

Software-Metriken und Refactoring

Thomas Haug
MATHEMA Software GmbH
209

Wer bin ich ...

- > Dipl.-Inf (Univ.)
- > Senior Consultant, Architekt und Trainer (MATHEMA Software GmbH)
- > 25+ Jahre Software Entwicklung
- > 12+ Jahre Java Enterprise
- > 7+ Jahre .Net

- > Schwerpunkte
 - Software Architektur
 - Verteilte Systeme
 - Objekt-Relationales Mapping

E-Mail: thomas.haug@mathema.de



AGENDA

- > Motivation
- > Metriken
- > Metriken kombinieren
- > Zusammenfassung, Literatur und Werkzeuge

AGENDA

> Motivation

> Metriken

> Metriken kombinieren

> Zusammenfassung, Literatur und Werkzeuge

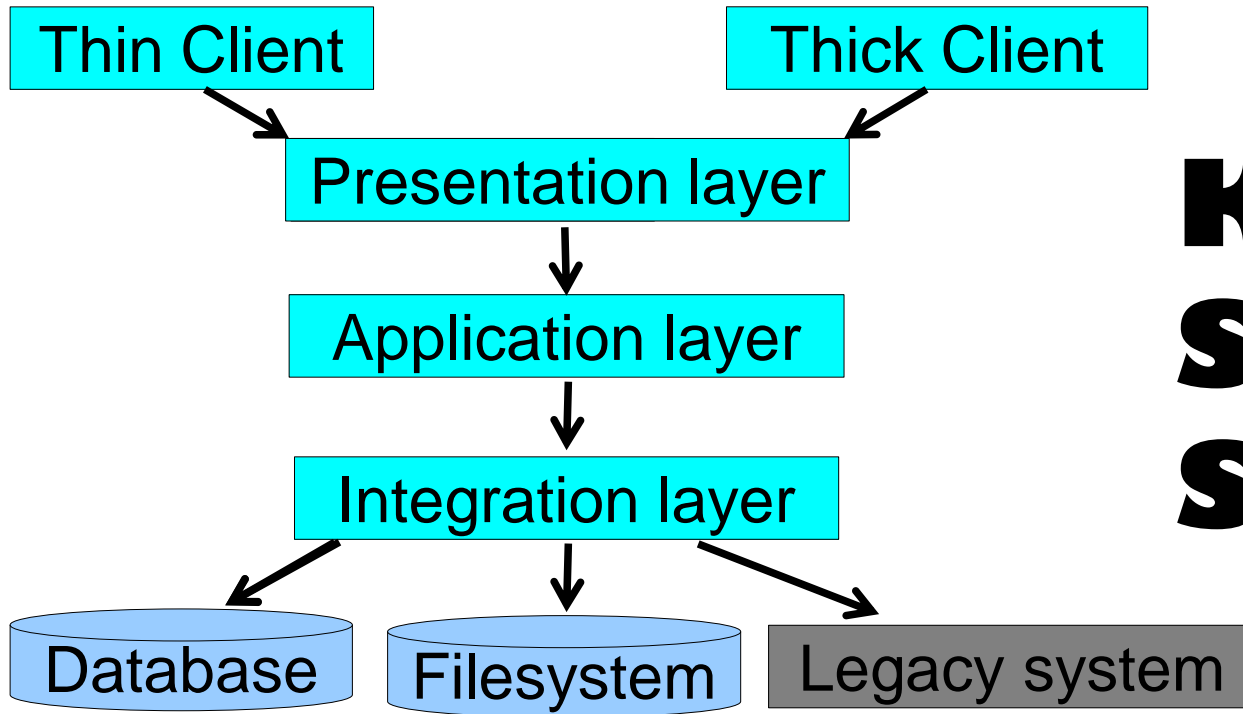
Motivation – Der Wartungsalptraum

- > 80% der Kosten stecken in der Wartung
- > Softwarewartung ist „die Veränderung eines Softwareprodukts nach dessen Auslieferung, um Fehler zu beheben, Performanz oder andere Attribute zu verbessern oder Anpassungen an die veränderte Umgebung vorzunehmen.“
(Definition gemäß IEEE 610.12-1990)

Year	Proportion of software maintenance costs	Definition	Reference
2000	>90%	Software cost devoted to system maintenance & evolution / total software costs	Erlikh (2000)
1993	75%	Software maintenance / information system budget (in Fortune 1000 companies)	Eastwood (1993)
1990	>90%	Software cost devoted to system maintenance & evolution / total software costs	Moad (1990)
1990	60-70%	Software maintenance / total management information systems (MIS) operating budgets	Huff (1990)
1988	60-70%	Software maintenance / total management information systems (MIS) operating budgets	Port (1988)
1984	65-75%	Effort spent on software maintenance / total available software engineering effort.	McKee (1984)
1981	>50%	Staff time spent on maintenance / total time (in 487 organizations)	Lientz & Swanson (1981)
1979	67%	Maintenance costs / total software costs	Zelkowitz <i>et al.</i> (1979)

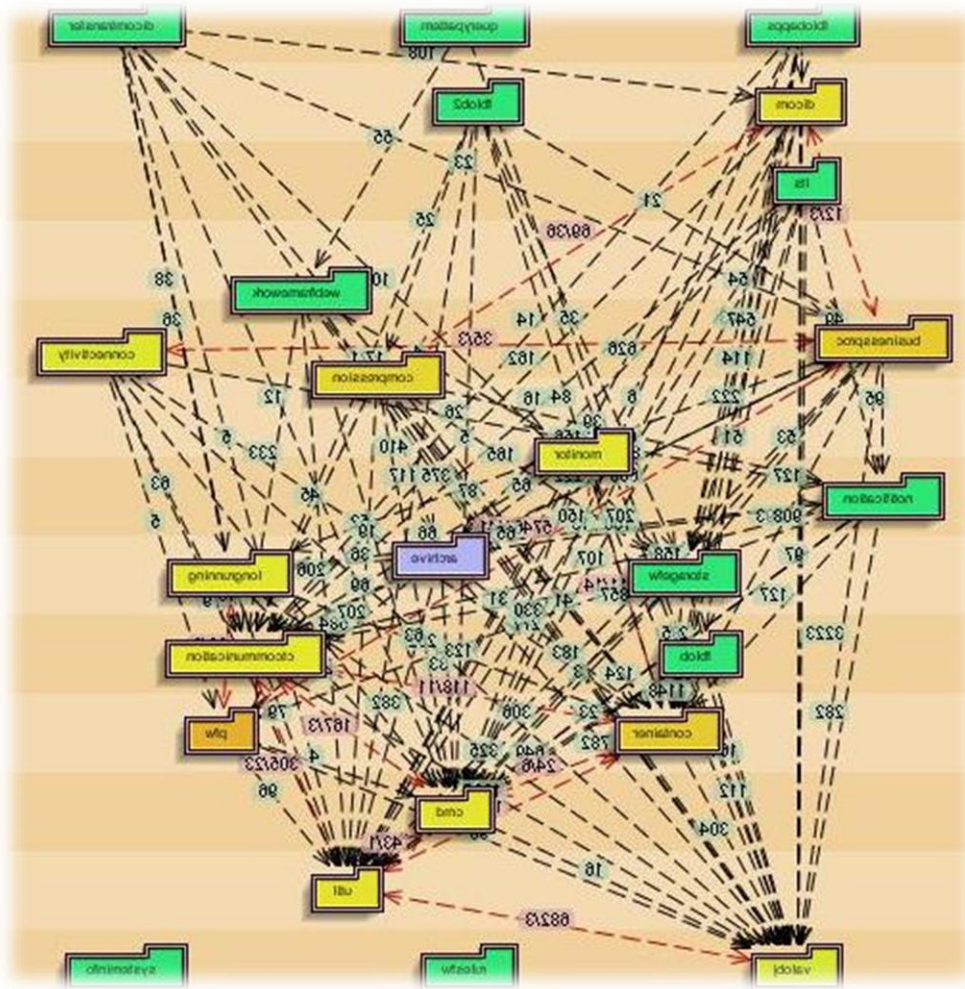
<http://users.jyu.fi/~koskinen/smcosts.htm>

Motivation – Der Widerspruch



**KEEP IT
SIMPLE
STUPID.**

Motivation - Warum



© by AskDaveTaylor at flickr

Was kann gegen den Verfall der Software getan werden?



