BONCODE

Code review
Reis-app
Eindpresentatie v

Klant: GVB

Project: Reis-app

29 augustus 2022

J. de Groot j.degroot@boncode.nl

J. Meetsma j.meetsma@boncode.nl

BonCode Software Remediation BV
Alexanderstraat 4
2514 JL The Hague
The Netherlands

Phone +31 (0)70 222 50 10
Mail: info@boncode.nl
Web: www.boncode.nl
KvK: 61728535

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

Inhoudsopgave

Managementsamenvatting

De Reis-app bestaat uit een groot gedeelte React Native front-end (88% van de codebase) en een kleine C# backend. De totale omvang van de codebase is middelgroot (78 duizend regels code) en bevat gangbare technologieën:

- De decompositie rating (indicatie van onderhoudbaarheid) van C# code is goed en scoort 79 uit 100, 73 is marktgemiddeld.
- De decompositie rating van React Native is 66 uit 100, de voornaamste oorzaak hiervan is:
    - Componenten zijn te veel verweven, functionaliteit zit architectureel niet altijd op de juiste plek
    - Structuur van de codebase heeft te lijden gehad onder druk van doorwerken voor het realiseren van features
    - Code is beperkt analyseerbaar en bevat op verschillende plekken complexe logica

Doorontwikkeling op deze verwaterde structuur heeft het risico bij uitbreidingen de kwaliteit en onderhoudbaarheid verder verergert.

De overdraagbaarheid is enigszins verminderd doch niet problematisch omdat het een middelgrote applicatie is met gangbare technologieën.

BonCode adviseert:

- Gewenste architectuur en bedachte structuur van de codebase te documenteren en de applicatie aan te passen op deze structuur. Documentatie dient beknopt en doordacht opgezet waarin belangrijkste inleidingen en beslissingen zijn vastgelegd.
- Lastig analyseerbare en complexe logica opschonen wanneer er toevoegingen/uitbreidingen in dat gedeelte van de code plaatsvindt.

Beantwoording onderzoeksvragen

1. How easy is it to take over the software maintenance for a new company?

   "We beantwoorden deze vraag door het beoordelen van de overdraagbaarheid van de Reis-app. Deze is enigszins verminderd doch niet problematisch als gevolg van een relatief kleine applicatie met gangbare technologieën. De verminderde overdraagbaarheid komt door een niet geheel uitgekristalliseerde structuur van de codebase, complexiteit binnen deel van de front-end modules en ontbrekende documentatie."

2. How safe and robust is it to add extra functionality to the existing software?

   "Het toevoegen van functionaliteit is mogelijk en de applicatie loopt nog niet tegen grenzen op. Verbetering in de uitbreidbaarheid is haalbaar door gewenste architectuur en structuur in de codebase te documenteren en de applicatie er op aan te passen. Daarnaast is het aan te raden om bij aanpassingen of uitbreiden van bestaande complexe code deze eerst op te schonen. Streef naar een decompositie rating van 70 of hoger."

3. What is the quality of the software up till now?

   "Softwarekwaliteit wordt gemeten d.m.v. de decompositie rating. De backend, C#-code, heeft een rating van 79 uit 100. De front-end in React Native omvat 88% van de codebase en heeft een rating van 66 uit 100 (dit is beneden marktgemiddeld, nl. 73) De grootste uitdagingen t.a.v. softwarekwaliteit zijn terug te vinden in de front-end."

2. Aanleiding en scope

Aanleiding van het onderzoek

- GVB verzorgt het openbaar vervoer in en rond Amsterdam. Dagelijks reizen meer dan 900.000 mensen met door GVB beheerde metro's, trams en bussen en met de veren over het IJ en Noordzeekanaal.
- Software speelt een essentiële rol om een goede dienstverlening naar de burger mogelijk te maken. Om de reiziger optimaal te informeren heeft GVB de reis app laten ontwikkelen door Infi.
- Deze gratis GVB app (voor iOS en Android) is dé app voor het reizen met tram, (nacht)bus, metro en veerpont in Amsterdam en in de rest van Nederland. De app maakt het simpel en snel plannen mogelijk:
    - Voor het maken van een reis, en,
    - te checken of er een omleiding of vertraging op de te maken route is.
- De app zal nog meer geïntegreerd worden met de GVB website. Hiertoe zal een aanbesteding in de markt komen.
- Om voor mogelijke inschrijvers een level playing field te creëren heeft GVB aan BonCode gevraagd de kwaliteit en overdraagbaarheid van de ontwikkelde code van de reis-app te onderzoeken.

