Progress®

My Code Is Better Than Yours

PUG Challenge 2024, Prague

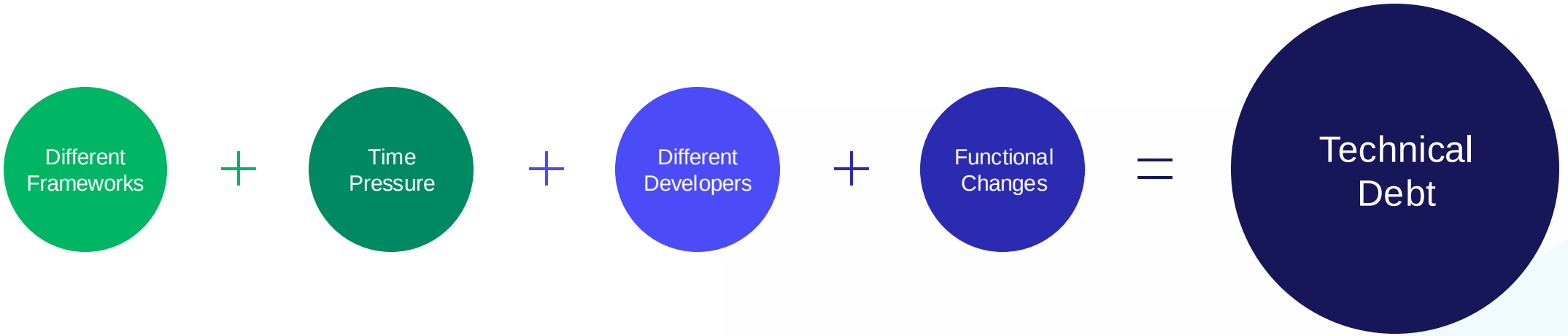
Jochen Zimmerman
Lead Enterprise Architect, proALPHA

Roland de Pijper
Sr Principal Consultant, Progress Software

Agenda

- History Lesson
- Who is SIG?
- Demo
- proALPHA Case Study
- Final Thoughts

Your Code Over Time



Procedural code

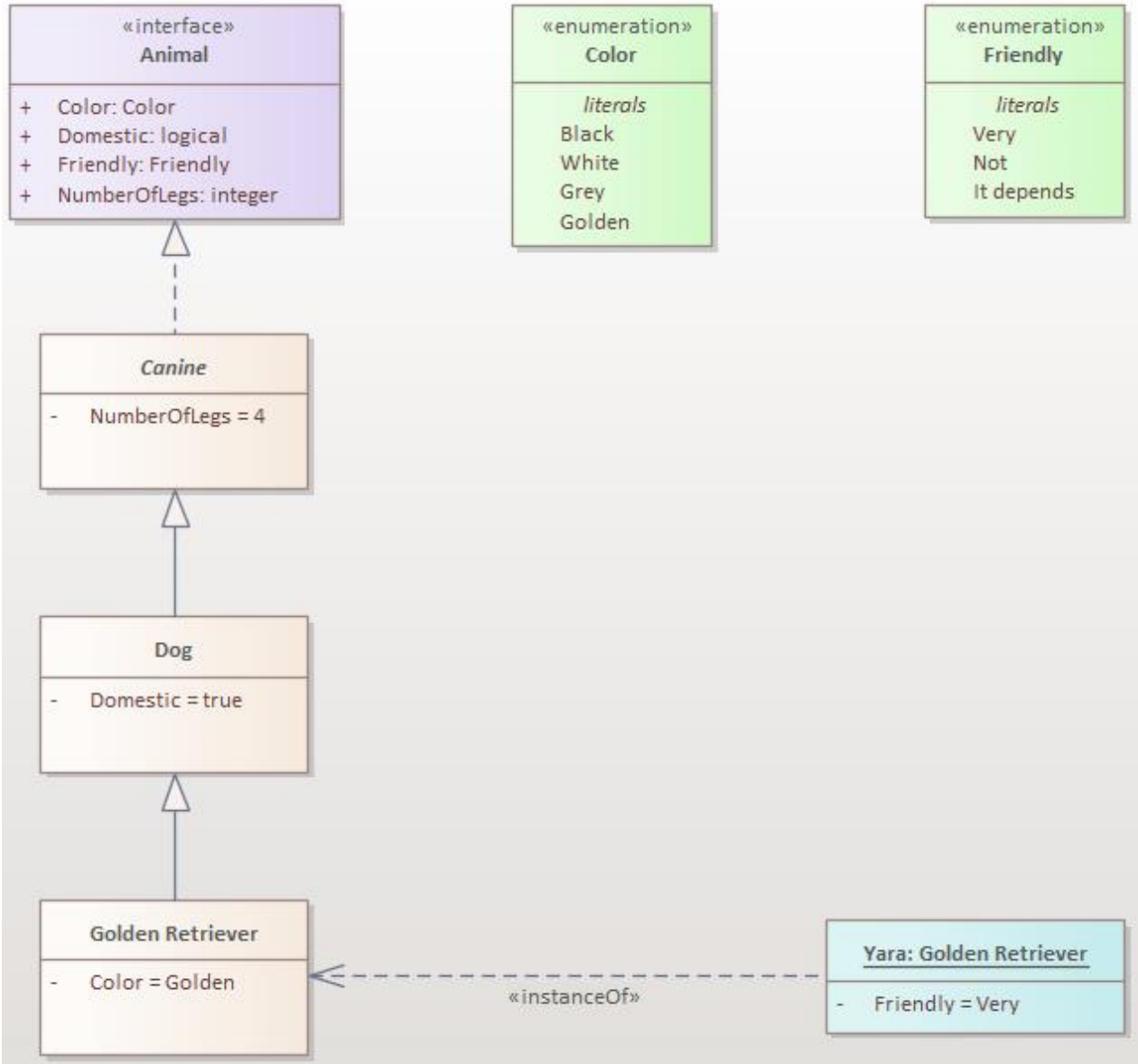
```
DEFINE VARIABLE cColor         AS CHARACTER INITIAL 'Golden' NO-UNDO.
DEFINE VARIABLE cFriendly      AS CHARACTER INITIAL 'Very'   NO-UNDO.
DEFINE VARIABLE iNumberOfLegs AS INTEGER   INITIAL 4        NO-UNDO.
DEFINE VARIABLE hYara          AS HANDLE                    NO-UNDO.

RUN goldenretriever.p(INPUT cColor,
                      INPUT cFriendly,
                      INPUT iNumberOfLegs)
    PERSISTENT SET hYara.
```

State-of-the-art OO Implementation

VAR Animal Yara =

NEW Golden_Retriever(Friendly:Very).

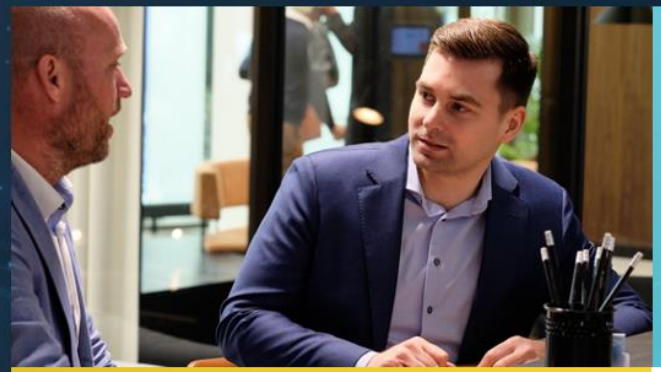


About Software Improvement Group

GETTING SOFTWARE RIGHT
FOR A HEALTHIER DIGITAL WORLD

Scientific research

The SIG research department develops our measurement models and contributes to advancements in the field of software engineering.



Sigrid

Sigrid® provides (continuous) insight into software build quality, costs, security, and other risks, to support ROI based transformations

Benchmarking

The SIG software analysis database is the largest in the world, containing more than 85 billion lines of code.

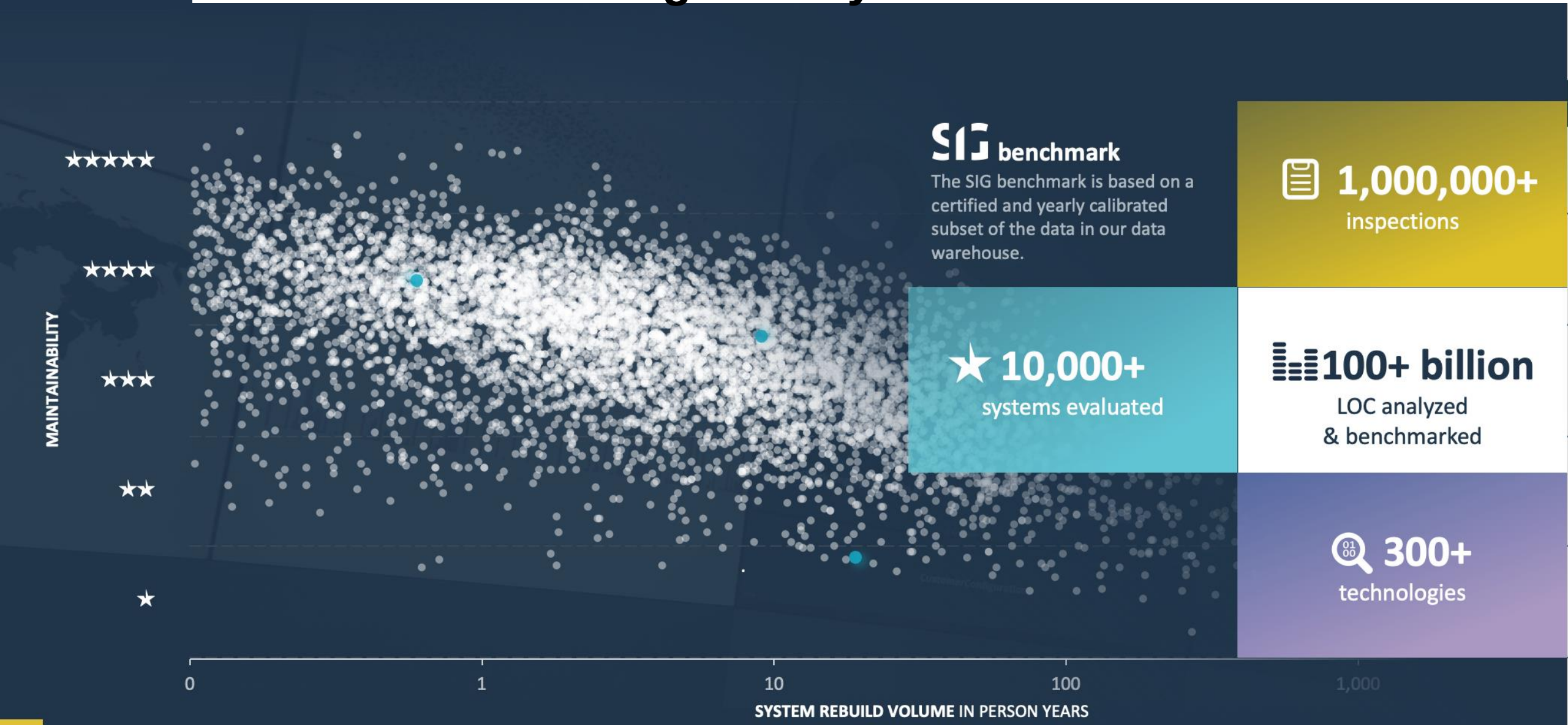
Sigrid®| Landscape Scan

Full scan of all software code to provides fact-based, risk identification within 2 weeks for prioritization, budgeting and planning purposes.

TÜViT® Certification

SIG is the first company in the world with a laboratory accredited by TÜViT to certify software for ISO 25010.

A benchmarked approach to reduce software risks, costs while increasing velocity



SIG Quality and Security Management

OpenEdge Application Quality and Security Management Service (QSM)

Goals

Set your goals
Security findings
Maintainability
Test code ratio
Architecture quality

Maintainability

Maintainability score compared to the industry
Where are the risks
Where to improve

Security

Automated finding of security threats in your codebase
Based on industry standards (OWASP, ISO 5055, CWE, PCI DSS)

Architecture

Shows the architecture as it is implemented and how it's being maintained
Helps you to find opportunities on how it can be improved

DEMO

Maintainability

Volume

Having an independent system will ease maintenance

Duplication

Write code once

Duplicated code wastes time, as future changes will need to be applied to all copies. This might also introduce bugs if you inadvertently forget to update one of the copies.