© by D'Arcy Norman at flickr

A close-up photograph of a complex mechanical assembly, likely a part of a machine or engine. The assembly is made of dark metal and features several bolts, nuts, and a large, ribbed cylindrical component. A thin, light-colored measuring tool, possibly a feeler gauge, is inserted into a gap between two parts of the assembly. The background is blurred, focusing attention on the mechanical details.

Messen
und
Kontrollieren!

Motivation – Measure and Check

- > Prozesse
 - Test Driven Development
 - Iteratives und inkrementelles Vorgehen
- > Laufzeit
 - Test Coverage
 - Memory Analysis
 - Deadlock Detection
 - Monitoring / Logging
- > Statische Codeanalyse
 - Design und Code guidelines
 - Design- and Code Reviews
 - Statische Analyse Werkzeuge
 - **Metriken**



AGENDA

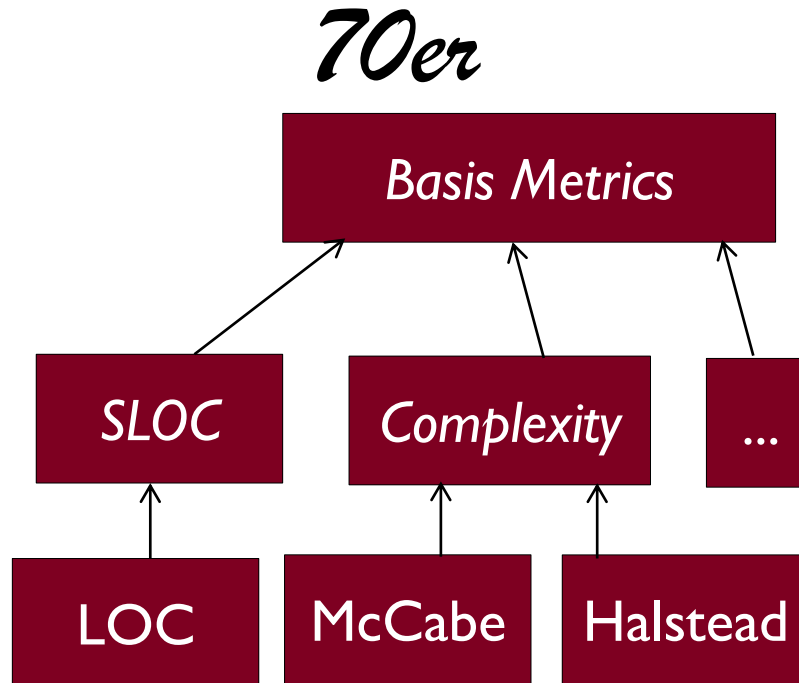
> Motivation

> **Metriken**

> Metriken kombinieren

> Zusammenfassung, Literatur und Werkzeuge

Klassifikation der Metriken



Lines of Code

Source Lines of Code (SLOC)

- > Physikalische LOC
 - Codezeilen, Kommentare, etc. zählen mit

- > Logical lines of code (ILOC) and Non Commented Source Statements (NCSS)

- Anweisungen abzählen

“Measuring programming progress by lines of code is like measuring aircraft building progress by weight”

Bill Gates

Lines of Code

> LOC vs ILOC

```
for (int i = 0; i < 10; i++)  
    System.out.println(i + ". value");
```

LOC = 2

ILOC = 2

```
for (int i = 0; i < 10; i++)  
{  
    // print to console  
    System.out.println(i + ". value");  
}
```

LOC = 5

ILOC = 2

Lines of Code - Tomcat 6.0.20

Name	LOCC	WMC
StandardContext	5611	661
ELParser	2106	
ELParserTokenManager	1334	
Parser	1747	
GenerateVisitor	2532	
Request	2583	
WebappClassLoader	2281	
WebdavServlet	2689	
Http11NioProcessor	1775	
XMLEncodingDetector	1595	
Http11AprProcessor	1739	
ValidateVisitor	1236	
DefaultServlet	2107	
ManagerServlet	1515	
Digester	2830	
DeltaManager	1474	
AbstractReplicatedMap	1422	
JspC	1333	
Http11Processor	1654	
MBeanUtils	1809	
RealmBase	1386	
IntrospectionUtils	992	
Mapper	1404	
JspUtil	1126	

JavaNCSS:

File Help

Packages	Classes	Methods	
193 149	7	0	7 org.apache.catalina.core.ApplicationFilterFactory
194 322	36	1	38 org.apache.catalina.core.ApplicationHttpRequest
195 68	23	0	24 org.apache.catalina.core.ApplicationHttpResponse
196 37	7	0	8 org.apache.catalina.core.ApplicationRequest
197 29	10	0	11 org.apache.catalina.core.ApplicationResponse
198 100	8	0	2 org.apache.catalina.core.AprLifecycleListener
199 7	0	0	0 org.apache.catalina.core.Constants
200 525	68	2	54 org.apache.catalina.core.ContainerBase
201 210	118	1	1 org.apache.catalina.core.DummyRequest
202 101	68	0	1 org.apache.catalina.core.DummyResponse
203 8	1	0	2 org.apache.catalina.core.JasperListener
204 343	26	0	27 org.apache.catalina.core.NamingContextListener
205 1926	264	0	227 org.apache.catalina.core.StandardContext
206 67	5	0	6 org.apache.catalina.core.StandardContextValve
207 173	27	0	14 org.apache.catalina.core.StandardEngine
208 13	3	0	4 org.apache.catalina.core.StandardEngineValve
209 249	41	0	37 org.apache.catalina.core.StandardHost
210 126	7	0	8 org.apache.catalina.core.StandardHostValve
211 176	19	0	16 org.apache.catalina.core.StandardPipeline
212 250	37	0	30 org.apache.catalina.core.StandardServer
213 234	33	0	24 org.apache.catalina.core.StandardService
214 123	27	2	3 org.apache.catalina.core.StandardThreadExecutor

Komplexität

> Cyclomatic Complexity (CC(N))