Scope

Het onderzoek zal zich alleen richten op een onderzoek van de source code. Documentatie en processen zijn buiten scope van deze opdracht.

Onderwerp van dit onderzoek is de volgende applicatie met de technologieën:

	GVB-reisapp (Infi)
Broncode	05-07-2022 Applicatie: GVB reis-app - React-native - dotnet core (5 and 6), - sql db - azure api management gateway - azure service bus

3. Onderzoeksvragen, proces en aanpak

Onderzoeksvragen

De volgende onderzoeksvragen zijn geformuleerd:

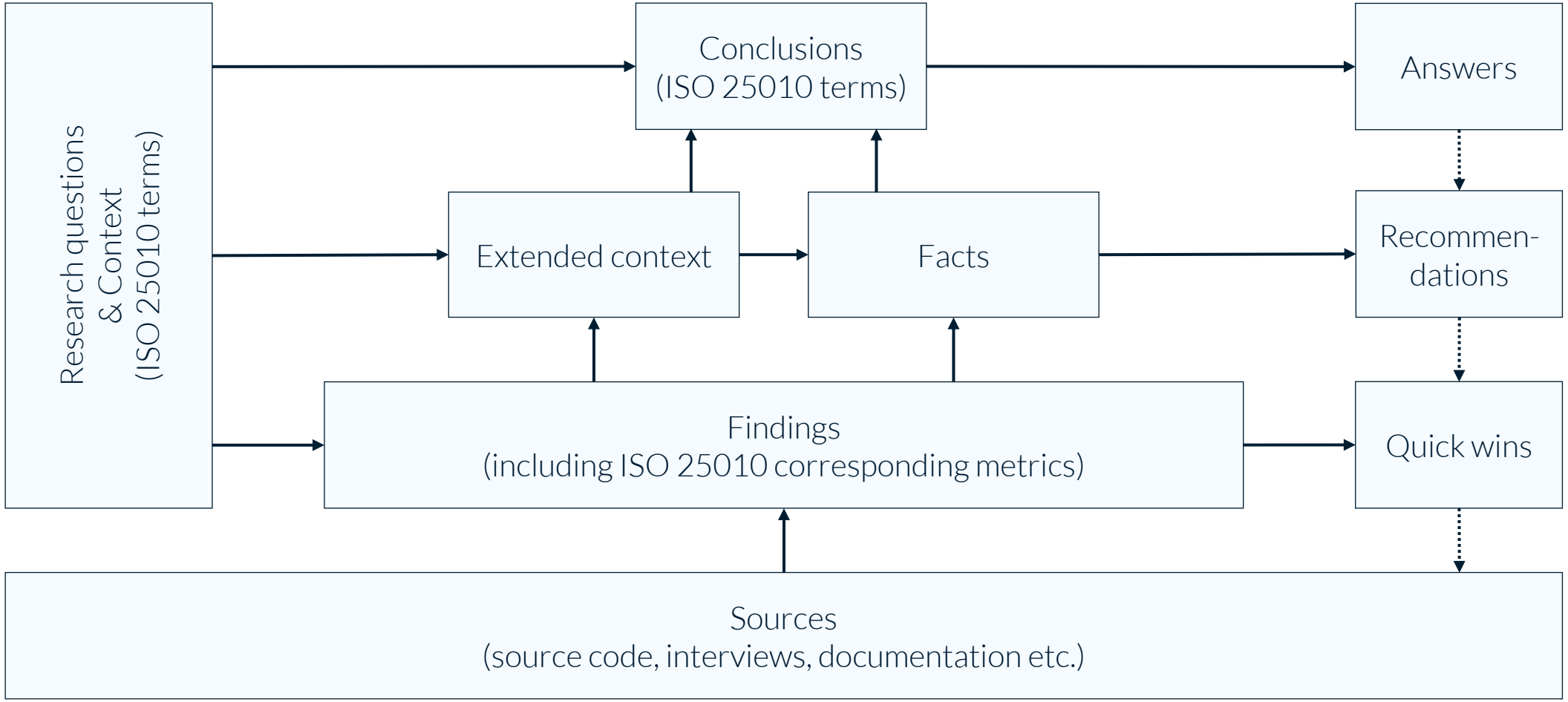
1. How easy is it to take over the software maintenance for a new company?
2. How safe and robust is it to add extra functionality to the existing software?
3. What is the quality of the software up till now?