Unit size

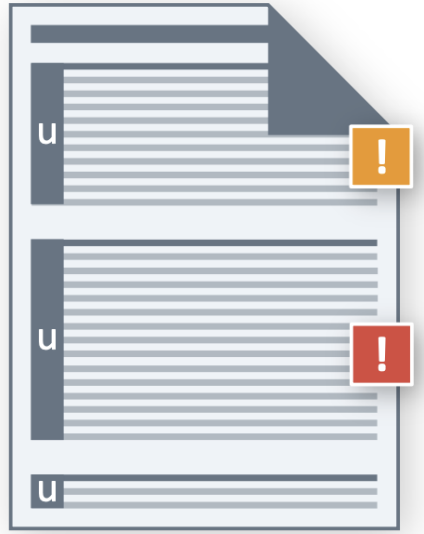
“The first rule of functions is that they should be small.

The second rule of functions is that they should be smaller than that.”

Example

Recommended practice

Inadvisable practice



Risk categories

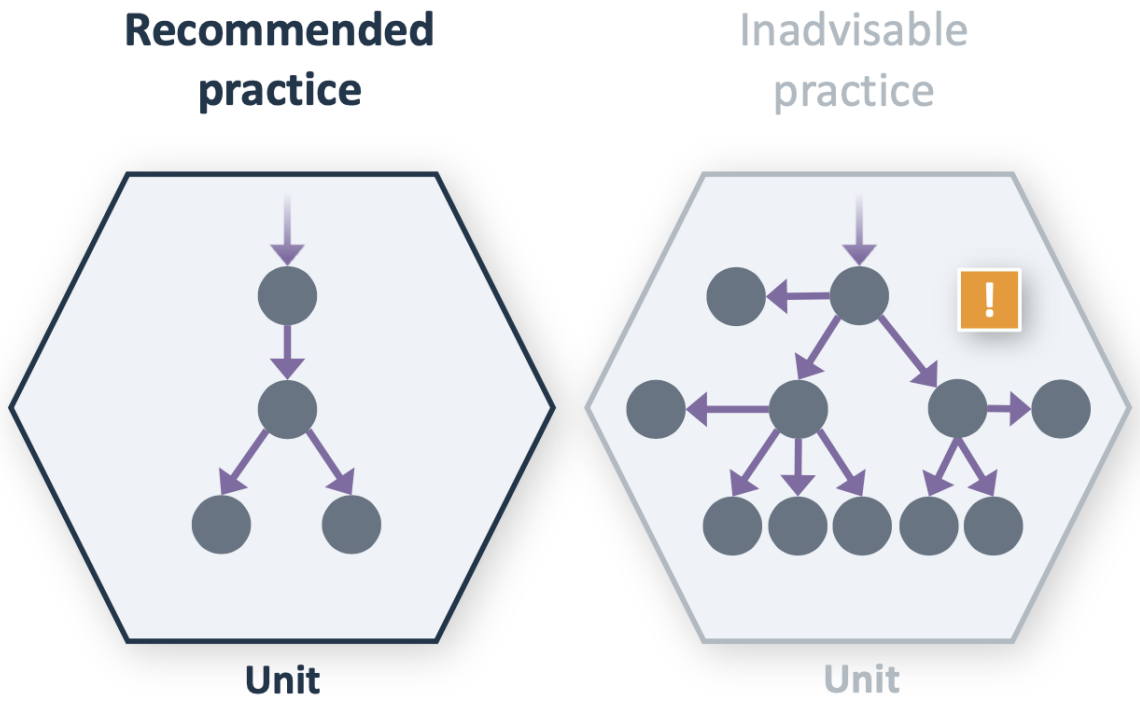
1-15 LOC	16-30 LOC	31-60 LOC	61+ LOC
low risk	moderate risk	high risk	very high risk

Unit complexity

McCabe Cyclomatic Complexity

NUMBER_OF_BRANCHES + 1

Example



Risk categories

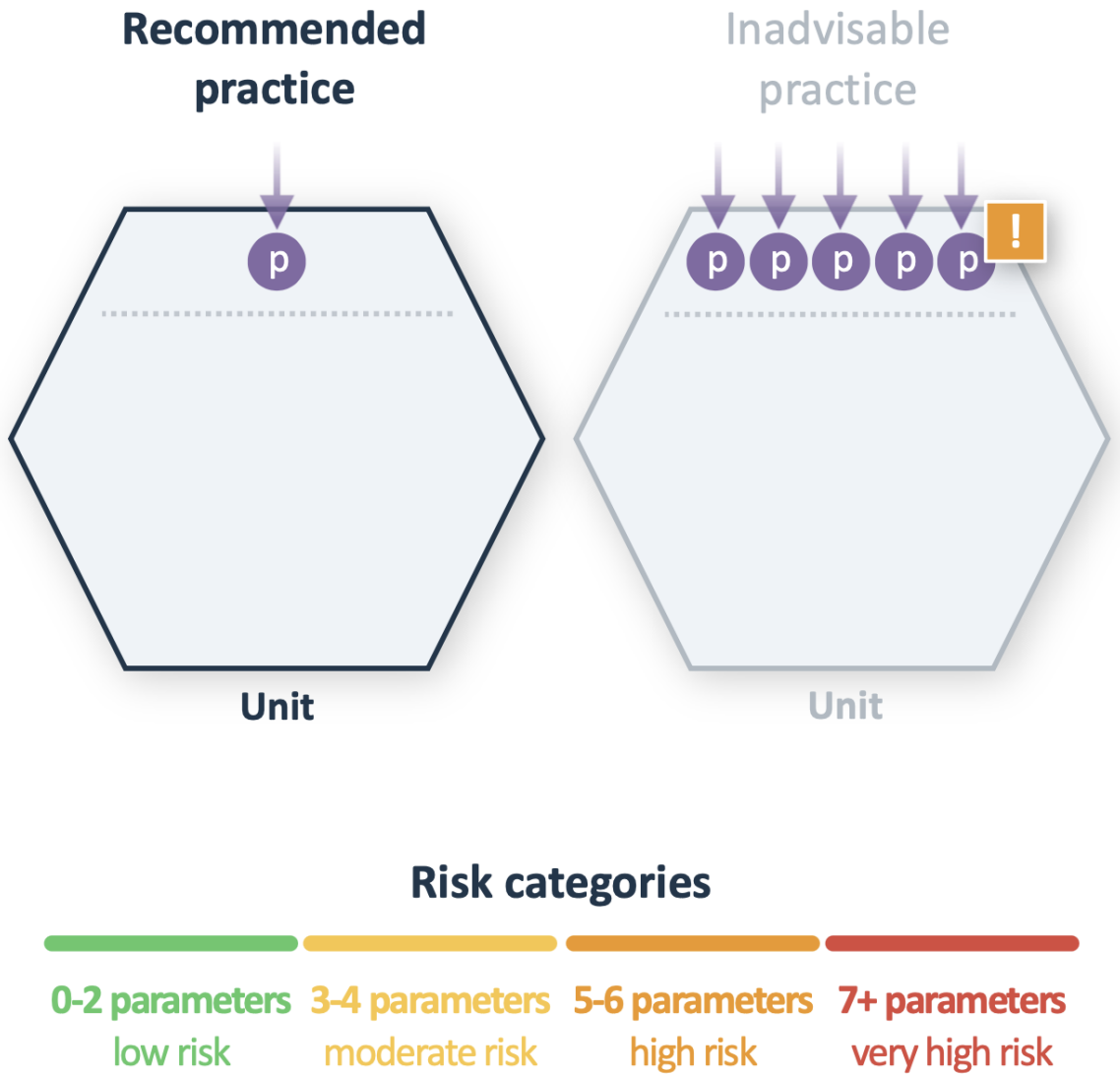
1-5 McCabe	6-10 McCabe	11-25 McCabe	26+ McCabe
low risk	moderate risk	high risk	very high risk

Unit interfacing

Keep unit interfaces small

Avoid creating procedures/methods that take many parameters, as it makes them inconvenient to call or reuse.

Example



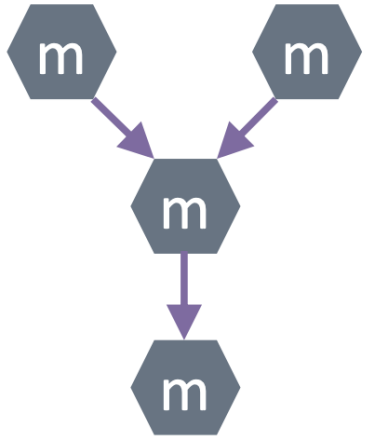
Module coupling

Separate concerns in modules

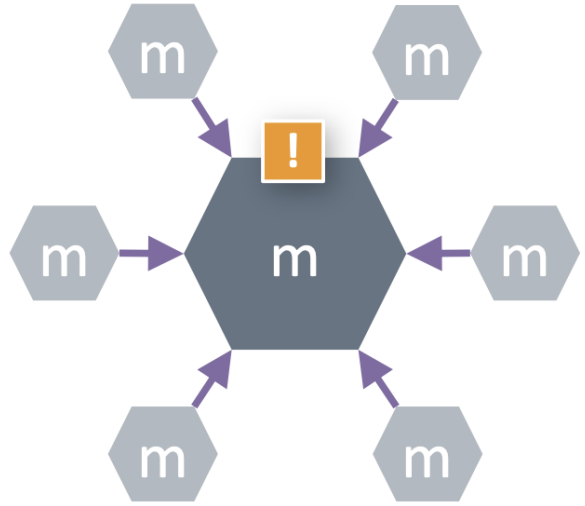
Separation of concerns leads to smaller and more loosely coupled modules (i.e. files).

Example

Recommended practice



Inadvisable practice



Risk categories

0-10 fan-in	11-20 fan-in	21-50 fan-in	51+ fan-in
low risk	moderate risk	high risk	very high risk

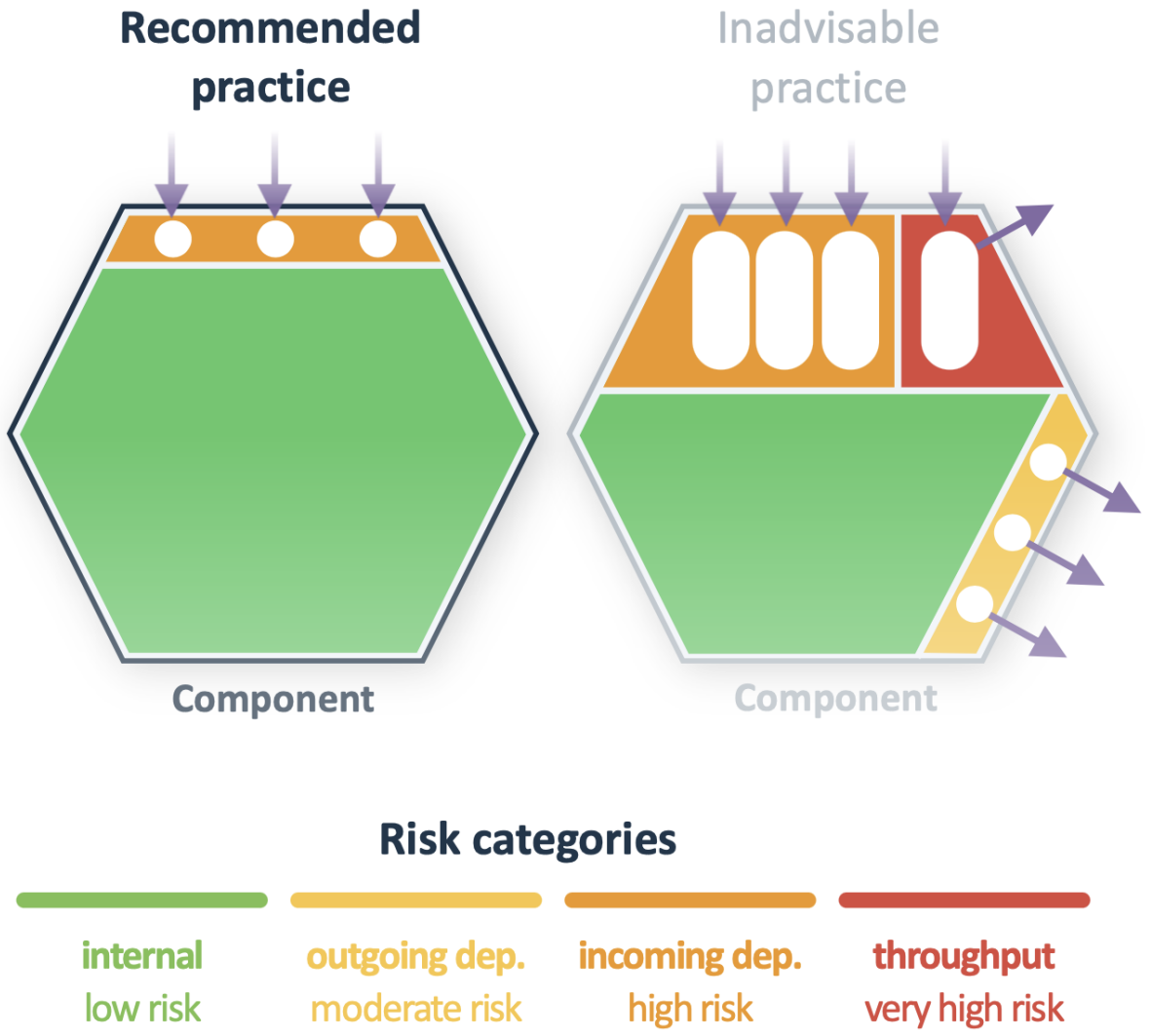
Progress®

Component independence

Couple architecture components loosely

Separate components into an interface, that receives incoming communication from other components, and an internal part.

