIT";1;65) IF ECON_BRC = "5111"
SUBSTR("HEIERSBEDRIJVEN (E.A. GESPECIALISEERDE FUNDERING)";1;65) IF ECON_BRC = "5112"
SUBSTR("BETONIJZERVLECHTERSBEDRIJVEN";1;65) IF ECON_BRC = "5113"
SUBSTR("SCHOORSTEEN- EN OVENBOUWBEDRIJVEN";1;65) IF ECON_BRC = "5114"
SUBSTR("DAKDEKKERSBEDRIJVEN";1;65) IF ECON_BRC = "5115"
SUBSTR("SLOPERSBEDRIJVEN (BOUWWERKEN)";1;65) IF ECON_BRC = "5116"
SUBSTR("AANNEMERSBEDRIJVEN VAN TIMMERWERKEN";1;65) IF ECON_BRC = "5117"
SUBSTR("GESPECIALISEERDE AANNEMERSBEDRIJVEN N.E.G.";1;65) IF ECON_BRC = "5119"
SUBSTR("AANNEMERSBEDRIJVEN VAN GROND-, WATER- EN WEGENBOUWK. WERKEN";1;65) IF ECON_BRC = "5121"
SUBSTR("GROND- EN PUTBOORDERIJEN, BRONBEMALINGSBEDRIJVEN";1;65) IF ECON_BRC = "5122"
SUBSTR("KABEL- EN BUIZENLEGGERSBEDRIJVEN (IN DE GROND)";1;65) IF ECON_BRC = "5123"
```

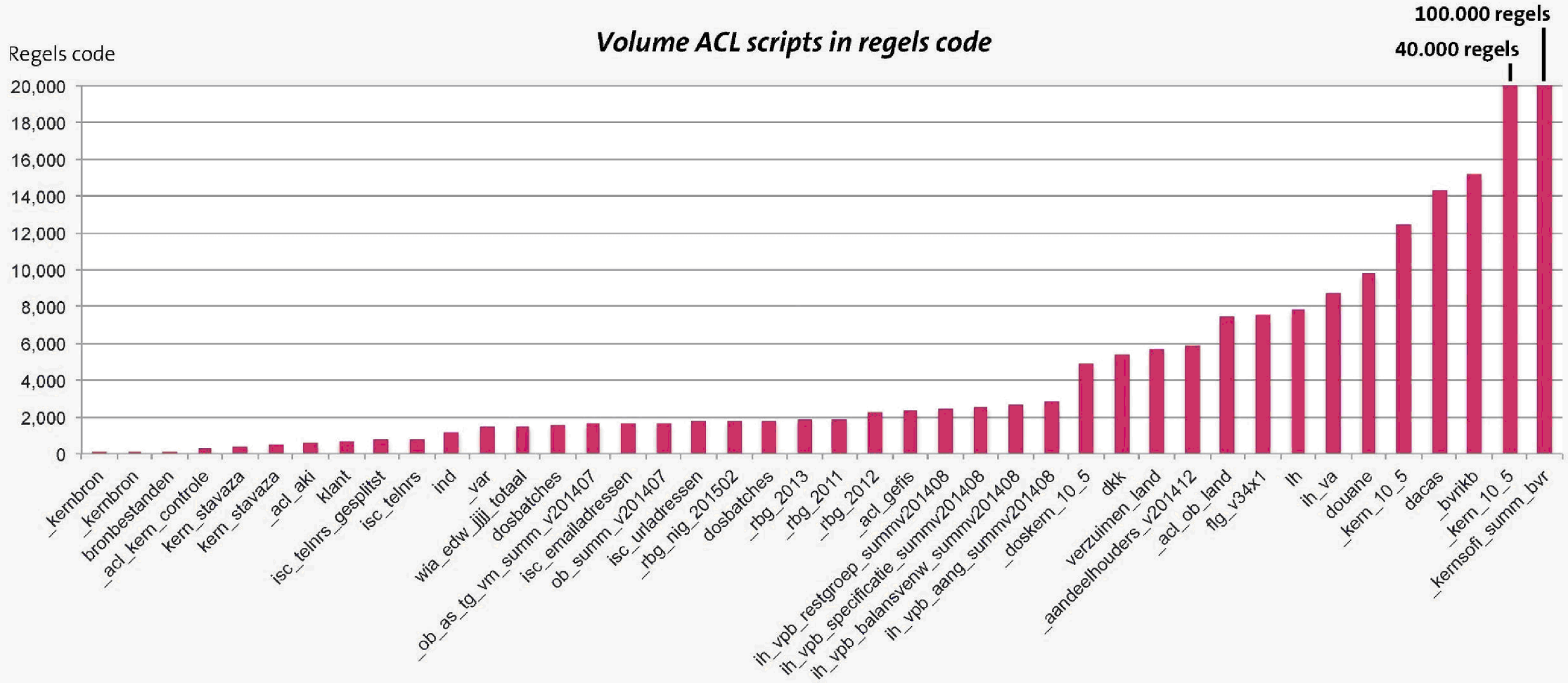
Voorbeeld uit KERN/CSV_INDEX/_KERN_10_5.ACL

De ACL scripts bevatten een combinatie van logica en data

- De data in de scripts is bovendien hard-coded, zoals de branchenamen en branchecodes in het voorbeeld hiernaast

Onderhoudbaarheid

75% van de ACL scripts zijn groter dan 1000 regels



Onderhoudbaarheid

Unit complexity: uitleg van de metriek

Veel complexe units in een systeem vermindert de onderhoudbaarheid

- Simpele units zijn makkelijker te testen
- Simpele units zijn makkelijker te begrijpen
- Simpele units zijn makkelijker te hergebruiken

Gemeten als risicoprofiel op basis van McCabe* complexiteit van units:

```
void removePackage(String name, RuleBase rb) {
   Package[] ps = rb.getPackages();
   if (ps == null) return;
   for (int i = 0; i < ps.length; i++) {
      Package p = ps[i];
      if (p.getName().equals(name)) {
         rb.removePackage(name);
         return;
      }
   }
}
```

McCabe: 4

Cyclomatic complexity	Risico-inschatting
1-5	Duidelijke code, klein risico
6-10	Complex, medium risico
11-25	Erg complex, hoog risico
> 25	Niet begrijpbaar, niet testbaar, zeer hoog risico

* naar: McCabe, *IEEE Transactions on Software Engineering*, 1976

Onderhoudbaarheid

Grote functies zijn vaak ook onnodig complex

```
If bUnion Then
    'effe dimmen
    ReDim sTmpSql(Bron.iAantal)
    'Vul sqlcmd aan
    For x = 2 To Bron.iAantal
        sWhere = ""
        If Len(SplitString(sUnionWhereUitTabelnaam, ",", x)) > 0 Then