Deze drie vragen komen neer op een onderzoek naar de Onderhoudbaarheid* van de tot op heden opgeleverde code. Het onderzoek en het beantwoorden van de gestelde vragen zal zich dan ook richten op dit onderdeel.

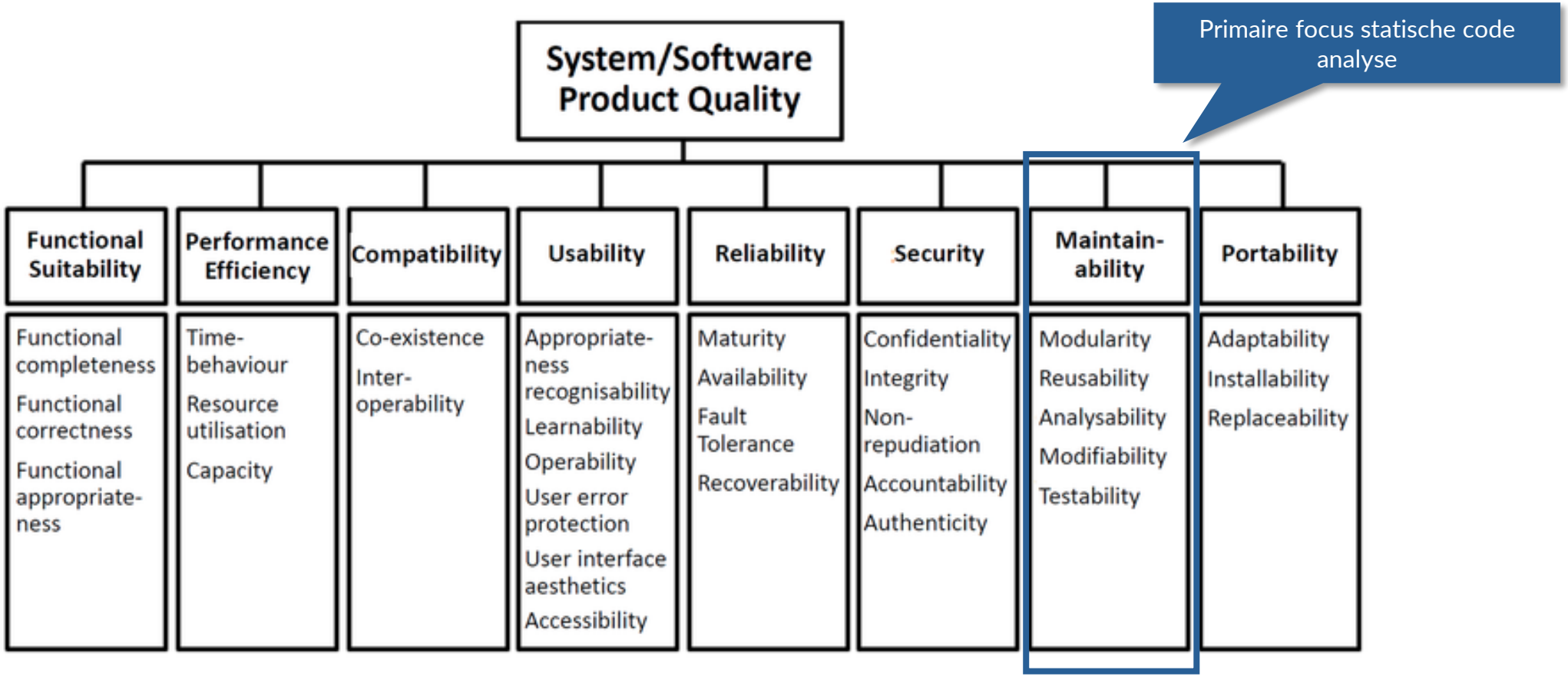
**) De kwaliteitseigenschap 'Onderhoudbaarheid' van de ISO 25010 norm luidt: "De mate waarin een product of systeem effectief en efficiënt gewijzigd kan worden door de aangewezen beheerders."*