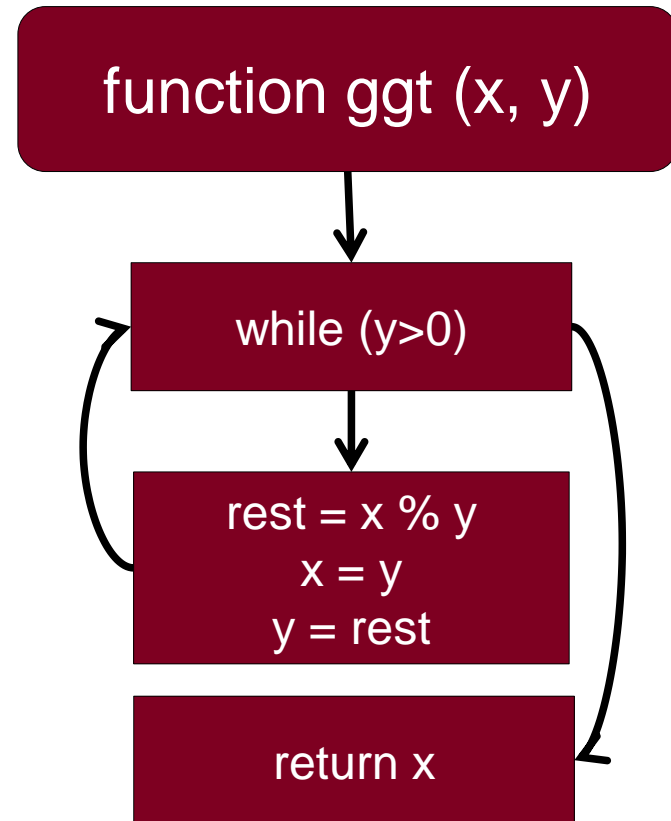
> McCabe, 1976

p: number of components

e: number of edges

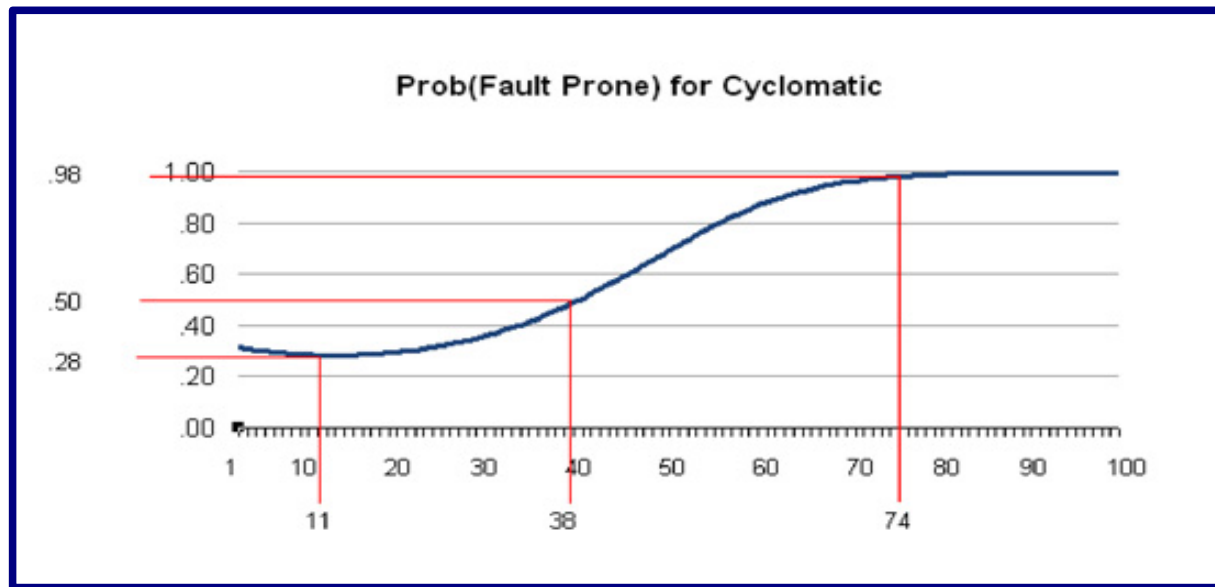
n: number of nodes

> Beispiel $CC = 4 - 4 + 2*1 = 2$



Komplexität

Original McCabe values	PMD (per method)
1-10 : low complexity	1-4 : low complexity
11-20 : medium complexity	5-7 : medium complexity
21-50 : high complexity	8-10 : high complexity
> 51 : very high complexity	> 11 : very high complexity



<http://www.enerjy.com/blog/?p=198>

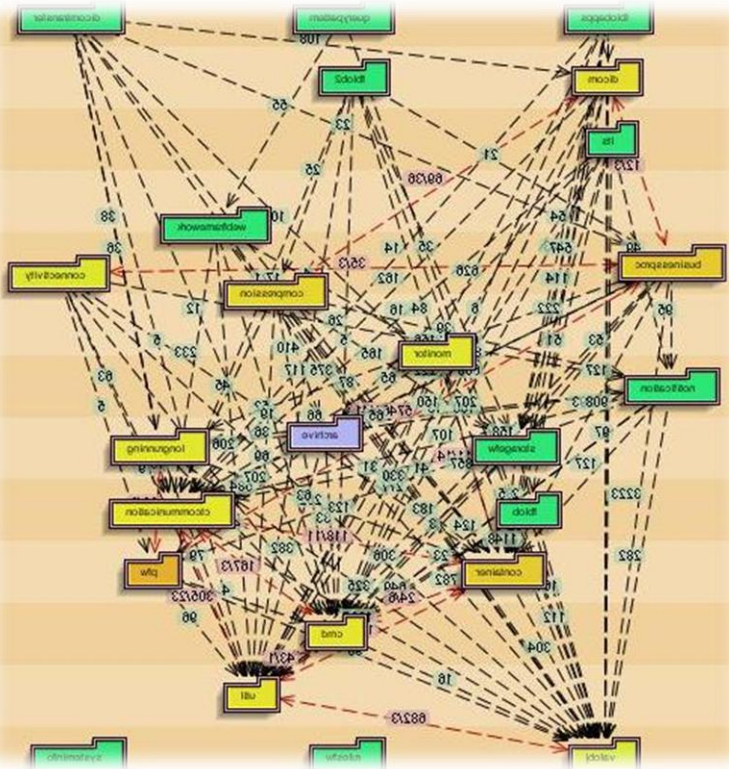
Komplexität - Beispiel Spring 2.5.6

method: CYCLO > 10.0 filter on (method group of system spring-framework-2.5) [165]

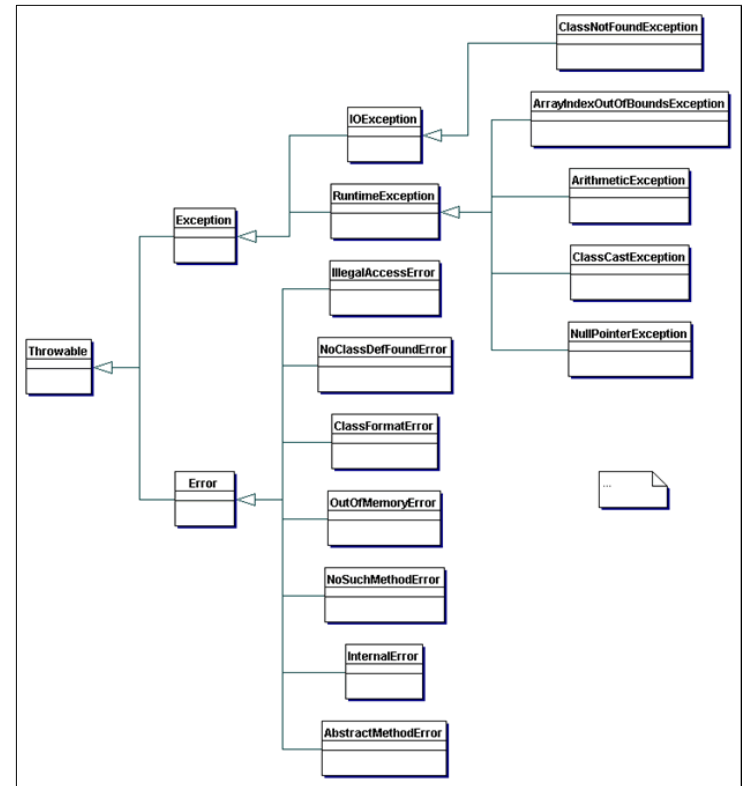
Name	scope	CYCLO
buildSessionFactory	Class: org.springframework.orm.hibernate3.LocalSessionFactoryBean - statute: NOR...	46
resolveArguments	Class: org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandlerAda...	36
doMatch	Class: org.springframework.util.AntPathMatcher - statute: NORMAL - location: FILE:C:\...	36
doTestReceive	Class: org.springframework.jms.core.JmsTemplate102Tests - statute: NORMAL - loc...	35
resolveArguments	Class: org.springframework.web.portlet.mvc.annotation.AnnotationMethodHandlerAda...	35
matchStrings	Class: org.springframework.util.AntPathMatcher - statute: NORMAL - location: FILE:C:\...	35
testActualPathMatching	Class: org.springframework.web.servlet.handler.PathMatchingUrlHandlerMappingTes...	33
reconcileParameters	Class: org.springframework.jdbc.core.metadata.CallMetaDataContext - statute: NORM...	33
setParameterValueInternal	Class: org.springframework.jdbc.core.StatementCreatorUtils - statute: NORMAL - locat...	31
convertIfNecessary	Class: org.springframework.beans.TypeConverterDelegate - statute: NORMAL - locati...	31
resolveHandlerMethod	Class: org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandlerAda...	30
doMapRow	Class: org.springframework.jdbc.core.AbstractBeanPropertyRowMapper - statute: NO...	30
resolveDependency	Class: org.springframework.beans.factory.support.AbstractAutowireCapableBeanFact...	30
translate	Class: org.springframework.jdbc.support.SQLExceptionTranslator - sta...	29
createCustomException	Class: org.springframework.jdbc.support.SQLExceptionTranslator - sta...	29
getBeanNamesForType	Class: org.springframework.beans.factory.support.DefaultListableBeanFactory - stat...	29
setPropertyValue	Class: org.springframework.beans.BeanWrapperImpl - statute: NORMAL - location: FI...	27
accept	Class: org.springframework.aop.framework.Cglib2AopProxy\$ProxyCallbackFilter - stat...	26
doTestReceive	Class: org.springframework.jms.core.JmsTemplateTests - statute: NORMAL - location...	25
instantiateUsingFactoryMethod	Class: org.springframework.beans.factory.support.ConstructorResolver - statute: NOR...	25
parseMapElement	Class: org.springframework.beans.factory.xml.BeanDefinitionParserDelegate - statute...	25
nullSafeEquals	Class: org.springframework.util.ObjectUtils - statute: NORMAL - location: FILE:C:\devj...	25
autowireConstructor	Class: org.springframework.beans.factory.support.ConstructorResolver - statute: NOR...	24
doBegin	Class: org.springframework.orm.hibernate3.HibernateTransactionManager - statute: ...	23
getColumnValue	Class: org.springframework.jdbc.core.SingleColumnRowMapper - statute: NORMAL - l...	23
invoke	Class: org.springframework.jdbc.datasource.LazyConnectionDataSourceProxy\$LazyC...	23
parseBeanDefinitionElement	Class: org.springframework.beans.factory.xml.BeanDefinitionParserDelegate - statute...	23
testAllMacros	Class: org.springframework.web.servlet.view.freemarker.FreeMarkerMacroTests - stat...	22
doHandle	Class: org.springframework.web.portlet.mvc.annotation.AnnotationMethodHandlerAda...	22

CC > 11 : very high complexity

Einen Moment...