            sWhere = sWhere & IIf(Len(sWhere) > 0, " AND ", "") & "(" & SplitString(sUnionWhereUitTabelnaam, ",", x)

        End If
        If Len(Trim(Bron.sUserFilter)) > 0 Then
            sWhere = sWhere & IIf(Len(sWhere) > 0, " AND ", "") & "(" & Bron.sUserFilter & ")"
        End If
        If Len(Trim(Bron.sRegioEnSegmentFilter)) > 0 Then
            If Bron.bBevatRegioEnRamtype Then
                sWhere = sWhere & IIf(Len(sWhere) > 0, " AND ", "") & "(" & Bron.sRegioEnSegmentFilter & ")"
            End If
        End If
        If Len(sWhere) > 0 Then sWhere = " WHERE (  " & sWhere & " )  "

        sTmpSql(x) = "SELECT " & sVelden & vbCrLf & " FROM " & SplitString(sUnionTabellen, ",", x) & vbCrLf &

        'Tijdelijke tabel toevoegen
        If cnOra.GetTempTabelActief Then
            sTmpSql(x) = cnOra.TmpTabel_AddJoin(sTmpSql(x) & vbCrLf, SplitString(sUnionTabellen, ",", x))
        End If
    Next
    Doel.sBestandNaam = Doel.sBestandNaam & "_" & SplitString(sUnionTabellen, ",", x - 1)
    'sSqlcmd en sTmpSql met union aan elkaar knopen
    For x = 2 To Bron.iAantal
        sSqlCmd = sSqlCmd & vbCrLf & "UNION"
        sSqlCmd = sSqlCmd & vbCrLf & sTmpSql(x)
    Next
End If
```

Voorbeeld uit mDataOracleADO

Grote functies leiden tot een groot aantal beslispunten per functie

- Dit maakt aanpassingen moeilijk testbaar (alle beslispunten moeten opnieuw worden getest)
- Het opsplitsen van deze functies in deelproblemen verminderd automatisch het aantal beslispunten, en daarmee de complexiteit

Onderhoudbaarheid

Vastleggen domeinkennis in code leidt tot extra complexiteit

```
"1019AT" IF POSTCODE = "1000BB" AND MATCH(ALLTRIM(ADRES);"CRUQUISWEG";"CRUQUIUSWEG";"CRUQUIUSWEG") AND
ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1082MD" IF POSTCODE = "1000CS" AND ALLTRIM(ADRES) = "CLAUDE DEBUSSYLAAN" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1019GM" IF POSTCODE = "1000CX" AND ALLTRIM(ADRES) = "PIET HEINKADE" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1062EA" IF POSTCODE = "1007JG" AND ALLTRIM(ADRES) = "DELFLANDLAAN" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1101AX" IF POSTCODE = "1008BA" AND ALLTRIM(ADRES) = "DE PASSAGE" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1096AX" IF POSTCODE = "1008BA" AND ALLTRIM(ADRES) = "JOOP GEESINKWEG" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1078HH" IF POSTCODE = "1011EE" AND ALLTRIM(ADRES) = "MAASSTRAAT" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1016NN" IF POSTCODE = "1011RW" AND ALLTRIM(ADRES) = "ROZENSTRAAT" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1260AC" IF POSTCODE = "1012AC" AND ALLTRIM(ADRES) = "POSTBUS" AND ALLTRIM(WOONPLAATS) = "BLARICUM"