Example

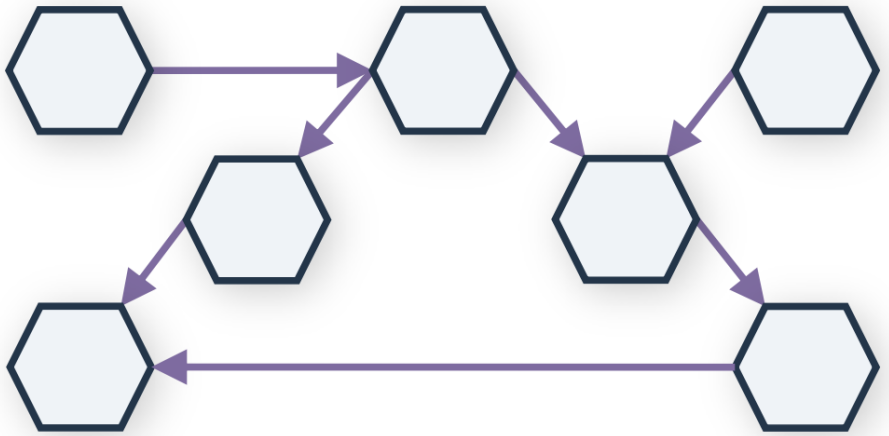


Component entanglement

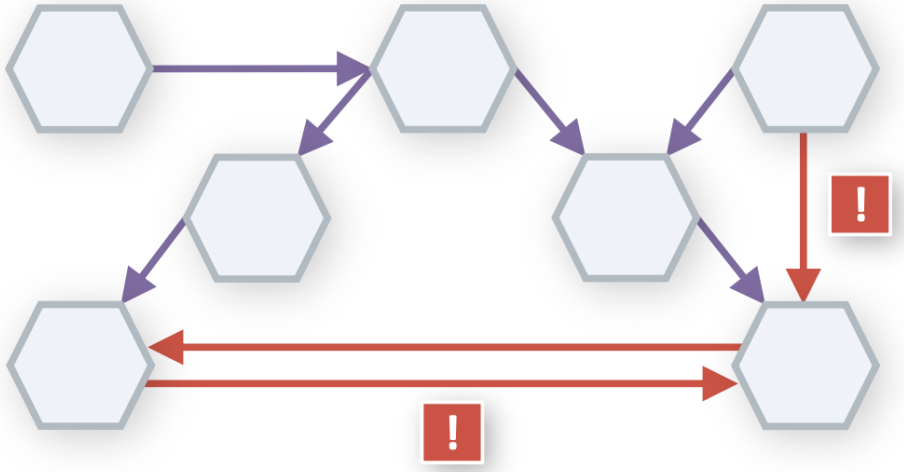
High entanglement indicates flaws in your functional decomposition

Example

Recommended practice



Inadvisable practice



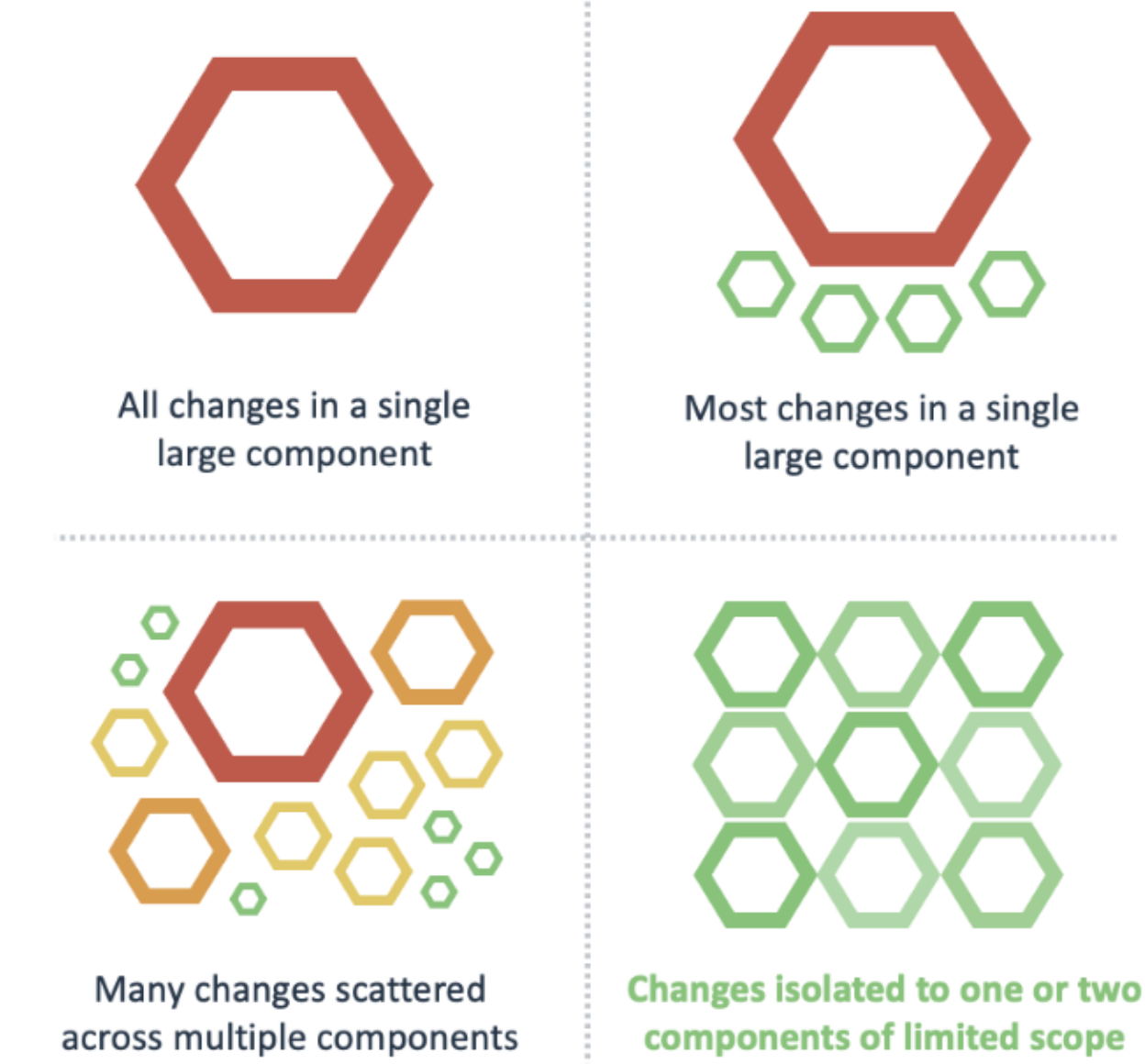
Architecture

Code breakdown

Components should be equally divided

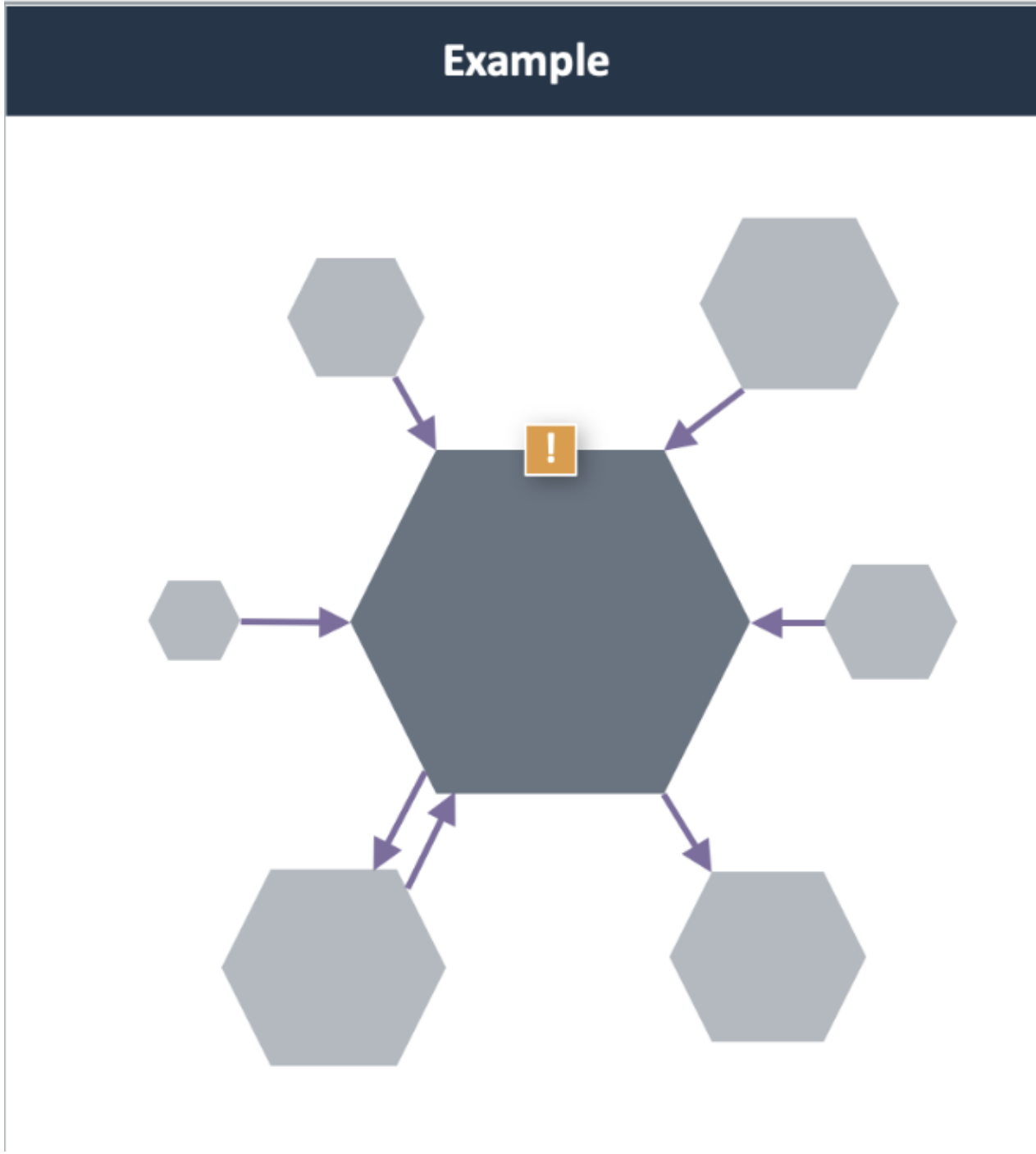
This will spread the need for modifications, make it easier to have several teams work on the same application and spread the knowledge.