? Kopplung ?



? Vererbung ?

Metriken in der Objekt-Orientierung

- > Klassischen Metriken ziehen wichtige Aspekte der Objekt-Orientierung nicht in Betracht.
- > OO Metriken bewerten die Dimensionen
 - Größe und Komplexität
 - Vererbung
 - Kopplung
- > Sehr viele (≥ 375)
- > Beispiele
 - Chidamber und Kemerer (CK) Metric suite (1994)
WMC, CBO, DIT, NOC, RFC, LCOM
 - Robert C. Martin Metrics (1994)
Coupling, Stability, Abstractness,...

Object-Oriented Metrics - CK Beispiele

> Weighted Method Count (WMC)

()

()

class: WMC > 51.0 filter on (class group of system spring-framework-2.5.6.SEC01) [83]

Name	method.sum_CYCLO	AMW
BeanDefinitionParserDelegate	215	4,57
AbstractBeanFactory	211	2,89
AbstractAutowireCapableBeanFactory	204	4,08
StringUtils	175	3,12
JdbcTemplate	161	1,85

⋮

class: WMC > 51.0 filter on (class group of system spring-framework-2.5.6.SEC01) [83]

Name	method.sum_CYCLO	AMW
AntPathMatcher	88	11
JdbcUtils	84	7
ConstructorResolver	84	14
Mock		

Sehr hohe Komplexität der Klassen, aber die durchschnittliche Komplexität ist niedrig.

Sehr hohe Klassenkomplexität und die durchschnittliche Komplexität hoch bzw. sehr hoch

Object-Oriented Metrics - CK Beispiele

> Coupling Between Object Classes (CBO)

()

- Heuristik: Werte über 14 vermeiden (Sahraoui, Godin, Miceli: „*Can Metrics Help Bridging the Gap Between the Improvement of OO Design Quality and Its Automation?*“)

class: CBO > 14.0 filter on (class group of system spring-framework-2.5.6.SEC01) [37]

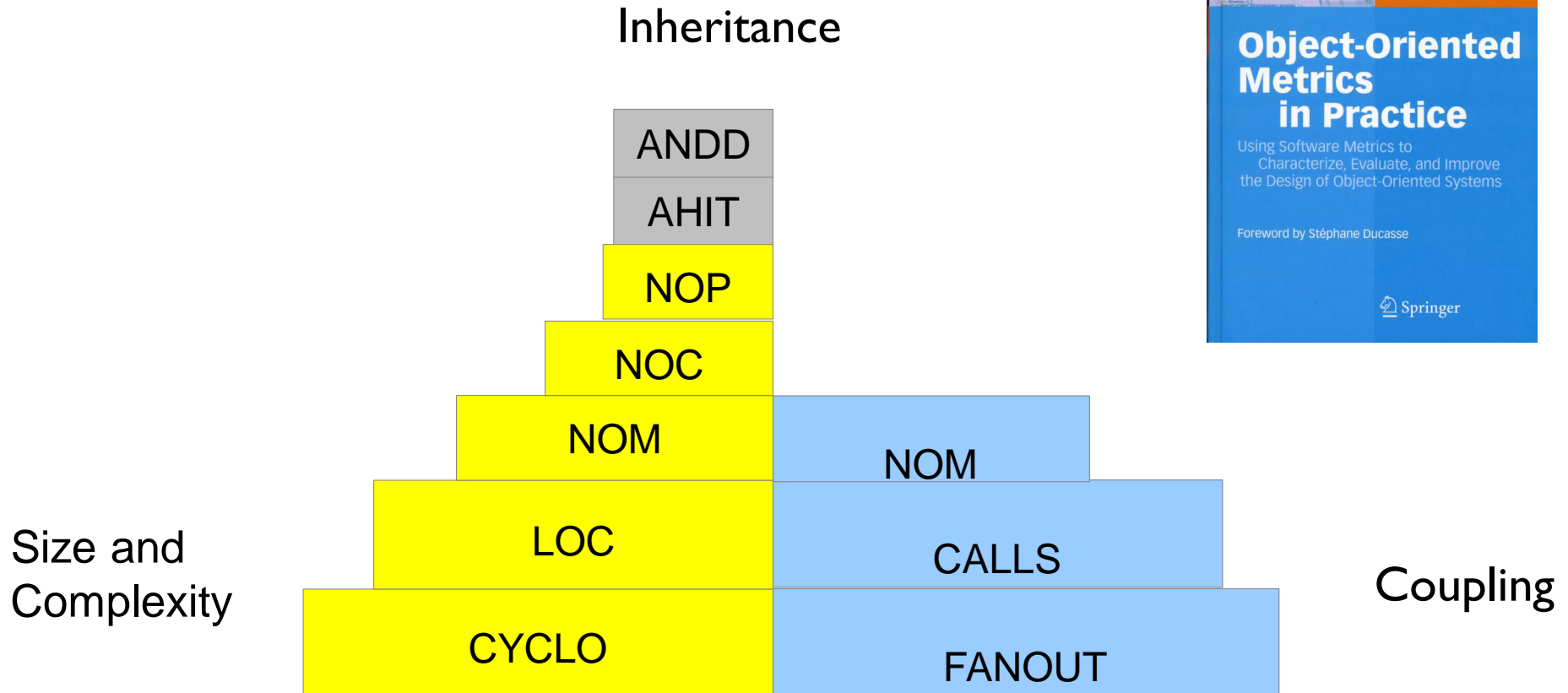
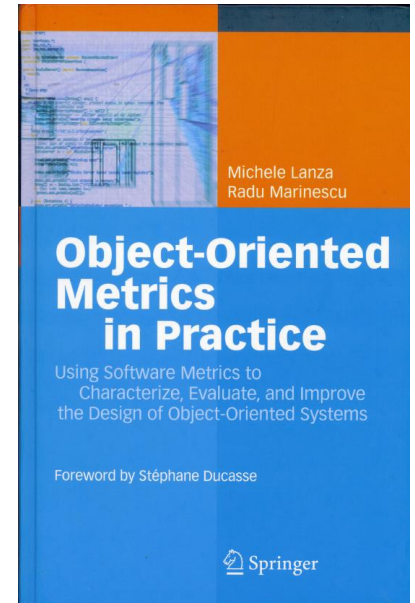
Name	CBO
BeanDefinitionParserDelegate	42
JdbcTemplate	36
AbstractAutowireCapableBeanFactory	34
ScriptFactoryPostProcessor	30
DispatcherServlet	29
MBeanExporter	28
AbstractBeanFactory	26
ConfigBeanDefinitionParser	25
DispatcherPortlet	25
ConstructorResolver	24
AbstractApplicationContext	24
XmlBeanDefinitionReader	23
PropertyEditorRegistrySupport	22
AbstractAspectJAdvice	21
HandlerMethodInvoker	21
Cglib2AopProxy	20
HibernateTransactionManager	20
SessionFactoryUtils	20
JpaTransactionManager	20

class: CBO > 14.0 filter on (class group of system hibernate-3.2) [102]