"1016CS" IF POSTCODE = "1012HG" AND ALLTRIM(ADRES) = "LEIDSEGRACHT" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1017SE" IF POSTCODE = "1012KM" AND ALLTRIM(ADRES) = "WETERINGSCHANS" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1075AE" IF POSTCODE = "1000AZ" AND ALLTRIM(ADRES) = "KONINGSLAAN" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"3068JH" IF POSTCODE = "1009CL" AND ALLTRIM(ADRES) = "KELLOGGPLAATS" AND ALLTRIM(WOONPLAATS) = "ROTTERDAM"
"2021GC" IF POSTCODE = "1010GC" AND ALLTRIM(ADRES) = "SCHOTERSINGEL" AND ALLTRIM(WOONPLAATS) = "HAARLEM"
"1011AD" IF POSTCODE = "1011AE" AND ALLTRIM(ADRES) = "9401486 B 2881" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1079BL" IF POSTCODE = "1011EE" AND ALLTRIM(ADRES) = "MAASSTRAAT" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1016NN" IF POSTCODE = "1011RW" AND ALLTRIM(ADRES) = "ROZENSTRAAT" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1260AC" IF POSTCODE = "1012AC" AND ALLTRIM(ADRES) = "POSTBUS" AND ALLTRIM(WOONPLAATS) = "BLARICUM"
"1016CS" IF POSTCODE = "1012HG" AND ALLTRIM(ADRES) = "LEIDSEGRACHT" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1017SE" IF POSTCODE = "1012KM" AND ALLTRIM(ADRES) = "WETERINGSCHANS" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1013AD" IF POSTCODE = "1013AC" AND ALLTRIM(ADRES) = "WESTERDOKSDIJK" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1013AT" IF POSTCODE = "1013AJ" AND ALLTRIM(ADRES) = "STAVANGERWEG" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1013AJ" IF POSTCODE = "1013AL" AND ALLTRIM(ADRES) = "GEVLEWG" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
"1013BW" IF POSTCODE = "1013BV" AND ALLTRIM(ADRES) = "TAANDWARSSTRAAT" AND ALLTRIM(WOONPLAATS) = "AMSTERDAM"
```

Voorbeeld uit BVRIKB.ACL

De batch scripts bevatten een groot aantal “magische constanten” waarbij domeinkennis in de code is vastgelegd, zoals bij deze postcodetabel

- Deze functionaliteit is waardevol, maar leidt wel tot extra complexiteit in de code.