Onderzoeksproces

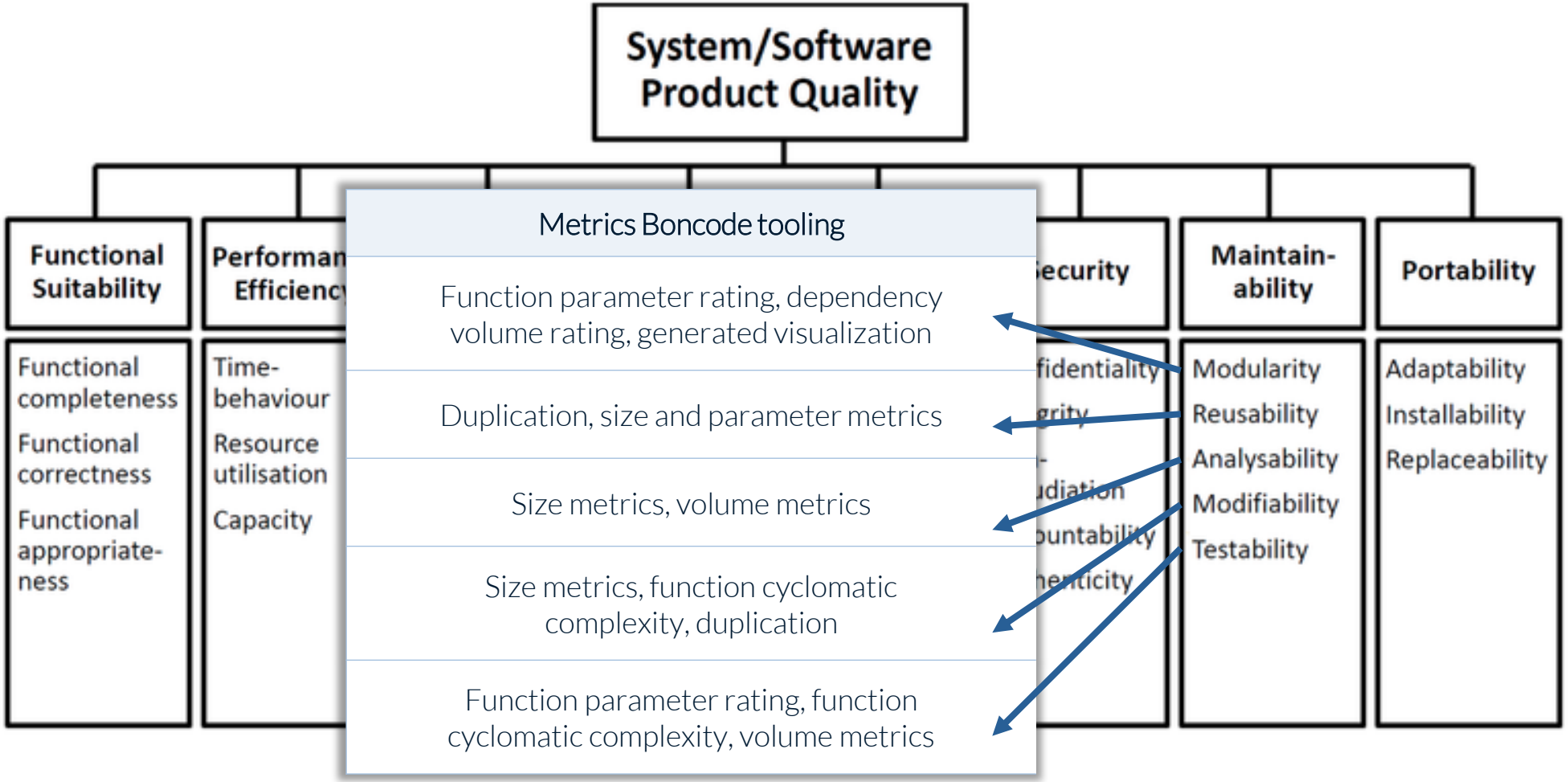
Unambiguous, fact-based founded advice



ISO 25010 software quality aspects (1/2)