Name	CBO
NewTests	105
SessionImpl	80
AbstractEntityPersister	79
HbmBinder	67
FooBarTest	59
HqlSqlWalker	58
SessionFactoryImpl	52
Loader	52
AbstractCollectionPersister	50
Configuration	45
DefaultLoadEventListener	37
ASTParserLoadingTest	35
StatelessSessionImpl	33
AbstractSaveEventListener	32
ParentChildTest	32
DefaultFlushEntityEventListener	30
FurnTest	30
MasterDetailTest	30
QueryTranslatorImpl	30
QueryTranslatorImpl	30
DefaultDeleteEventListener	28
CriteriaQueryTest	28

Lanza und Marinescu - Pyramidal Overview

Lanza und Marinescu 2006



Lanza und Marinescu - Pyramidal Overview

iPlasma 6.1

Load Group Property Filter

~root

- class group of system hibernate-3.2
 - God Class filter
- method group of system hibernate-3.2
 - Brain Method filter**
 - Feature Envoy filter
 - Intensive Coupling filter

Brain Method filter on (method group of system hibernate-3.2) [102]

Name	CYCLO	LOC
AbstractCollectionPersister	35	338
SingleTableEntityPersister	37	305
buildSettings	23	272
AbstractEntityPersister	22	259
JoinedSubclassEntityPersister	24	256
bindCollection	31	197
token	41	172
generateCustomReturns	19	156
bindPropertyResults	24	155

Back Forward Save ~root Search

~root

System Detail: hibernate-3.2

NDD	0.18
HIT	0.43
NOP	259
NOC	1862
NOM	19069
LOC	139303
CYCLO	26994
CALL	46269
FOUT	24855

Interpretation of the Overview Pyramid for module **hibernate-3.2**

Class Hierarchies tend to be **tall** and **narrow** (i.e. inheritance trees tend to have many depth-levels and base-classes with few directly derived sub-classes)

Classes tend to:

- be rather **large** (i.e. they define many methods);
- be organized in rather **fine-grained packages** (i.e. few classes per package);

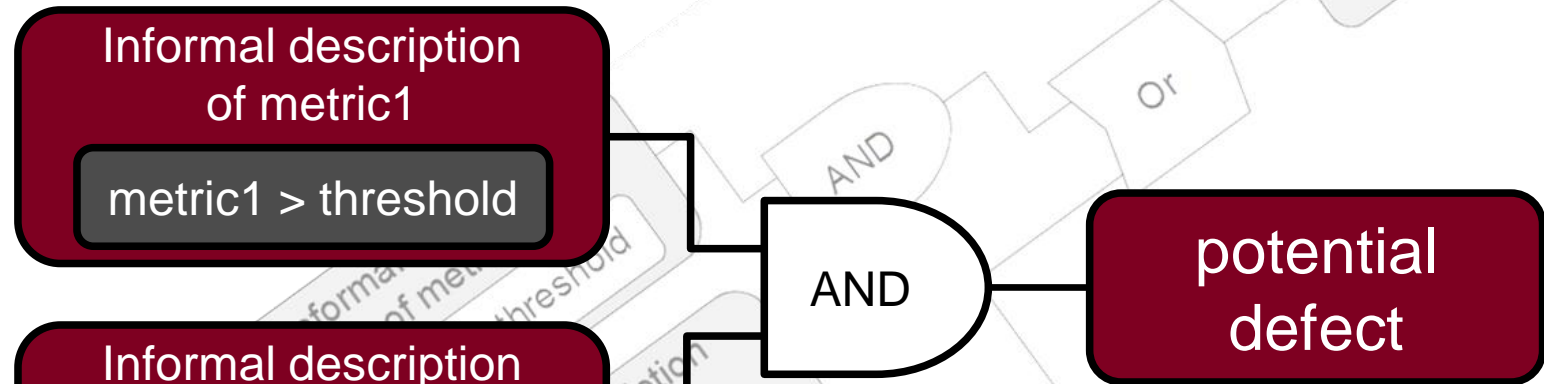
Methods tend to:

- tend to be rather **short** and having an **average logical complexity** ;
- tend to call an **several methods** from **few other classes** (low coupling dispersion);

Wie kann die Suche nach Refaktorisierungskandidaten eingeschränkt werden?

© gjeewAaytee at flickr

Lösung – Metriken kombinieren



AGENDA

- > Motivation
- > Metriken
- > Metriken kombinieren
- > Zusammenfassung, Literatur und Werkzeuge

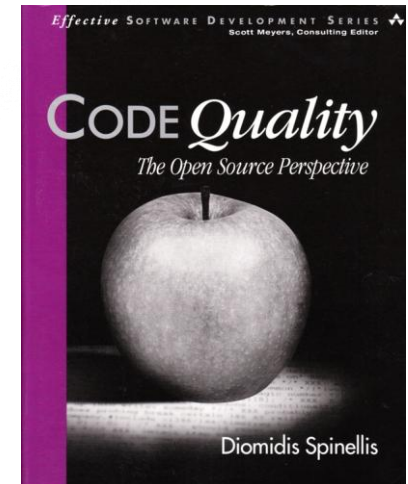
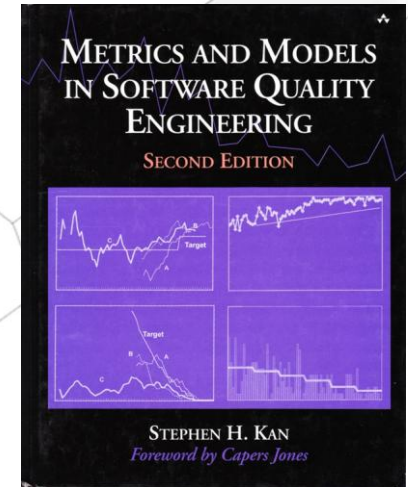
Metriken kombinieren

NASA Heuristiken (Rosenberg et al, 1999)

> Eine Klasse ist ein möglicher Kandidat, wenn mindestens zwei der folgenden Bedingungen erfüllt sind

- WMC > 100
- CBO > 5
- RFC > 100 (Response for a class)
- NOM > 40
- RFC > 5 * NOM

> Spinellis empfiehlt 3 oder 4 Bedingungen als Grundlage für nachfolgende Untersuchungen



Metriken kombinieren

WMC > 100

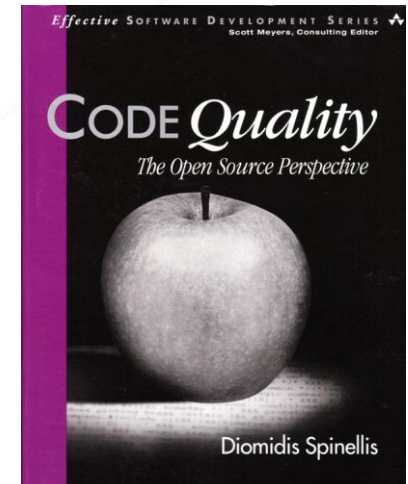
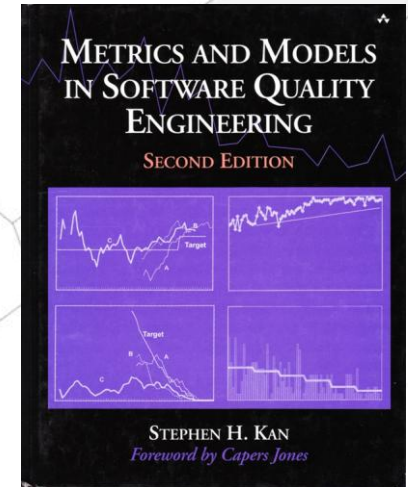
CBO > 5