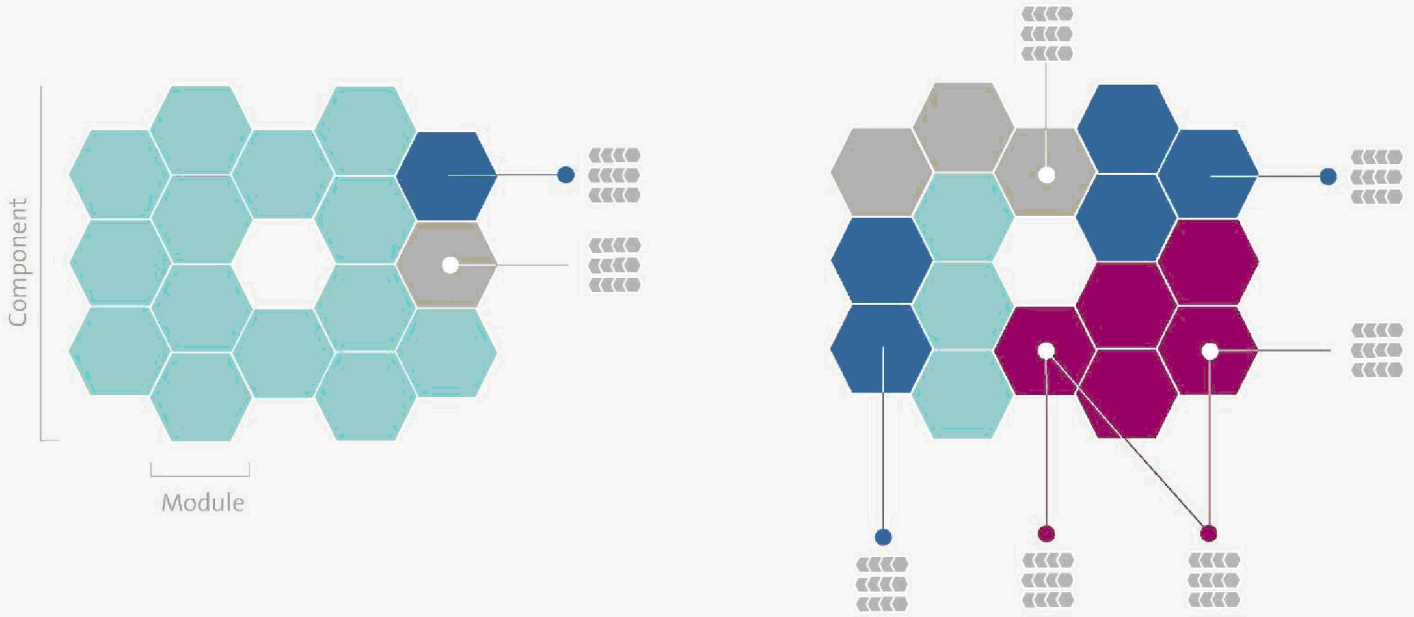
Onderhoudbaarheid

Component independence: uitleg van de metriek

Component independence drukt uit in hoeverre componenten onafhankelijk van elkaar gewijzigd kunnen worden

Wordt gemeten als verhouding 'hidden' vs. 'interface' code:

- *Hidden code*: modules met alleen interne of alleen uitgaande calls
- *Interface code*: modules met alleen inkomende of zowel inkomende als uitgaande (throughput) calls



Onderhoudbaarheid

Geen code-afhankelijkheden, wel afhankelijkheden op bestanden en op de database

Componenten, zowel in VBA/Excel als in ACL, hebben geen directe code-afhankelijkheden op elkaar