Example



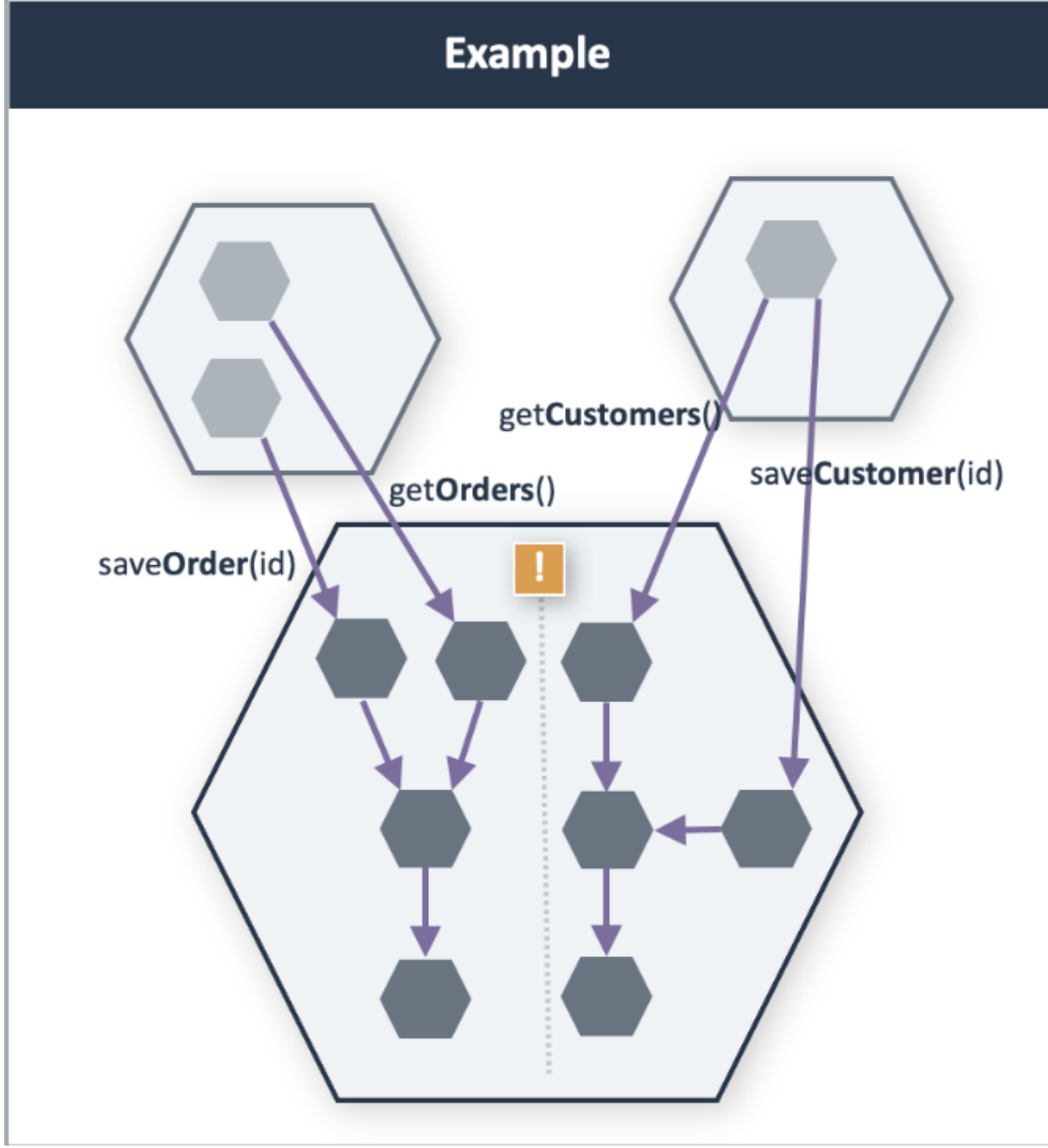
Component coupling

Sum of incoming and outgoing dependencies between components



Component cohesion

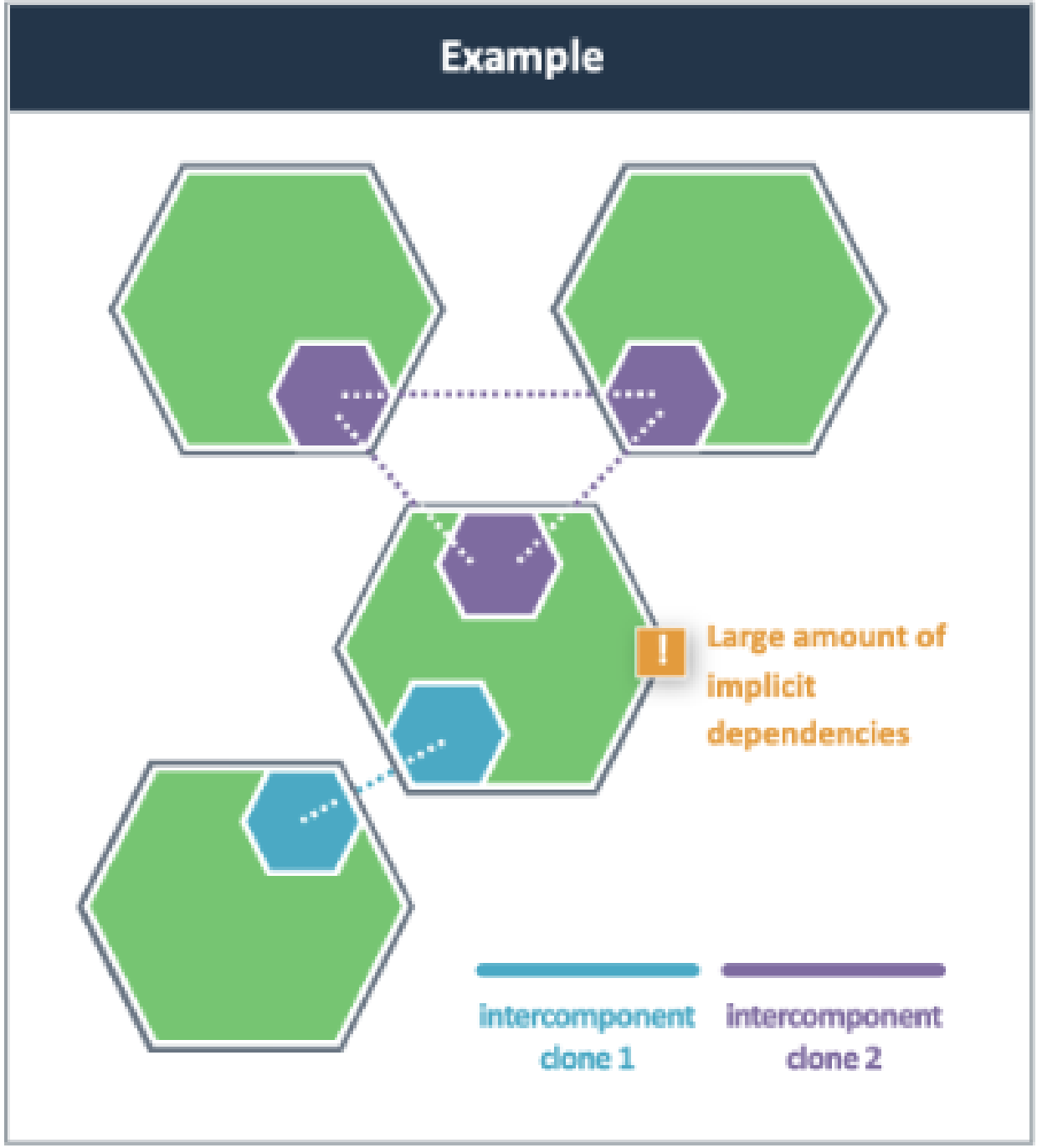
Ratio between the component's internal and external dependencies. Higher is better.



Code reuse

Shows duplication within and between components.

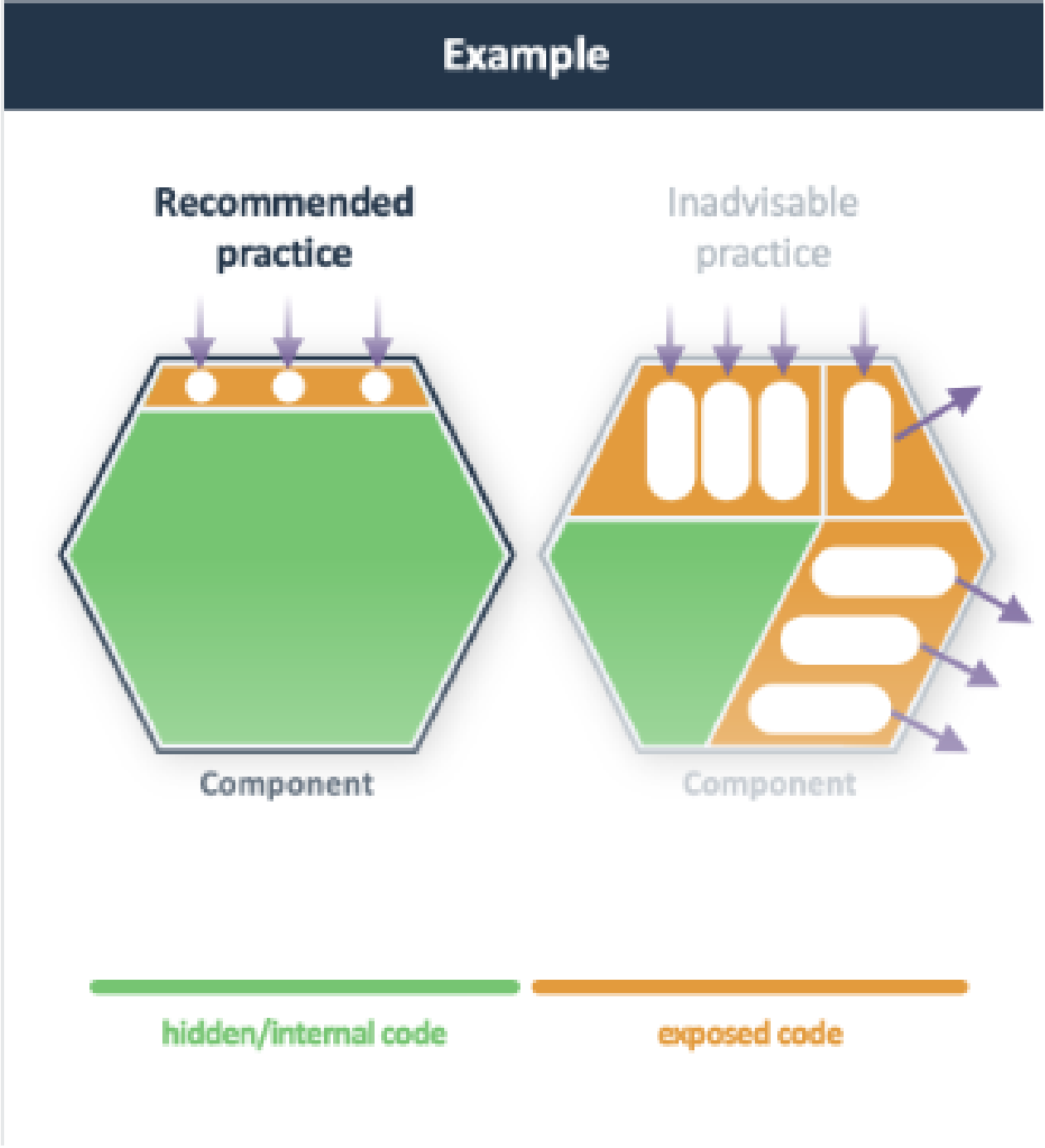
Adds up the number of lines that are duplicate in total.



Communication centralization

Percentage of code NOT involved in direct communication with other components.

Higher is better.

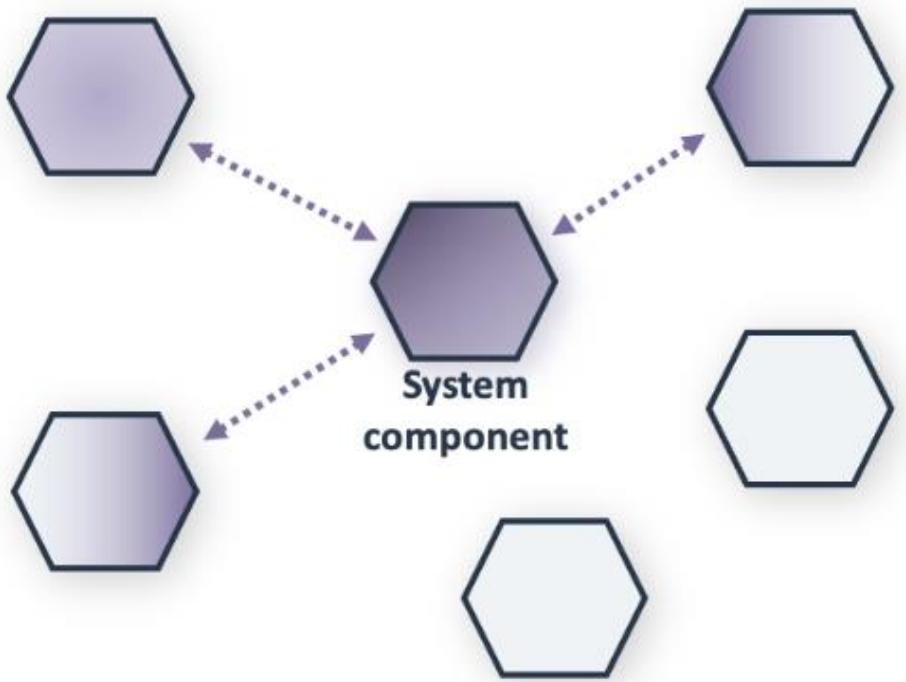


Bounded evolution

Measures the degree of co-evolution of components within a system based on the frequency of coupled code modifications over time.