NOM > 40

NASA Heuristics

Name	WMC	CBO	NOM	LOCC
BeanDefinitionParserDelegate	215	42	47	1288
AbstractBeanFactory	211	26	73	1315
AbstractAutowireCapableBeanFactory	204	34	50	1334
StringUtils	175	2	56	1060
JdbcTemplate	161	36	87	1279
HibernateTemplate	156	6	96	1205
DefaultListableBeanFactory	142	17	33	667
MockHttpServletRequest	139	6	103	798
AbstractPlatformTransactionManager	130	15	49	1198
DispatcherServlet	130	29	36	1139
AbstractBeanDefinition	129	13	77	946
AbstractApplicationContext	126	24	76	1087
MBeanExporter	119	28	42	986
DispatcherServlet	115	25	30	1035
BeanWrapperImpl	114	19	35	814
ClassUtils	112	2	48	934
JtaTransactionManager	111	19	53	1066
JmsTemplate	108	10	73	951
LocalSessionFactoryBean	104	10	39	915
CallMetaDataContext	101	11	27	540
SessionFactoryUtils	101	20	23	705
MimeMessageHelper	101	4	67	985
AdvisedSupport	77	15	43	531
RequestContext	71	16	44	668
TopLinkTemplate	60	8	51	431
JdoTemplate	51	7	44	530

or

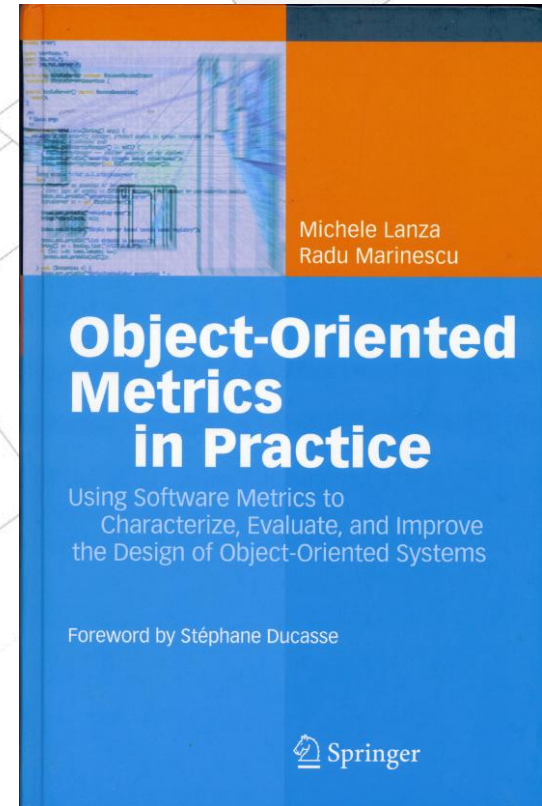


Spring 2.5.6 (26 Klassen = NASA Heuristiken geben keine Hinweise welche Probleme vorliegen)

Metriken kombinieren

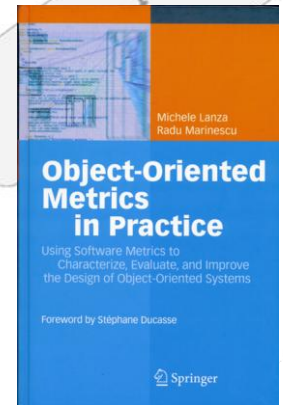
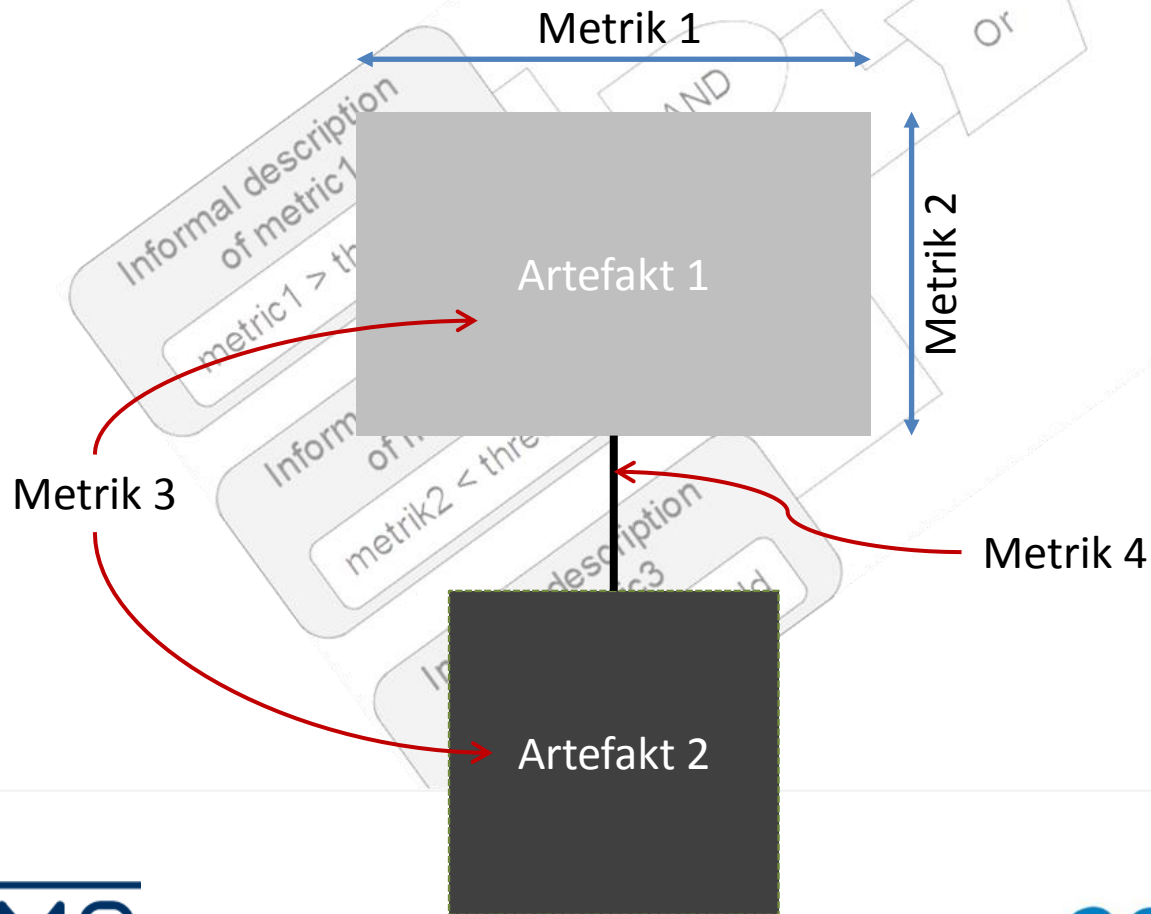
Lanza und Marienscu

- > Nutzen bestehende OO-Metriken, z. B. CK-Suite
- > Definieren eigene Metriken
- > legen Schwellwerte für Metriken fest
 - Basierend auf 45 Java
 - und 37 C++ Projekten
- > Visualisierung
 - Pyramidal Overview
 - **Polymetric views**
 - Class Blue Print
- > **Beschreiben code smell detection strategies basierend auf Metriken**