ISO 25010 software quality aspects (2/2)



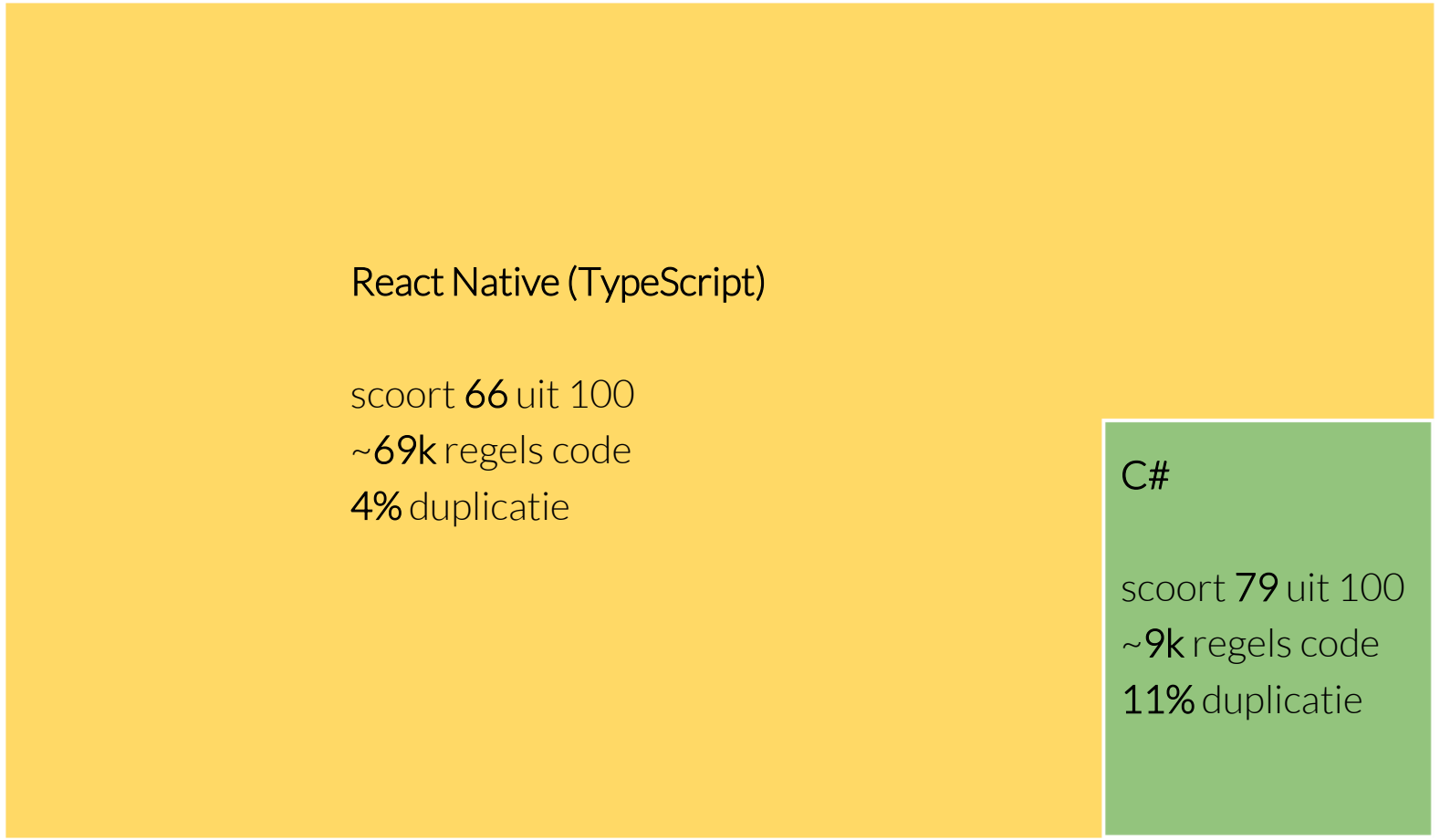
4. Bevindingen

Overzicht belangrijkste bevindingen

Volume en decompositie rating

Reis-app

Gewogen score van **68**
~**78k** regels code



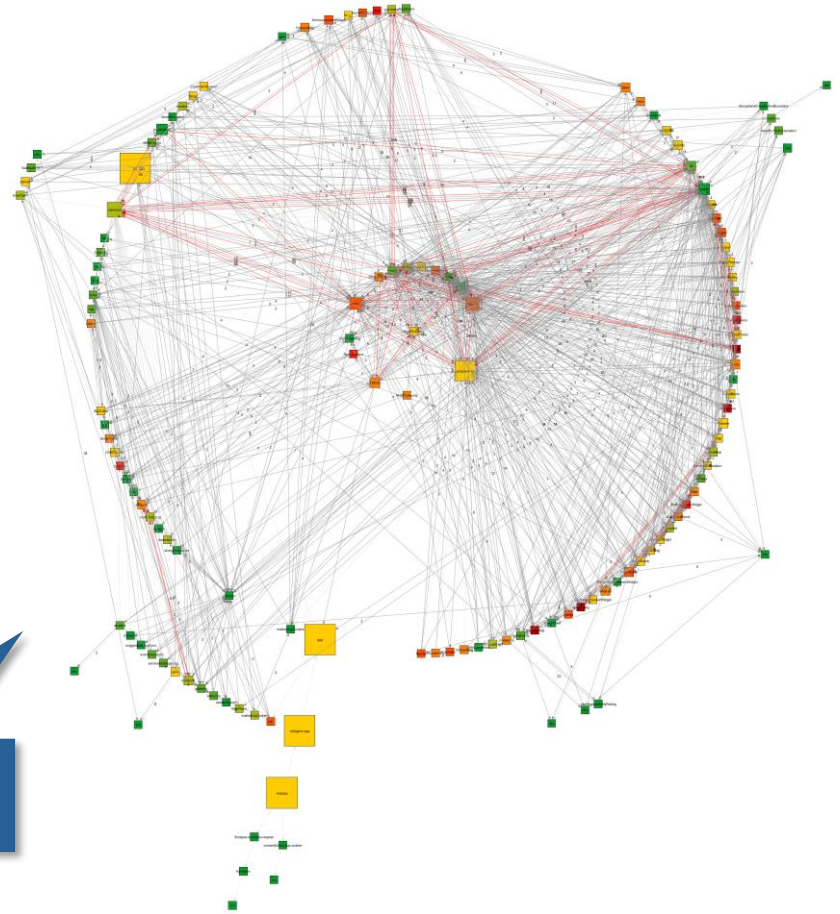
React Native – Cyclische afhankelijkheden (1/2)

Verwijzingen (pijlen) in het rood zijn cyclisch

Structuur van de codebase is niet uitgekristalliseerd. Architectureel zitten componenten niet altijd op de juiste plek.

Cyclische afhankelijkheden in de software zegt iets over de modulariteit en testbaarheid van de software. Wanneer cyclische afhankelijkheden bestaan tussen stukken software is het inherent lastiger om code aan te passen of te hergebruiken.

Over het algemeen is zijn cyclische afhankelijkheden een indicatie dat modules of componenten in de codebase niet op hun plek zitten of dat bepaalde concepten niet vastgelegd zijn in de structuur van de code.



395 referenties gemarkeerd als **cyclisch**.