Example

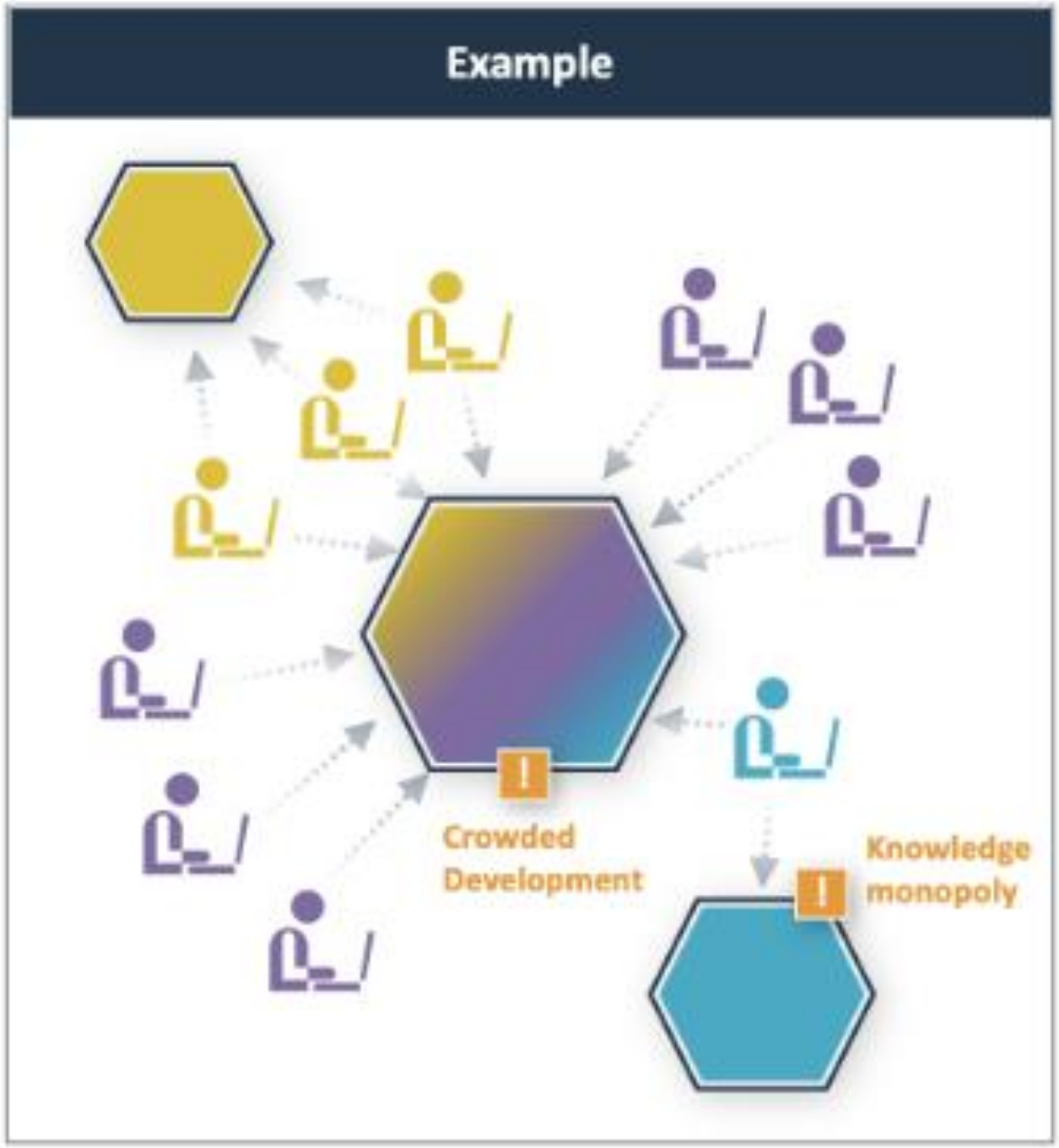
Change introduced to one component can have an effect on other components



Code Churn Percentage

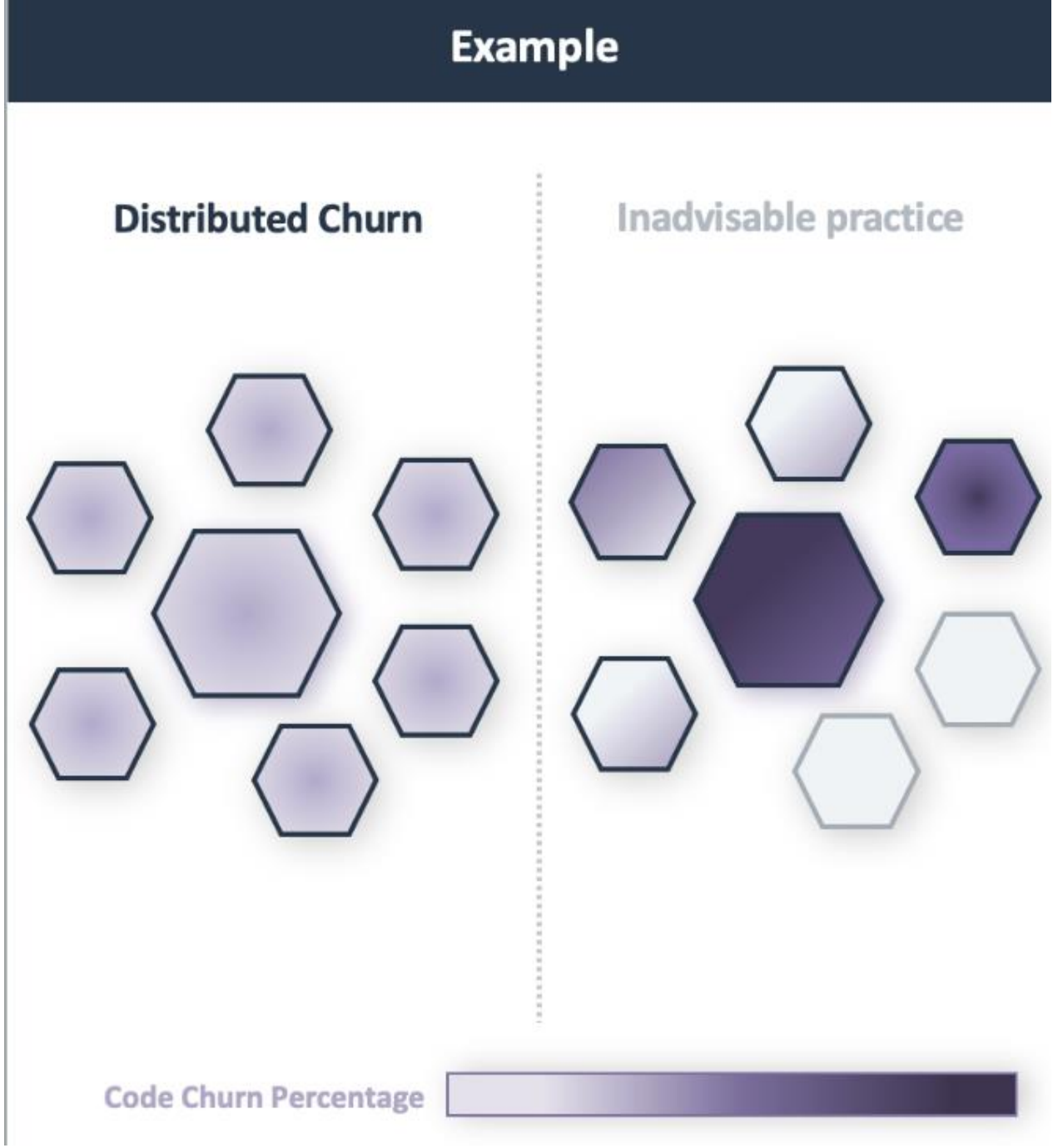
Knowledge distribution

Measures the degree to which development can grow and retain knowledge over a given system.



Component freshness

Measures the degree to which components are actively being kept up to date and maintained.



Security

Security

ISO 5055 - Security

CWE Top 25 Most Dangerous Software Weaknesses (2023)

OWASP Low-Code/No-Code Top 10 (2022)

PCI DSS v4.0 - Control Objectives (2022)

SIG Security

OWASP Top 10 (2021)

OWASP ASVS 4.0 - Sections

OWASP ASVS 4.0 - Chapters

OpenEdge specific rules

Type	Location	CVSS severity
Variables should be initialized with the NO-UNDO flag Weakness: Improper Resource Shutdown or Release Origin: SIG SAT Violations	.../Array.cls:212	0.0 I

```
79 for each Tenant share-lock by Tenant.Name:
80
81     if not can-find(first ApplicationUser where
82                           ApplicationUser.TenantId eq Tenant.TenantId and
83                           ApplicationUser.EmployeeId ne '') then
84         next.
85
```

A SHARE-LOCK is used instead of an EXCLUSIVE-LOCK

Done

Problem Description

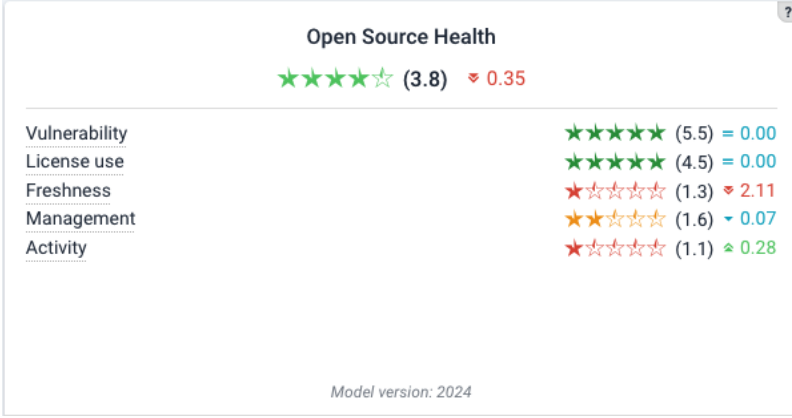
In concurrent programming, managing access to shared resources is critical to prevent race conditions and ensure data integrity. A SHARE-LOCK allows multiple threads or processes to read a shared resource but prevents them from writing to it simultaneously. However, if a situation requires exclusive access to a resource for updates or critical operations, an EXCLUSIVE-LOCK should be used instead. Using a SHARE-LOCK when an EXCLUSIVE-LOCK is needed can lead to deadlocks, where two or more operations are waiting indefinitely for each other to release locks.

Problem Example

```
1 FIND FIRST Customer SHARE-LOCK NO-ERROR.
2 IF AVAILABLE Customer THEN DO:
3     Customer.Balance = Customer.Balance + Invoice.Total.
4     UPDATE Customer.Balance.
5 END.
```

Open Source Health

Open Source Health



QSM scans your 3rd party libraries for

- Known vulnerabilities
- Freshness
- Activity
- Stability
- Management
- Legal licenses

proALPHA
Case Study

Better decisions.
For a better future.