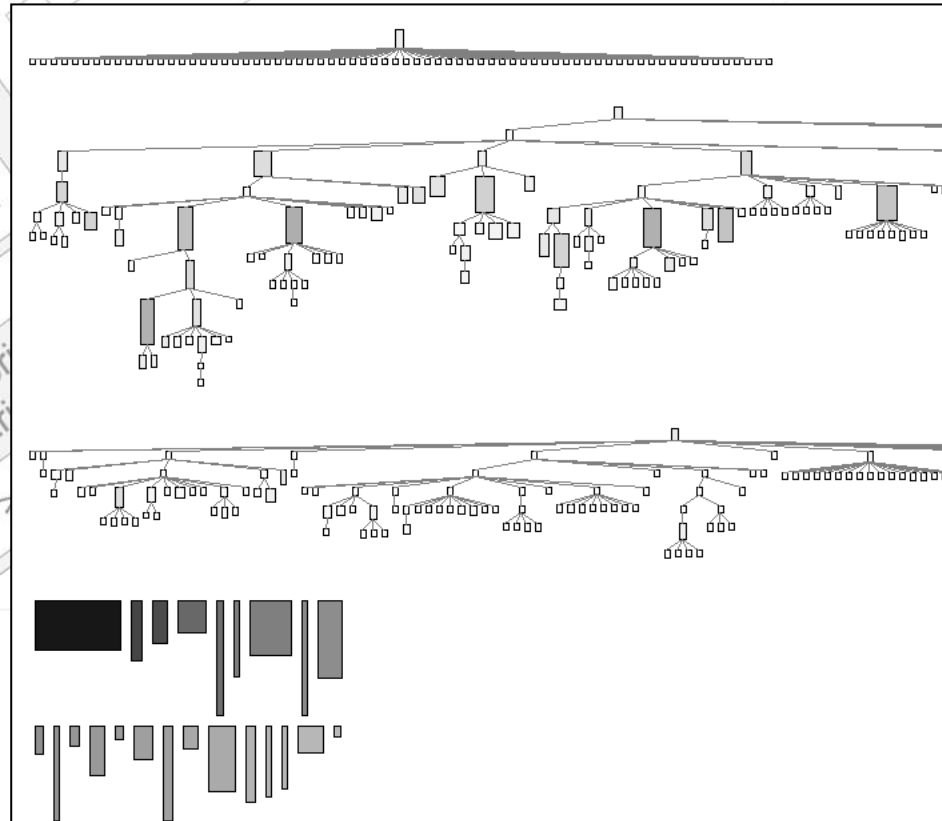
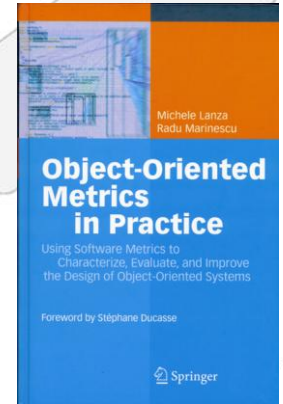
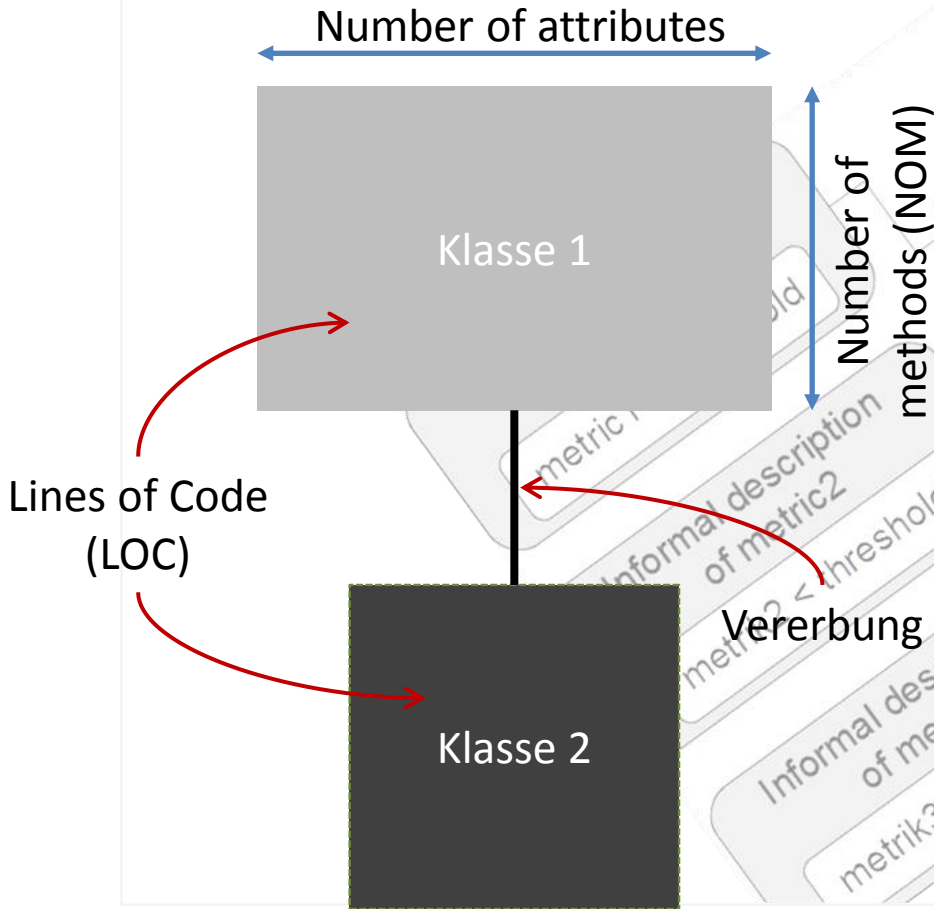
Metriken kombinieren – Polymetric View

- > Dient der gleichzeitigen Visualisierung von unterschiedlichen Metriken auf Artefakten



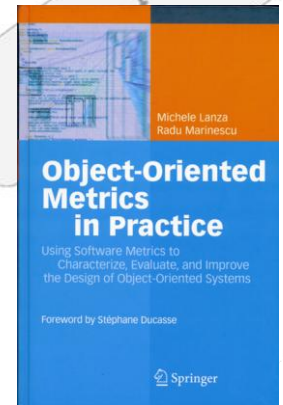
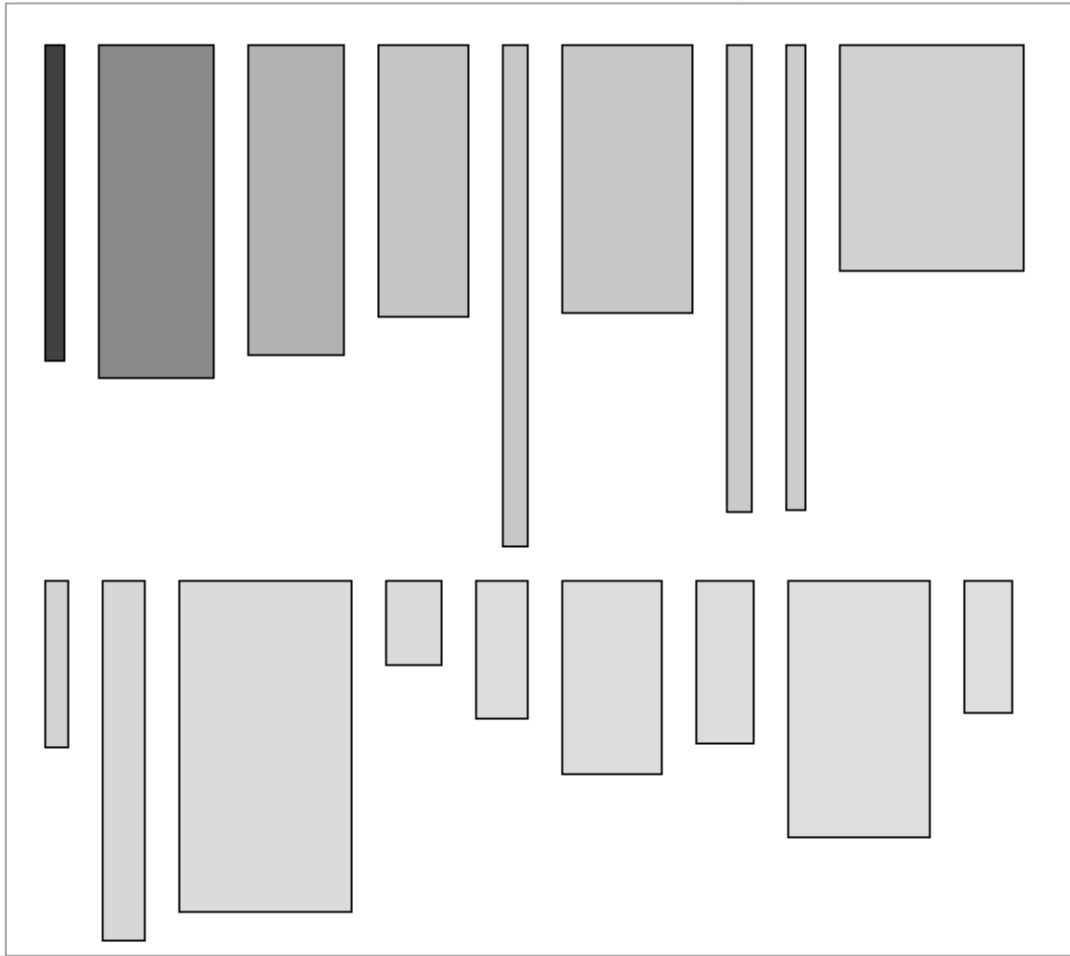
Metriken kombinieren – System Complexity View