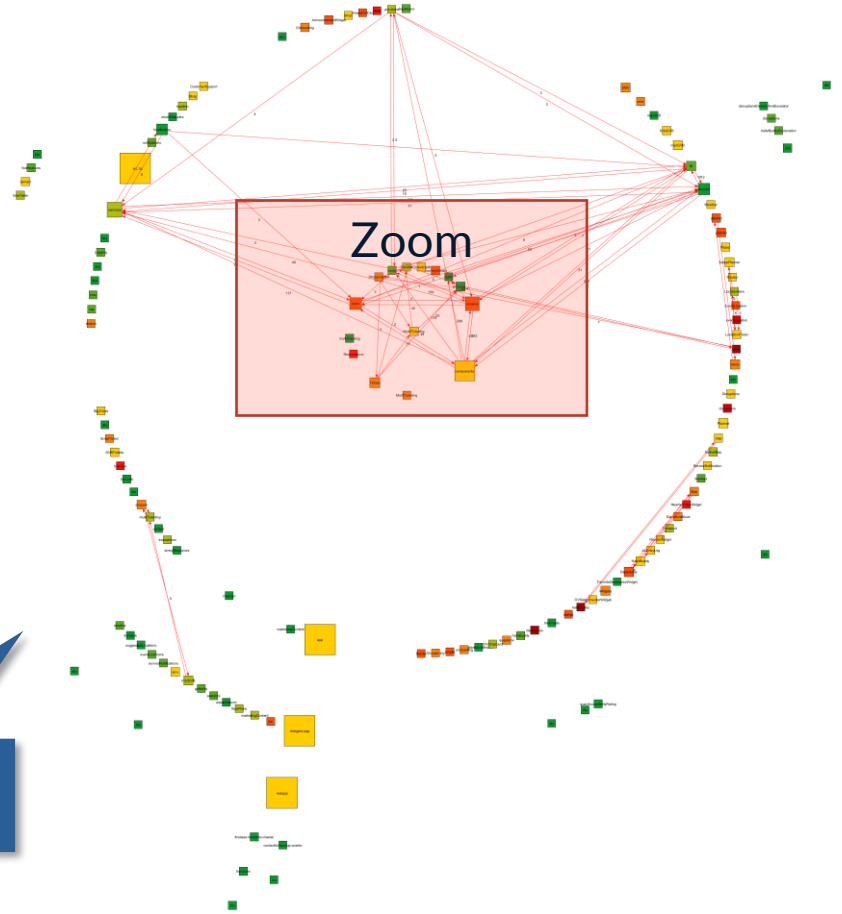
React Native – Cyclische afhankelijkheden (1/2)

Verwijzingen (pijlen) in het rood zijn cyclisch

Structuur van de codebase is niet uitgekristalliseerd. Architectureel zitten componenten niet altijd op de juiste plek.

Cyclische afhankelijkheden in de software zegt iets over de modulariteit en testbaarheid van de software. Wanneer cyclische afhankelijkheden bestaan tussen stukken software is het inherent lastiger om code aan te passen of te hergebruiken.