Quality & Security Management

19.09.24

Unternehmen der proALPHA Gruppe

PROALPHA
BÖHME & WEIHS
Corporate Planning
curecomp
DIG
ENIT
EMPOLIS
GEDYS INTRANARE
humanIT Software GmbH
persis
SKILL SOFTWARE
tisoware

proALPHA ERP at a Glance

Reliable

>1,200
Employees

>30 years
On the market

3x
Stronger growth

Efficient

~2,000
Customers

39
Partners

€160,8m
Revenue in FY22/23

International

50
Countries

27
Branch offices

15
Localizations and language versions

proALPHA Codebase

ca. 9 Million Lines of ABL Code

All Generations of ABL Paradigms

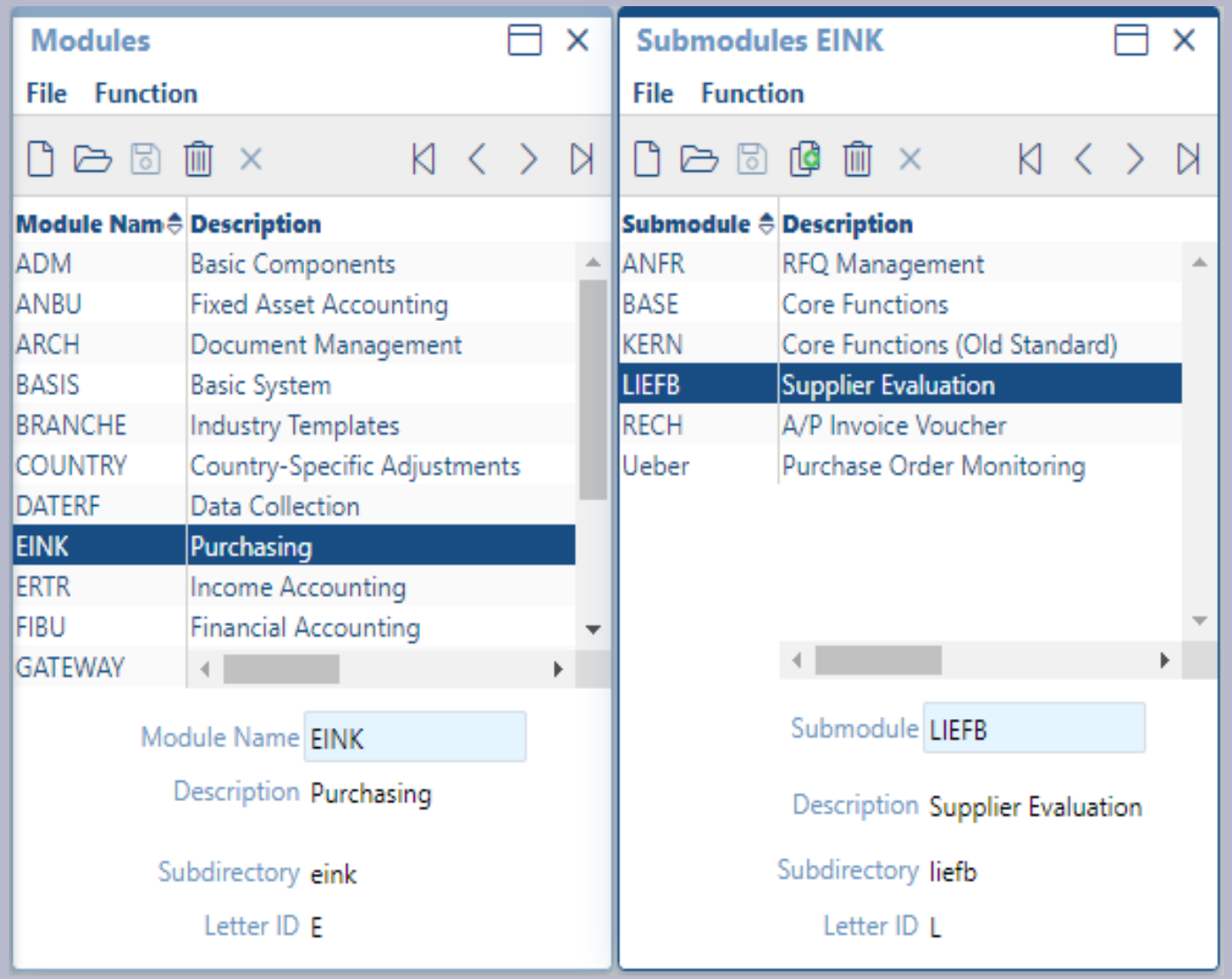
- Includes
- Super-Procedures
- OOABL

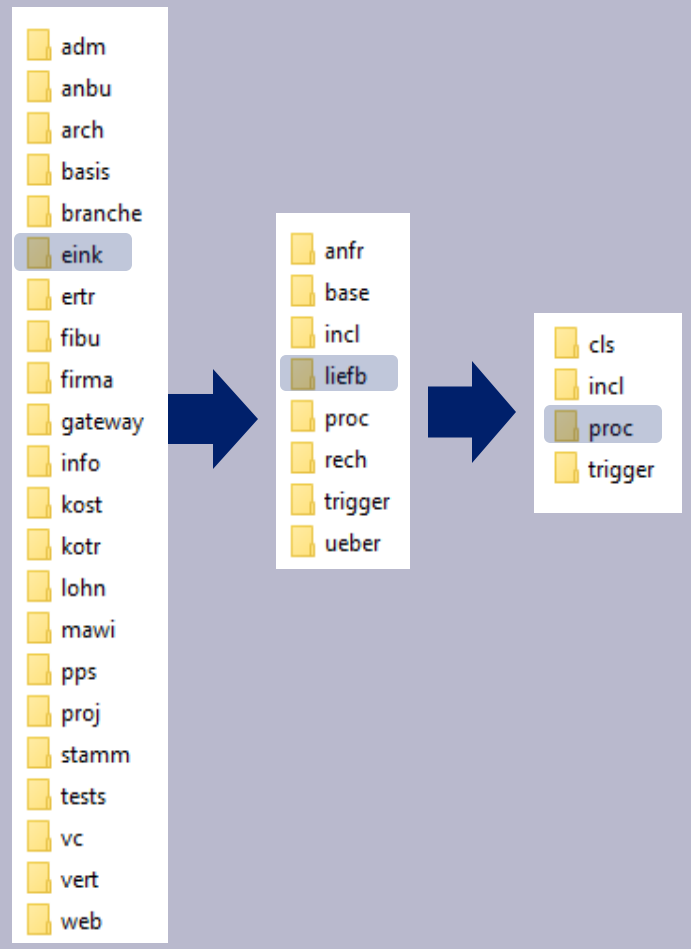
Code is well organized with a very strict folder and naming scheme

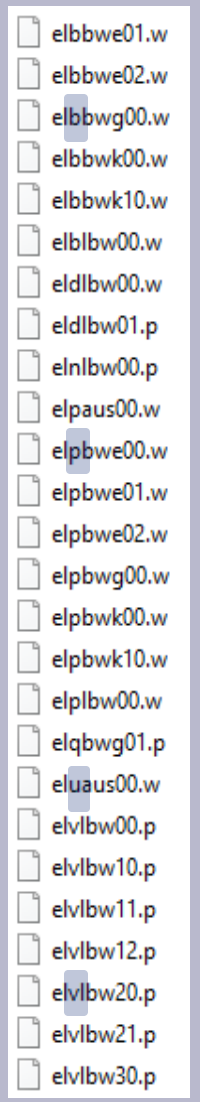
Nevertheless – Two Tier Architecture

- Business Logic in UI
- Business Logic in Triggers

proALPHA Codebase







proALPHA Codebase

- Code is well organized with a very strict folder and naming scheme

- First two letters are always module / submodule

- Third letter is type of program such as
  - b for browser
  - v for viewer
  - r for super procedure
  - and many more …

- Very strict formatting rules, checked automatically with static code checks …
  - Our code base looks the same for all the 9 million lines of code

proALPHA Codebase

- With this measures (and some more) we have very good control of our code base and can work on it with 200+ developers
- Nevertheless – we do have a noticeable increase of maintenance effort over the last couple of years
- We were looking for ways to lower this maintenance effort again

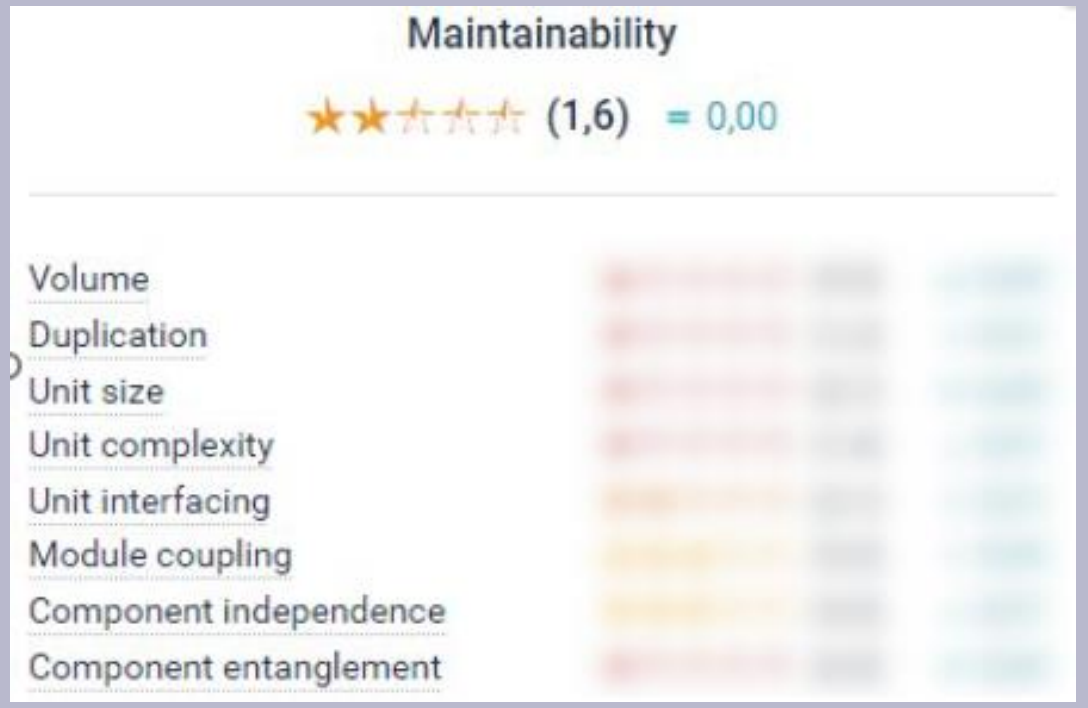
SIG comes into the picture!

proALPHA and SIG

- Progress asked us if we want to do a POC with SIG in December 2022
- The POC was planned for 6 month and ended in July 2023
- After the initial POC we did not immediately sign a contract
- The reason was SIG's handling of dependencies (details to follow)
- SIG corrected all our complaints, so we use SIG since April 2024

SIG Rating of our Codebase

- On a first glance this looks very bad
- Maintainability is NOT Quality!!
- Our strict code organisation is reflected in good numbers for Module Coupling and Component independence
- Duplication, Unit Size and Unit Complexity reflects more Progress „Specialities“ than absolute quality
- Does not change that improving this numbers should lower maintenance efforts!!