- Aanpassingen in de code van component A heeft dus geen gevolgen op de code van component B
- Keerzijde hiervan is de eerder genoemde grote hoeveelheid duplicatie

De VBA/Excel componenten kennen afhankelijkheden op de database-structuur

- Er wordt een “schaduwadministratie” bijgehouden van welke tabellen en kolommen in de database beschikbaar zijn, bij wijzigingen in de database moet ook deze administratie worden aangepast

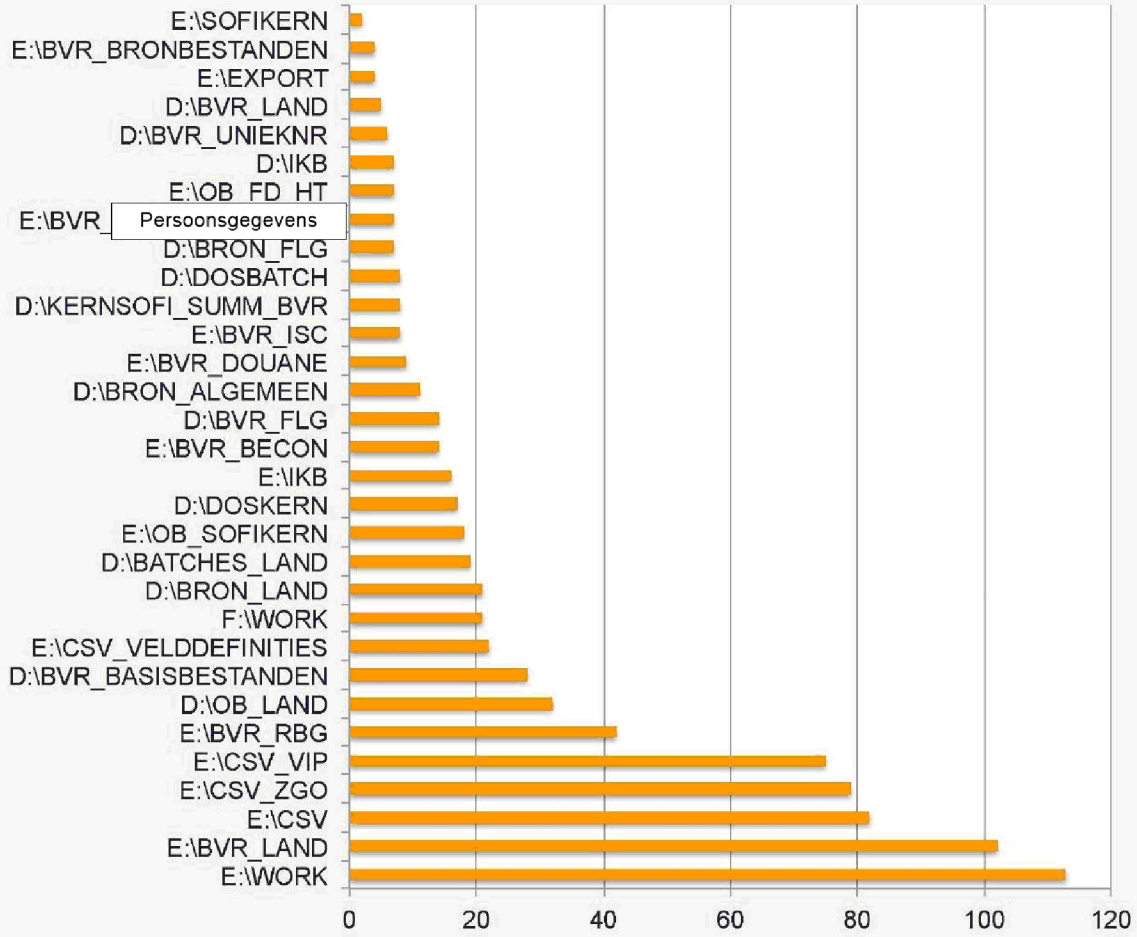
De lokatie van bronbestanden en outputbestanden staat “hard” in de ACL scripts