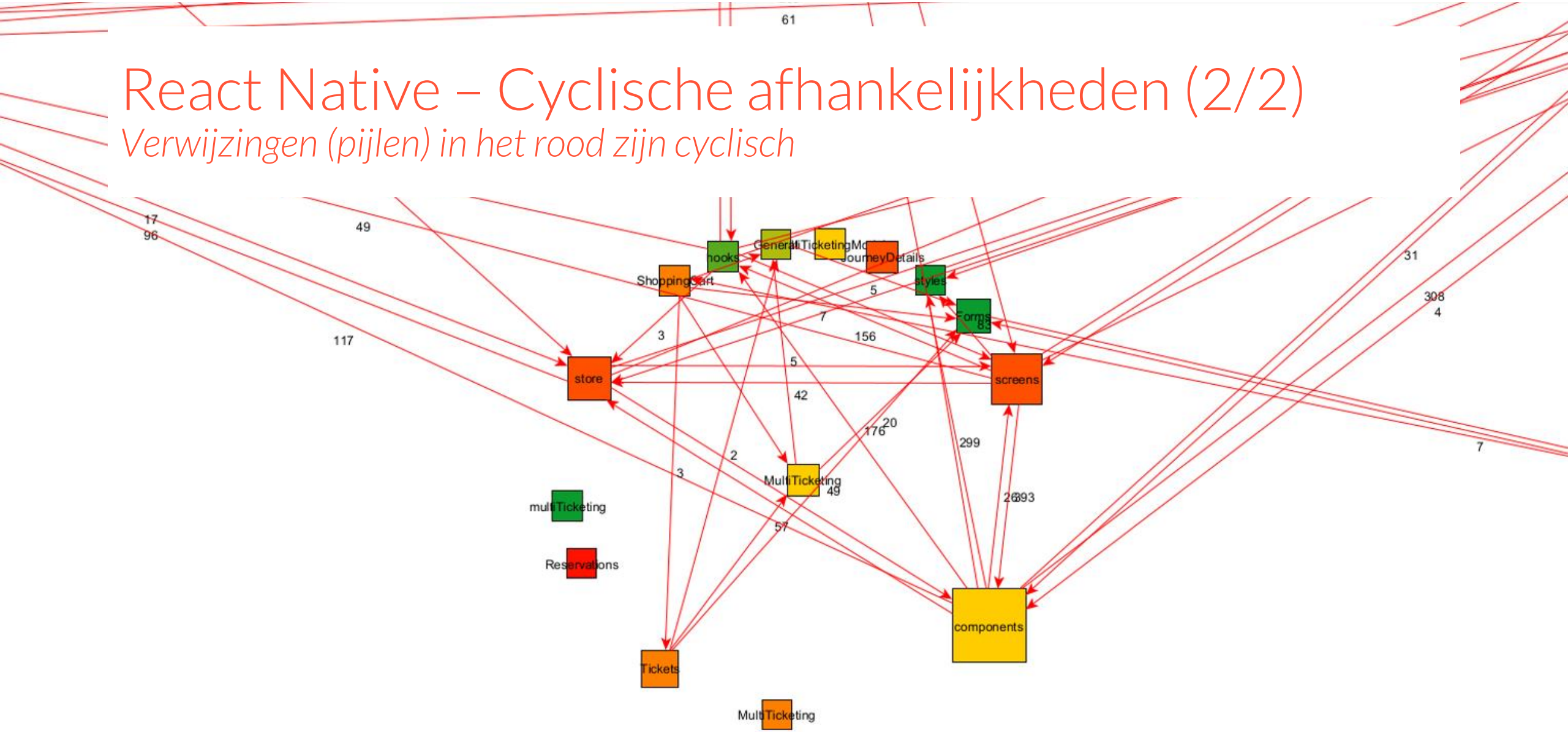
Over het algemeen is zijn cyclische afhankelijkheden een indicatie dat modules of componenten in de codebase niet op hun plek zitten of dat bepaalde concepten niet vastgelegd zijn in de structuur van de code.



395 referenties gemarkeerd als **cyclisch**.

React Native – Cyclische afhankelijkheden (2/2)

Verwijzingen (pijlen) in het rood zijn cyclisch



Beperkte analyseerbaarheid en complexiteit in logica

- Code is beperkt analyseerbaar en bevat op verschillende plekken complexe logica.
  - Voornamelijk in initieel opgezette code.
  - Complexiteit komt onder andere voor in:
    - Amyyon service
    - Ticket migratie
  - De definities van Redux stores leiden tot grote functies die veel (soms complexe) logica bevatten.
  - Grote schermen, bijvoorbeeld het Dashboard scherm, bevatten een grote hoeveelheid afhankelijkheden die verspreid zijn over de codebase.

5. Aanbevelingen

Voorkom doorontwikkeling op verwaterde structuur

Beperk het risico van verdere verslechtering van de kwaliteit en overdraagbaarheid. Reserveer tijd om:

- Gewenste architectuur en bedachte structuur van de codebase te documenteren en de applicatie daarop in te richten
- Documentatie dient beknopt en doordacht opgezet waarin belangrijkste inleidingen en beslissingen zijn vastgelegd.

Incorporeer kwaliteitsverbeteringen gedurende toekomstige software ontwikkelingen (toevoegingen/uitbreidingen)

- Schoon lastig analyseerbare en complexe logica op wanneer eraan gewerkt wordt*
- Streef dan naar een decompositie rating van 70 of hoger voor aangeraakte gedeelten in de code
- Verbeter tegelijkertijd de testdekking, dit is sterk van toegevoegde waarde bij complexere onderdelen van de codebase

**top 10 lijsten van functiecomplexiteit zijn terug te vinden in het dashboard*

Vragen, opmerkingen?

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

Software facts for everyone

HOW WE DO IT

- Fully independent, evidence-based and actionable facts about your software
- Focus on **maintainability** is key
- Optimized for the various roles in your organisations (not only developers!)
- Trigger a positive chain reaction of constructive discussions

Technology agnostic

STANDARDIZED APPROACH FOR ANY LANGUAGE