Evaluation of the key figures in context of OE / pA

- Volume
- our strict naming conventions are an adequate replacement for smaller repositories
- In the foreseeable future, we will not split the proALPHA code base into smaller repositories, so this number will not improve

Evaluation of the KPIs in context of OE / pA

- Duplications
- there is a lot of generated code, especially for the UI (Appbuilder) by using XFTRs
- This KPI does not show the correct value for the self-written business code

Description

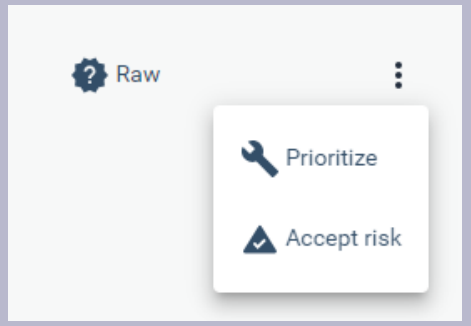
34 lines occurring 1.159 times in 1.159 files:
eublie01.w, ebbdea03.w, fabkpi01.w, d_bglo20.w, bebtxt00.w, sbbcbu12.w, s_bvar00.w, sbbgua01.w, vsbpos02.w, imbsti00.w, bjbstc00.w, s_bgro02.w, mbbakt07.w, drbcls00.w, ivbbra20.w, mlbort03.w, ivbaka02.w, mpbres10.w, fbbsit00.w, ebbqmt01.w, usbprc10.w, s_bbra00.w, s_bart04.w, sbbdup00.w, abbvor00.w, drbare01.w, fobbvb01.w, rbbsld03.w, dbbmen00.w, s_bbnk00.w, rabekb01.w, vsbsoc01.w, vsbcal04.w, vpbpla05.w, vcbkok00.w, vbboap01.w, d_blic00.w, vubrah06.w, bbbcti02.w, s_babii0.w, ebbqmt34.w, s_blie02.w, sbbmet11.w, sbbmet00.w, sbbdutn1.w, bubrec11.w, bobprt10.w, bubben01.w, mbbakt05.w, s_bsna00.w, ivbknz20.w, mmbpac22.w, e_bwe_03.w, bubfkt12.w, jbbpro04.w, vubart01.w, sbbdfl30.w, a_bidx00.w, iabspa00.w, e_bbel15.w,

```
620  &ANALYZE-SUSPEND _UIB-CODE-BLOCK _PROCEDURE adm-row-available B-table-Win  adm/support/proc/ds_rec00.p
621  PROCEDURE adm-row-available :
622  /* Description -----------------------------------------------------------*/
623  /*                                                                         */
624  /* Dispatched to this procedure when the Record-Source has a new row       */
625  /* available.  This procedure tries to get the new row (or foreign keys)   */
626  /* from the  Record-Source and process it.                                 */
627  /*                                                                         */
628  /* Parameters ------------------------------------------------------------*/
629  /*                                                                         */
630  /* <none>                                                                  */
631  /*                                                                         */
632  /*-------------------------------------------------------------------------*/
633
634  /*-------------------------------------------------------------------------*/
635  /* Processing                                                              */
636  /*-------------------------------------------------------------------------*/
637
638  /* Define variables needed by this internal procedure.                     */
639
640  {adm/template/incl/row-head.i}
641
642
643  /* Process the newly available records (i.e. display fields, open queries,    */
644  /* and/or pass records on to any RECORD-TARGETS).                          */
645
646  {adm/template/incl/row-end.i}
647
648  end procedure. /* adm-row-available */
```

Evaluation of the KPIs in context of OE / pA

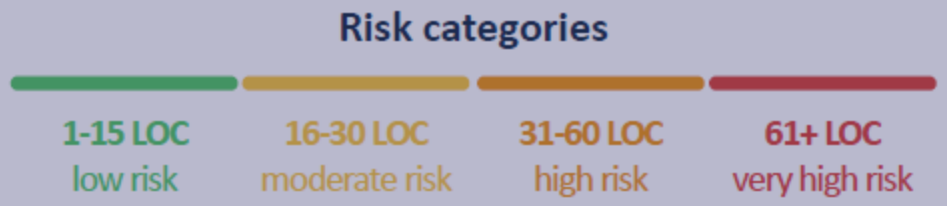
- Duplications
- there is a lot of generated code, especially for the UI (Appbuilder)
- This KPI does not show the correct value for the self-written business code

- You need to exclude these hits manually
- If any of the files are changed, the hit will reappear
- We are negotiating with SIG to change this



Evaluation of the KPIs in context of OE / pA

- Code Snippet has 32 lines
- Only this isolated snippet is already high risk
- What do you really do with something like that?




```
5333    for each bV_BelegPos
5334      where bV_BelegPos.Firma         = V_BelegKopf.Firma
5335        and bV_BelegPos.Belegart      = V_BelegKopf.Belegart
5336        and bV_BelegPos.ReferenzNr    = V_BelegKopf.ReferenzNr
5337        and bV_BelegPos.LfdNr_SR      = 0
5338        and bV_BelegPos.Satzart       = 'A':U
5339        and bV_BelegPos.Wertposition  = no
5340        and bV_BelegPos.Herk_Belegart = 'VUA':U
5341      use-index Main
5342      no-lock
5343      on error undo, throw:
5344
5345      if can-find(first VU_RA_Lieferung
5346                  where VU_RA_Lieferung.Firma          = bV_BelegPos.Firma
5347                    and VU_RA_Lieferung.BelegArt       = bV_BelegPos.Herk_BelegArt
5348                    and VU_RA_Lieferung.ReferenzNr     = bV_BelegPos.Herk_ReferenzNr
5349                    and VU_RA_Lieferung.PositionsNr    = bV_BelegPos.Herk_PositionsNr
5350                    and VU_RA_Lieferung.LieferscheinNr = V_BelegKopf.Belegnummer
5351                    and VU_RA_Lieferung.LieferscheinPos = bV_BelegPos.PositionsNr) then
5352
5353        run vert/auf/proc/vuvlie00.p (            bV_BelegPos.Firma,
5354                                                  bV_BelegPos.Herk_BelegArt,
5355                                                  bV_BelegPos.Herk_ReferenzNr,
5356                                                  bV_BelegPos.Herk_PositionsNr,
5357                                                  bV_BelegPos.ReferenzNr,
5358                                                  V_BelegKopf.Belegnummer,
5359                                                  V_BelegKopf.BelegDatum,
5360                                                  bV_BelegPos.PositionsNr,
5361                                                  ?,
5362                                                  '':U,
5363                                                  '':U,
5364                                                  0,
5365                                                  'ChangeDocDate':U,
5366                                        input-output iTempRA_Lieferung).
5367
5368    end. /* for each bV_BelegPos */
```

Evaluation of the KPIs in context of OE / pA

Implication

- If you have a method with 180 LOC and split it up in two methods with 90 LOC, you do not improve the rating!


```
5333    for each bV_BelegPos
5334      where bV_BelegPos.Firma         = V_BelegKopf.Firma
5335        and bV_BelegPos.Belegart      = V_BelegKopf.Belegart
5336        and bV_BelegPos.ReferenzNr    = V_BelegKopf.ReferenzNr
5337        and bV_BelegPos.LfdNr_SR      = 0
5338        and bV_BelegPos.Satzart       = 'A':U
5339        and bV_BelegPos.Wertposition  = no
5340        and bV_BelegPos.Herk_Belegart = 'VUA':U
5341      use-index Main
5342      no-lock
5343      on error undo, throw:
5344
5345      if can-find(first VU_RA_Lieferung
5346                  where VU_RA_Lieferung.Firma          = bV_BelegPos.Firma
5347                    and VU_RA_Lieferung.BelegArt       = bV_BelegPos.Herk_BelegArt
5348                    and VU_RA_Lieferung.ReferenzNr     = bV_BelegPos.Herk_ReferenzNr
5349                    and VU_RA_Lieferung.PositionsNr    = bV_BelegPos.Herk_PositionsNr
5350                    and VU_RA_Lieferung.LieferscheinNr = V_BelegKopf.Belegnummer
5351                    and VU_RA_Lieferung.LieferscheinPos = bV_BelegPos.PositionsNr) then
5352
5353        run vert/auf/proc/vuvlie00.p (            bV_BelegPos.Firma,
5354                                                  bV_BelegPos.Herk_BelegArt,
5355                                                  bV_BelegPos.Herk_ReferenzNr,
5356                                                  bV_BelegPos.Herk_PositionsNr,
5357                                                  bV_BelegPos.ReferenzNr,
5358                                                  V_BelegKopf.Belegnummer,
5359                                                  V_BelegKopf.BelegDatum,
5360                                                  bV_BelegPos.PositionsNr,
5361                                                  ?,
5362                                                  '':U,
5363                                                  '':U,
5364                                                  0,
5365                                                  'ChangeDocDate':U,
5366                                        input-output iTempRA_Lieferung).
5367
5368    end. /* for each bV_BelegPos */
```

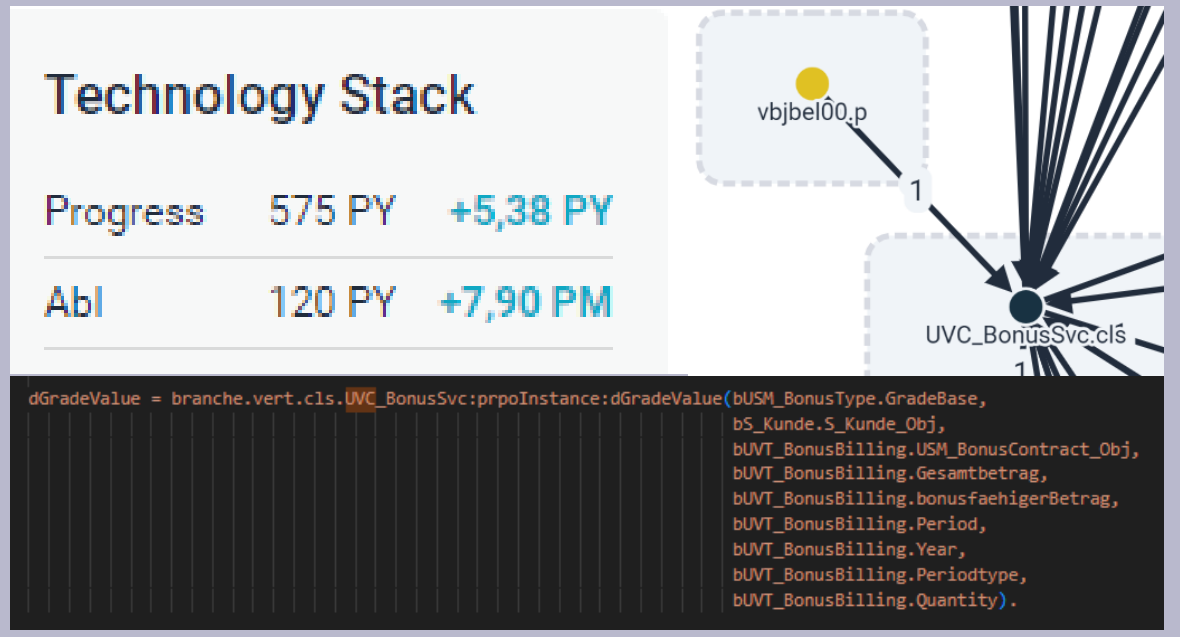
Evaluation of the KPIs in context of OE / pA

- OOABL was introduced 2004
- It is a design choice to pass the buffer or the primitive parameters
- pA mostly pass the parameters explicitly


```
5333    for each bV_BelegPos
5334      where bV_BelegPos.Firma         = V_BelegKopf.Firma
5335        and bV_BelegPos.Belegart      = V_BelegKopf.Belegart
5336        and bV_BelegPos.ReferenzNr    = V_BelegKopf.ReferenzNr
5337        and bV_BelegPos.LfdNr_SR      = 0
5338        and bV_BelegPos.Satzart       = 'A':U
5339        and bV_BelegPos.Wertposition  = no
5340        and bV_BelegPos.Herk_Belegart = 'VUA':U
5341      use-index Main
5342      no-lock
5343      on error undo, throw:
5344
5345      if can-find(first VU_RA_Lieferung
5346                  where VU_RA_Lieferung.Firma          = bV_BelegPos.Firma
5347                    and VU_RA_Lieferung.BelegArt       = bV_BelegPos.Herk_BelegArt
5348                    and VU_RA_Lieferung.ReferenzNr     = bV_BelegPos.Herk_ReferenzNr
5349                    and VU_RA_Lieferung.PositionsNr    = bV_BelegPos.Herk_PositionsNr
5350                    and VU_RA_Lieferung.LieferscheinNr = V_BelegKopf.Belegnummer
5351                    and VU_RA_Lieferung.LieferscheinPos = bV_BelegPos.PositionsNr) then
5352
5353        run vert/auf/proc/vuvlie00.p (            bV_BelegPos.Firma,
5354                                                  bV_BelegPos.Herk_BelegArt,
5355                                                  bV_BelegPos.Herk_ReferenzNr,
5356                                                  bV_BelegPos.Herk_PositionsNr,
5357                                                  bV_BelegPos.ReferenzNr,
5358                                                  V_BelegKopf.Belegnummer,
5359                                                  V_BelegKopf.BelegDatum,
5360                                                  bV_BelegPos.PositionsNr,
5361                                                  ?,
5362                                                  '':U,
5363                                                  '':U,
5364                                                  0,
5365                                                  'ChangeDocDate':U,
5366                                        input-output iTempRA_Lieferung).
5367
5368    end. /* for each bV_BelegPos */
```