- De scripts zijn hierdoor afhankelijk van de indeling van het Belastingdienst netwerk

Onderhoudbaarheid

Afhankelijkheden op bronbestanden zijn op basis van “harde” bestandsnamen in de code

Aantal bronbestanden: top 30 meest voorkomende lokaties

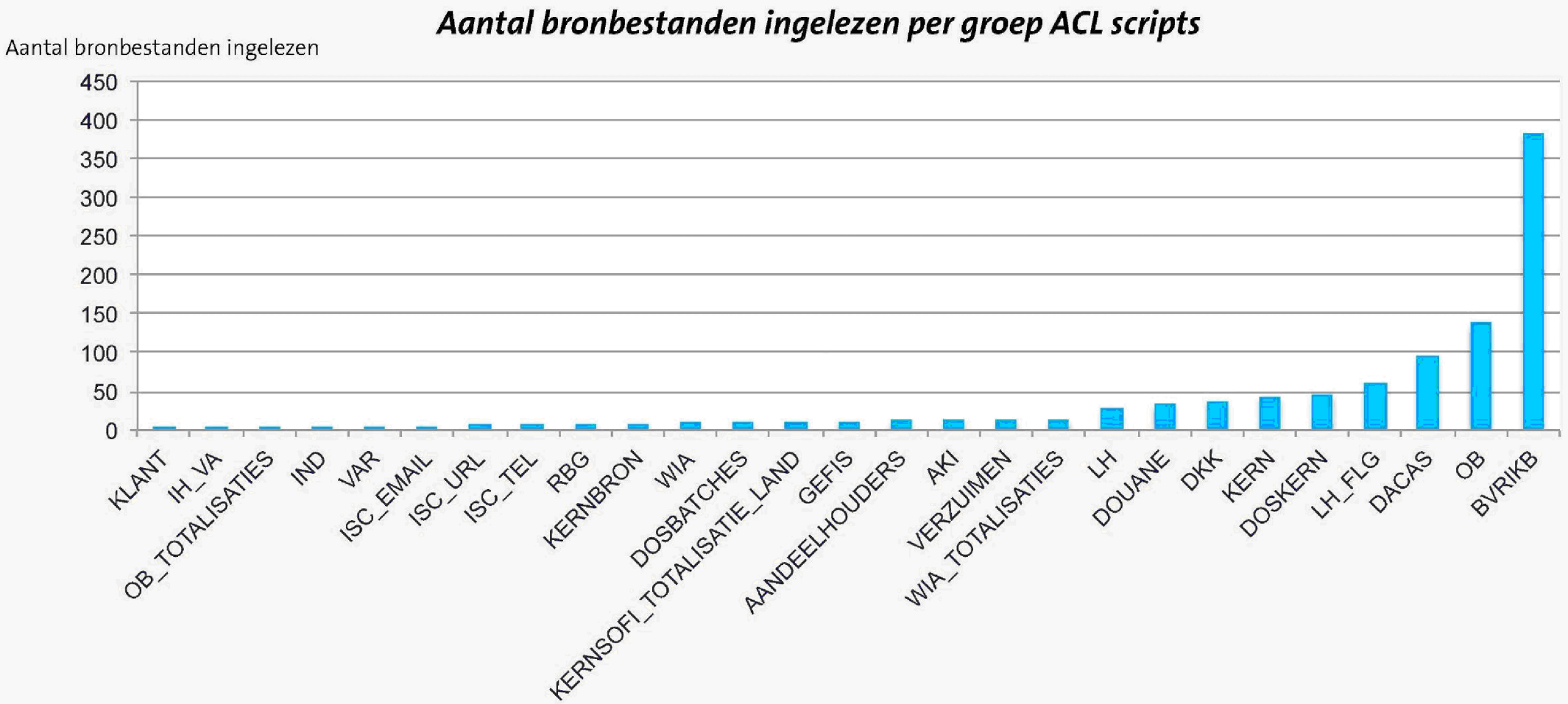


De ACL scripts bevatten afhankelijkheden op 856 verschillende bronbestanden binnen het Belastingdienst netwerk

- Deze bestanden zijn verdeeld over 63 verschillende lokaties

Onderhoudbaarheid

Een groep ACL scripts leest gemiddeld 38 verschillende bronbestanden in



Agenda

1 Onderzoeksvragen en aanpak
2 Technische componenten RAM