Dependency Specialties

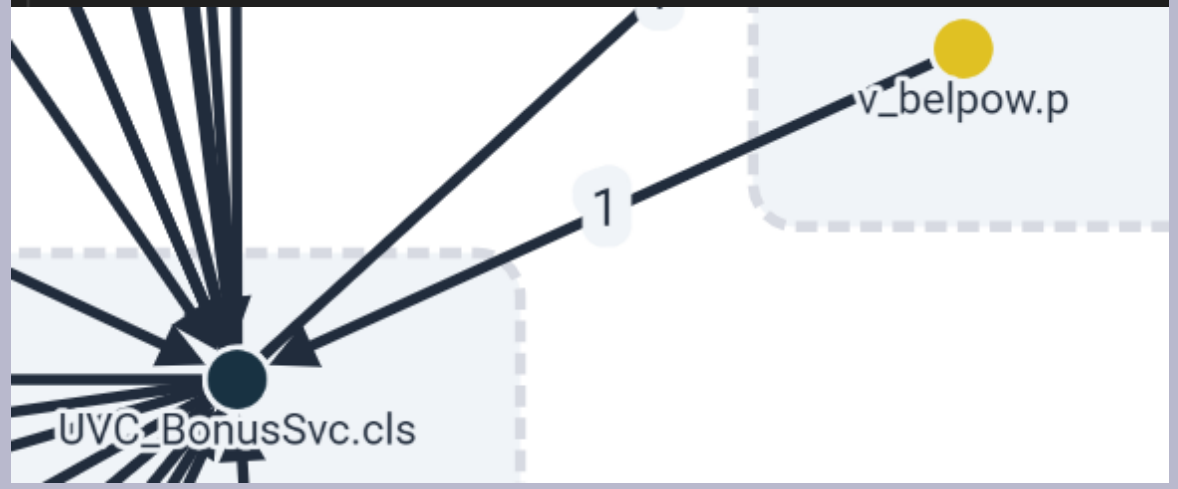
- SIG handles OOABL and Progress as separate technologies
- Normally there can't be dependencies between codebases of different technologies
- SIG corrected this for OpenEdge in cooperation with proALPHA



Dependency Specialties

- Database Triggers are a very special construct
- Whenever a record is created or changed a database trigger fires
- This creates a dependency from the class or program to the trigger
- SIG implemented this in cooperation with proALPHA

```
create bV_BelegPos.
assign
  bV_BelegPos.Firma           = bV_BelegKopf.Firma
  bV_BelegPos.BelegArt        = bV_BelegKopf.BelegArt
  bV_BelegPos.ReferenzNr      = bV_BelegKopf.ReferenzNr
  bV_BelegPos.lfdNr_SR        = 0
  bV_BelegPos.PositionsNr     = dLastPos + 0.1
  bV_BelegPos.Satzart         = 'T':U
  .
validate bV_BelegPos.
```



proALPHA Strategy for the near future

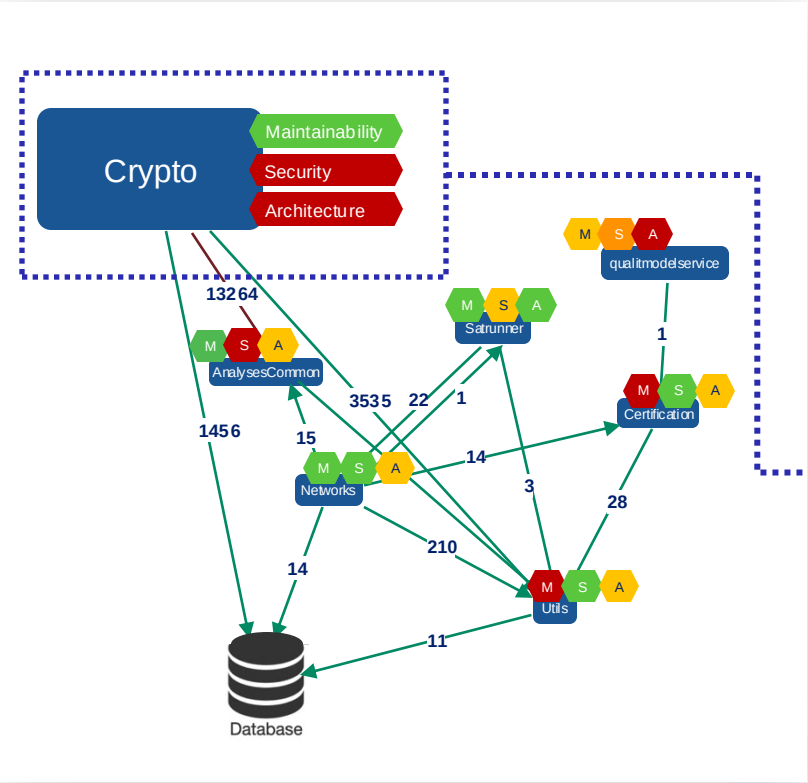
- Work on Duplications
- if we refactor other aspects first (for example complexity), we remove literal code duplications but the semantic duplication stays and is not detected anymore
- Work on cyclic dependencies
- Cyclic dependencies prevent us from splitting the code base into smaller pieces

Improvement on Duplications

DEMO


End of Case Study

Smart application optimization plan

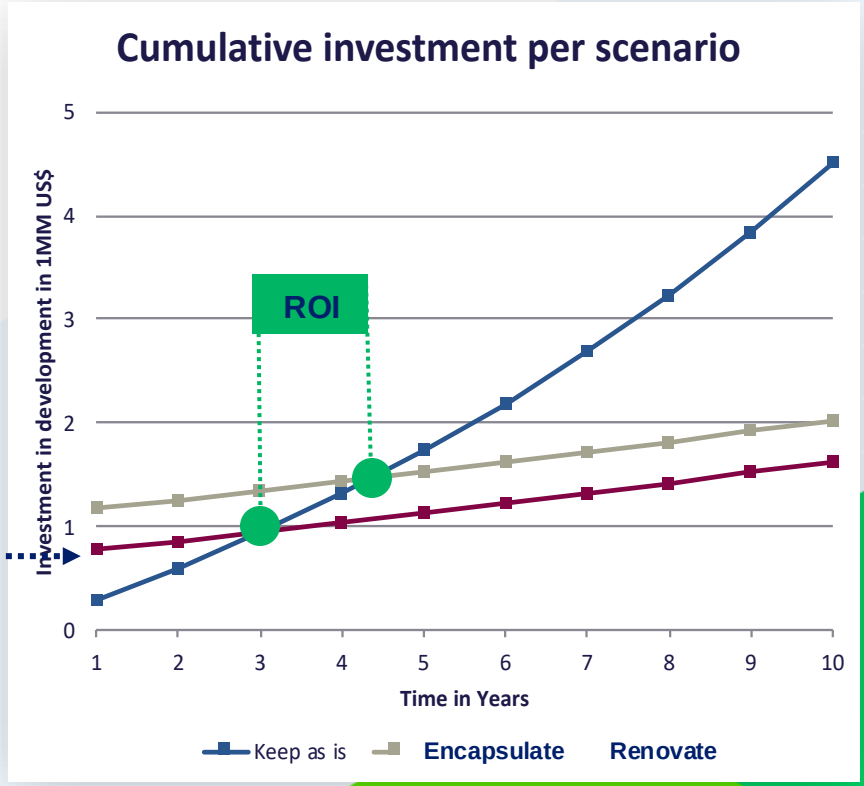


IDENTIFICATION BY OPENEDGE QSM

Hotspot optimization options

- Keep as is
- Rebuild
- Renovate
- Rearchitect
- Encapsulate

EVALUATION



ANALYSIS

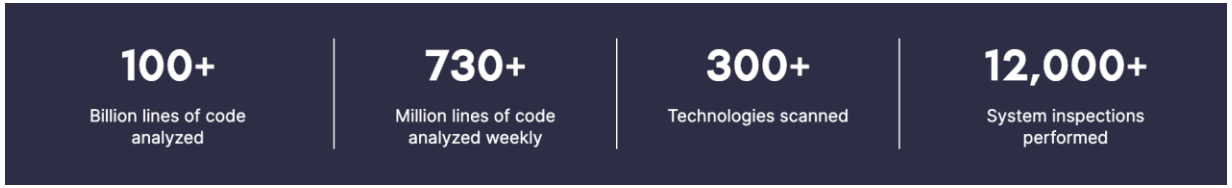
Quality & Security Management: What's Included

License to use Analysis tool (Sigrid)

- Data-driven software intelligence platform
- Analyses Progress source code
- Derive holistic insights into
  - Risks, costs, opportunities
  - On multiple software quality aspects

Consultancy to run the assessment

Fully documented assessment results



Leveraging Sigrid® Platform
Analyzing source code quality and security – multi-metrics, multi-roles, and risk-based prioritized

Largest software benchmark in the world 1,000,000+ Inspections

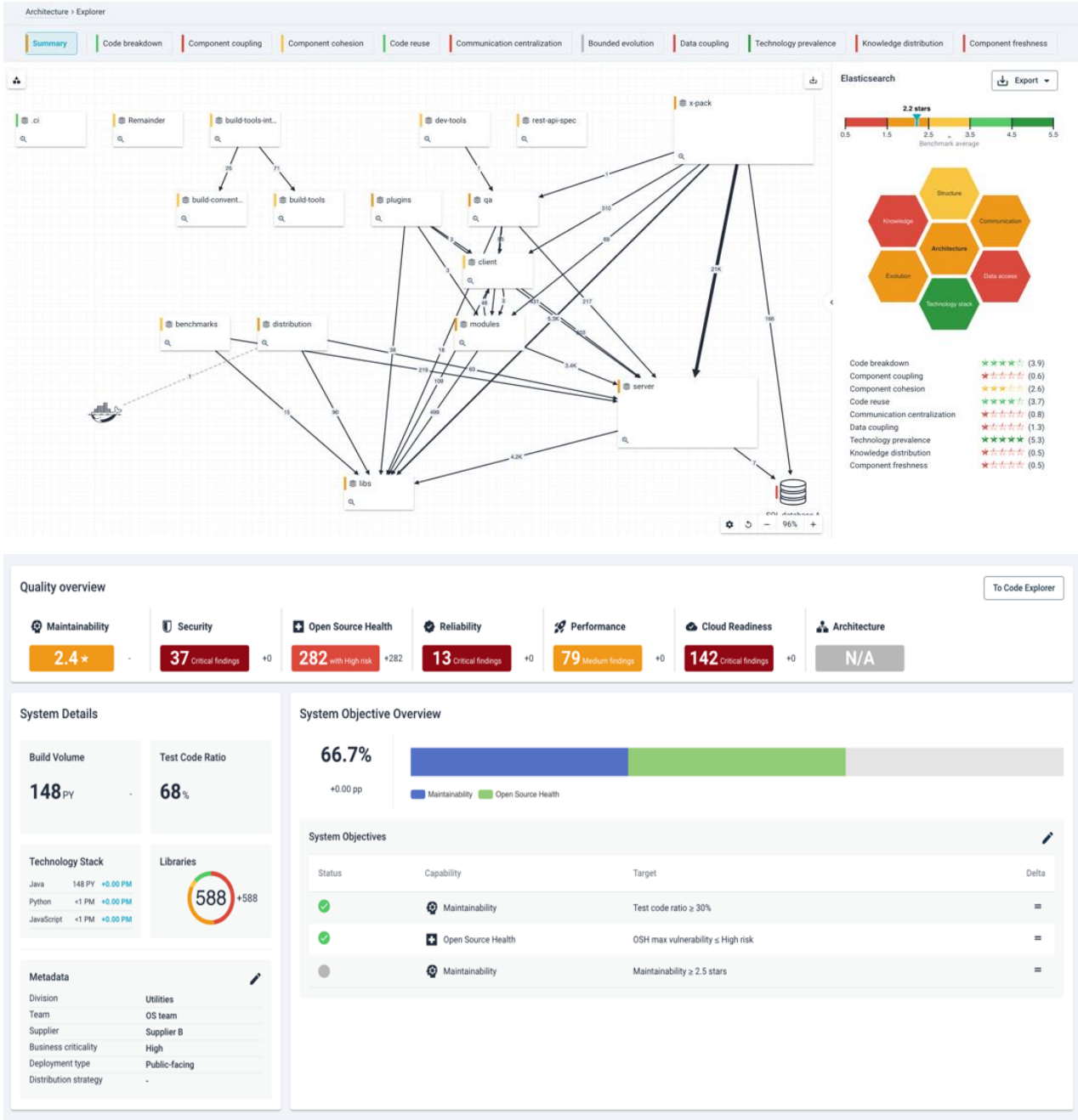
Why Application Quality and Security Management?

Create awareness and insights on current application state

Justify why **modernization** is needed and where and how it will pay off most

Get fact-based insights to drive innovation, manage risks and lower costs

Increase transparency from the **development team** to **boardroom** to set priorities for your software development investments



Progress® OpenEdge®

News You Can Use


Progress®