3 Volumebepaling
4 Onderhoudbaarheid
5 Ontwikkelproces
6 Samenvatting bevindingen

Ontwikkelproces

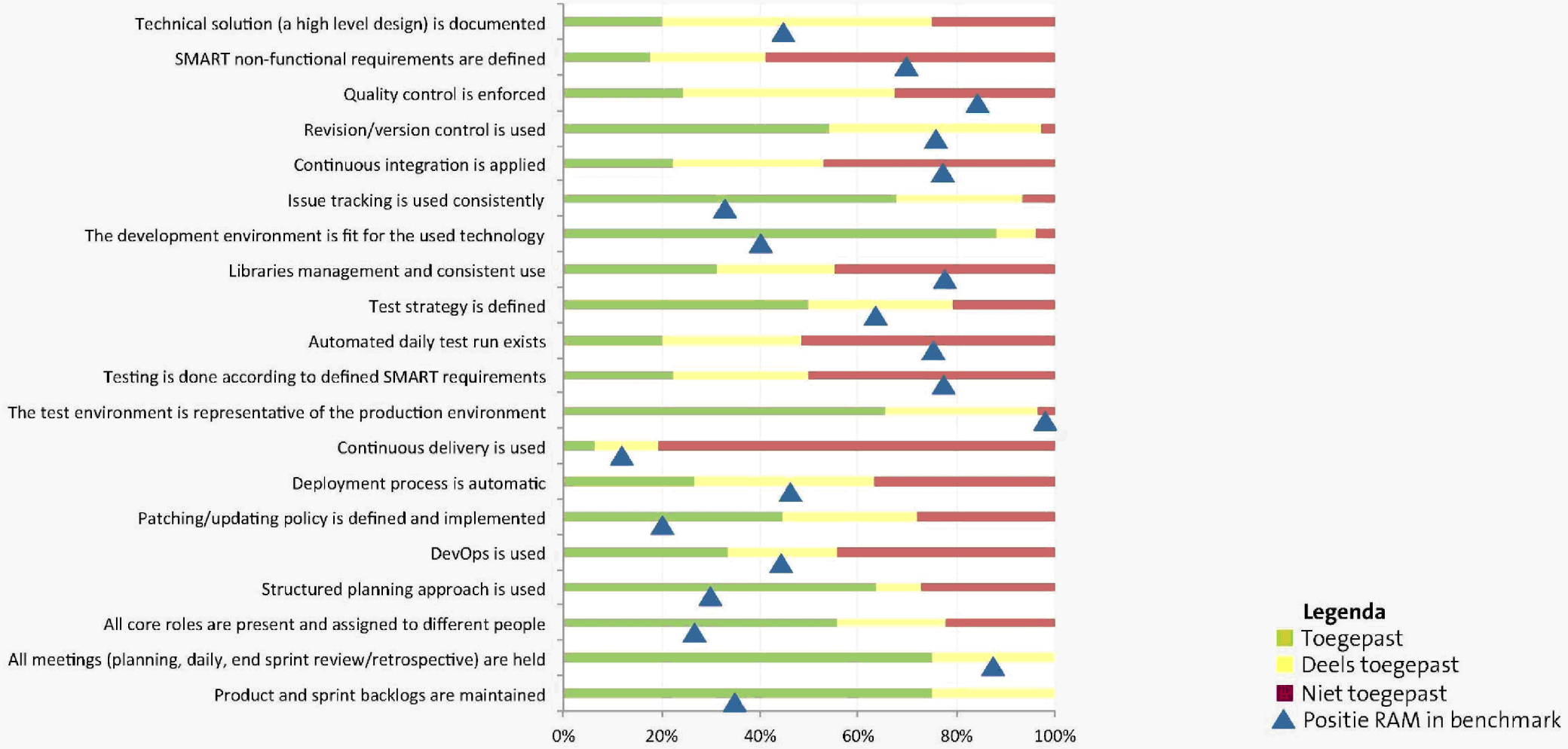
SIG beoordeelt een ontwikkelproces op basis van best practices

SIG heeft het RAM-ontwikkelproces bekeken op basis van het Development Process Assessment model

- Dit model toetst aan de hand van 20 best practices in software-ontwikkeling
- De uitkomsten van deze toetsing worden vervolgens vergeleken met andere ontwikkelprojecten in de SIG benchmark

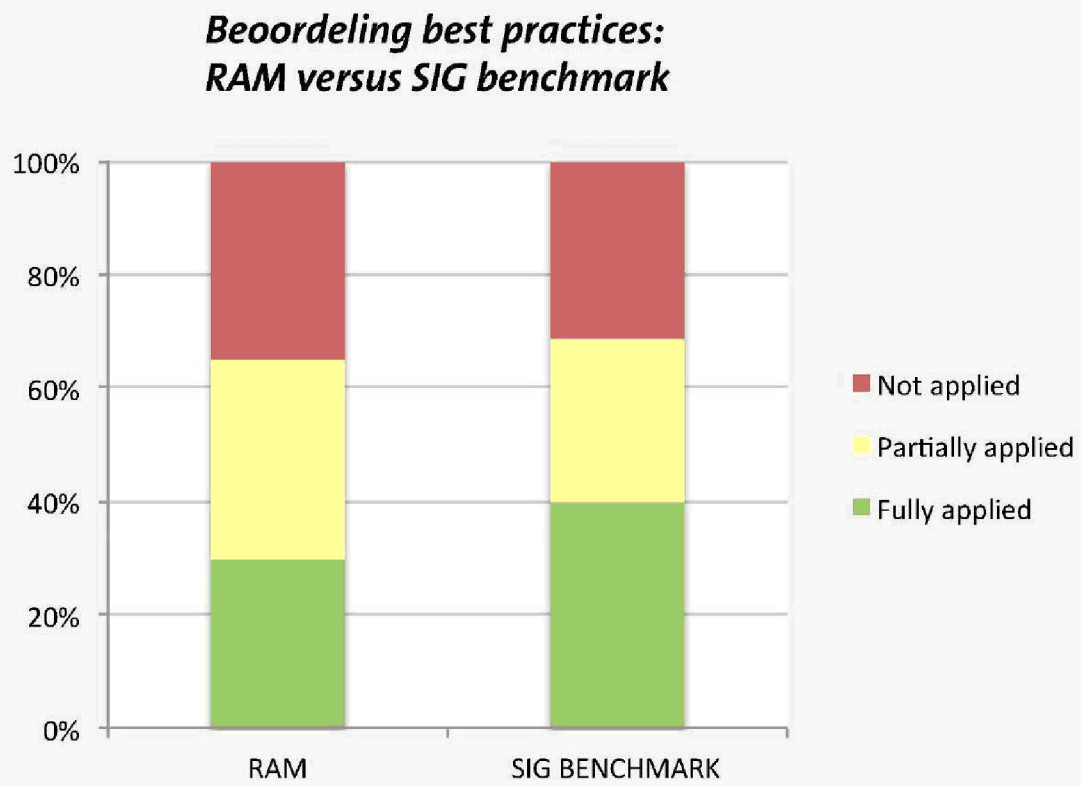
Ontwikkelproces

Vergelijking RAM en SIG benchmark voor best practices ontwikkelprocessen



Ontwikkelproces

Het RAM ontwikkelproces scoort hoog op kortcyclisch werken, laag op automatisering



Over het geheel gezien scoort RAM qua ontwikkelproces iets ondergemiddeld in vergelijking met de SIG benchmark

- Dit totaalbeeld doet echter geen recht aan de werkelijke situatie:
  - RAM scoort zeer hoog op “continuous delivery” aspecten: regelmatige releases, korte lijnen tussen ontwikkelaars en gebruikers, duidelijke en flexibele roadmap
  - Bij de mate van automatisering van het ontwikkelproces, met name de testautomatisering, is wel duidelijk ruimte voor verbetering