> System Complexity View



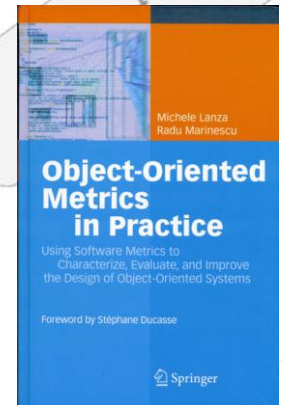
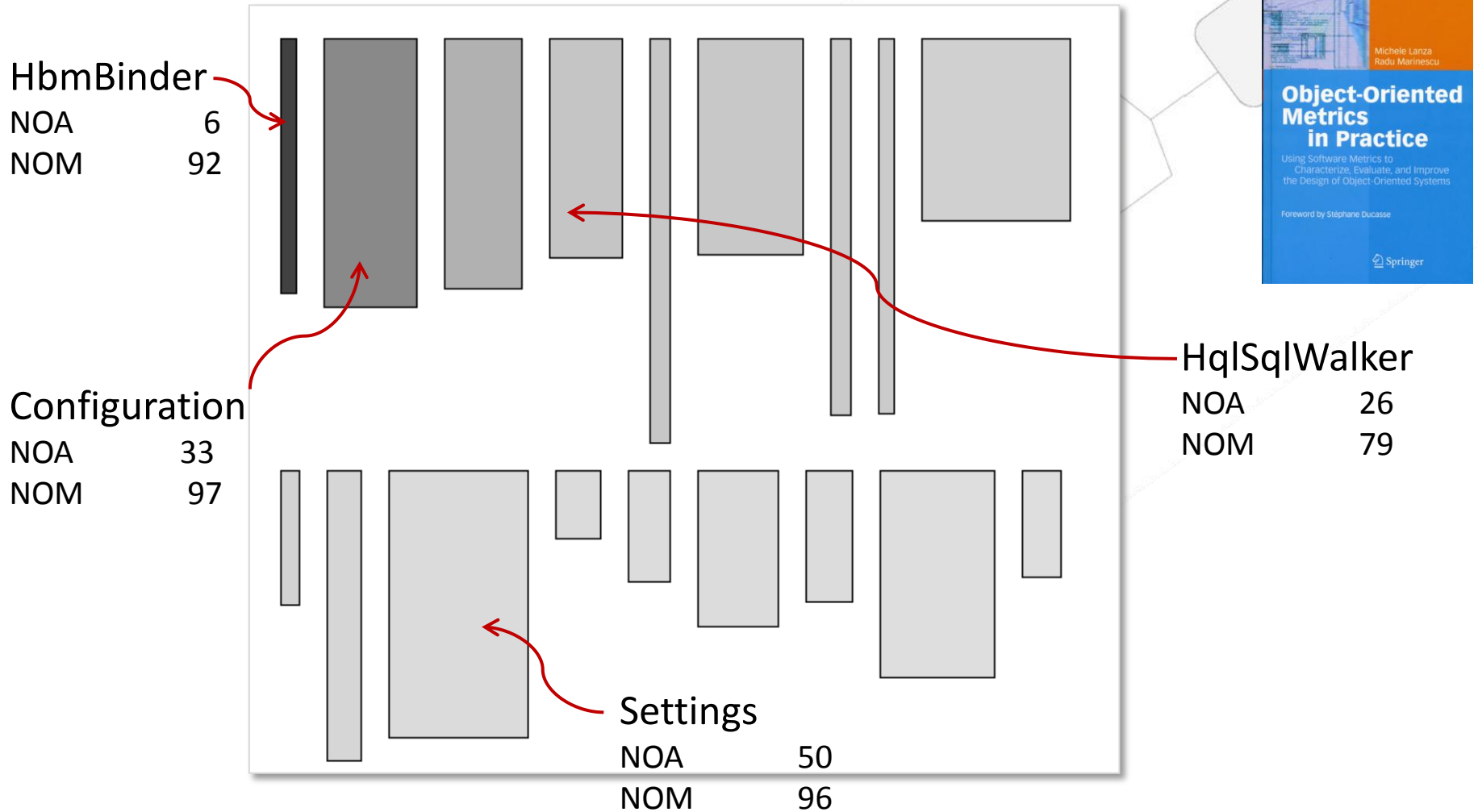
Metriken kombinieren – System Complexity View

Hibernate 3.3.2



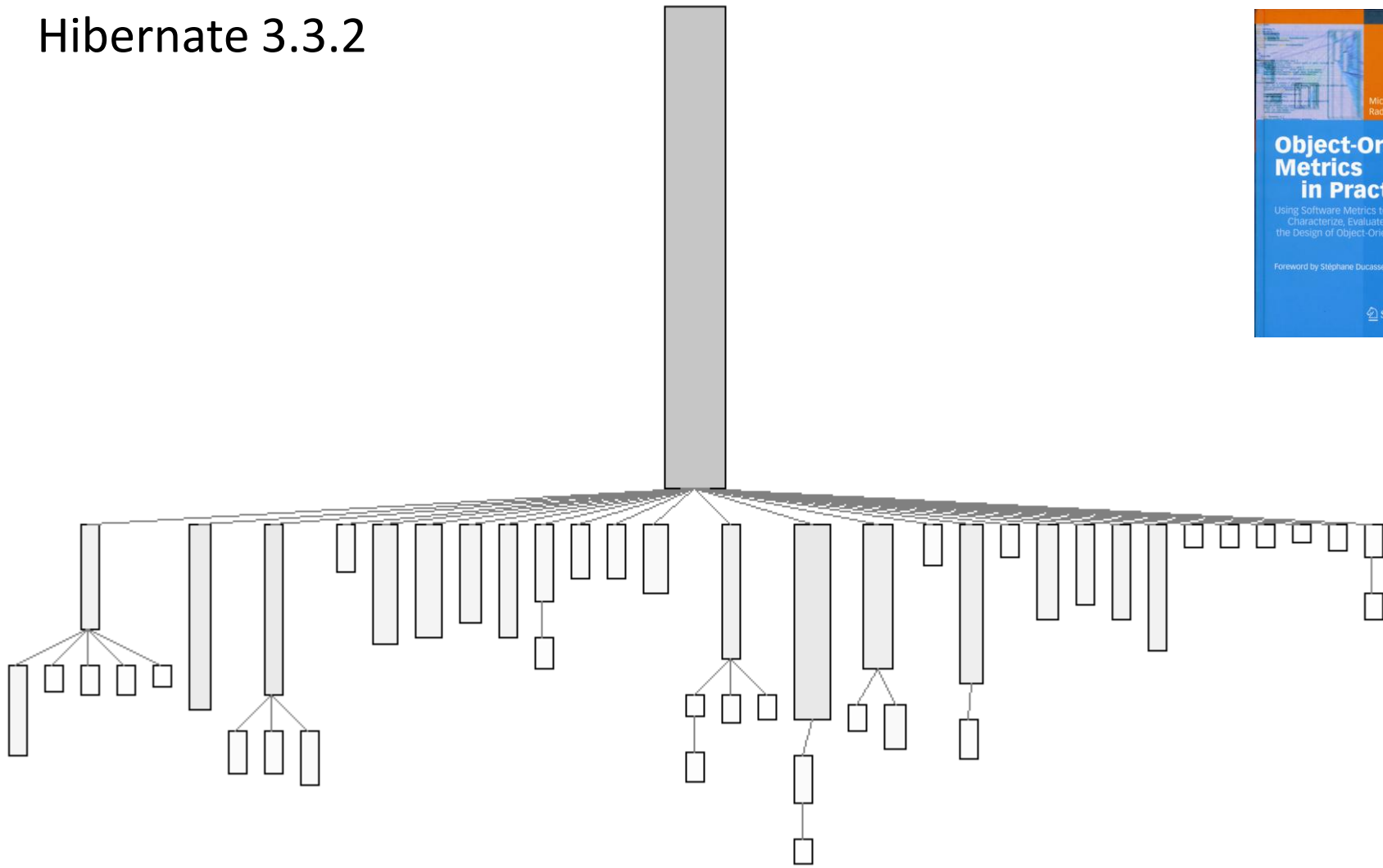
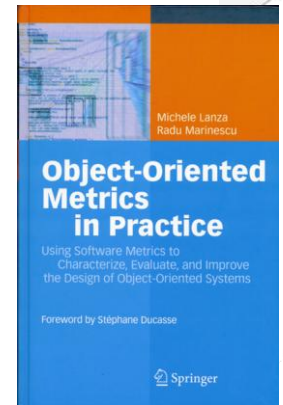
Metriken kombinieren – System Complexity View

Hibernate 3.3.2



Metriken kombinieren – System Complexity View

Hibernate 3.3.2



Metriken kombinieren – System Complexity View

Hibernate 3.3.2

AbstractDialect

NOA 17
NOM 134

AbstractTransactSQLDialect

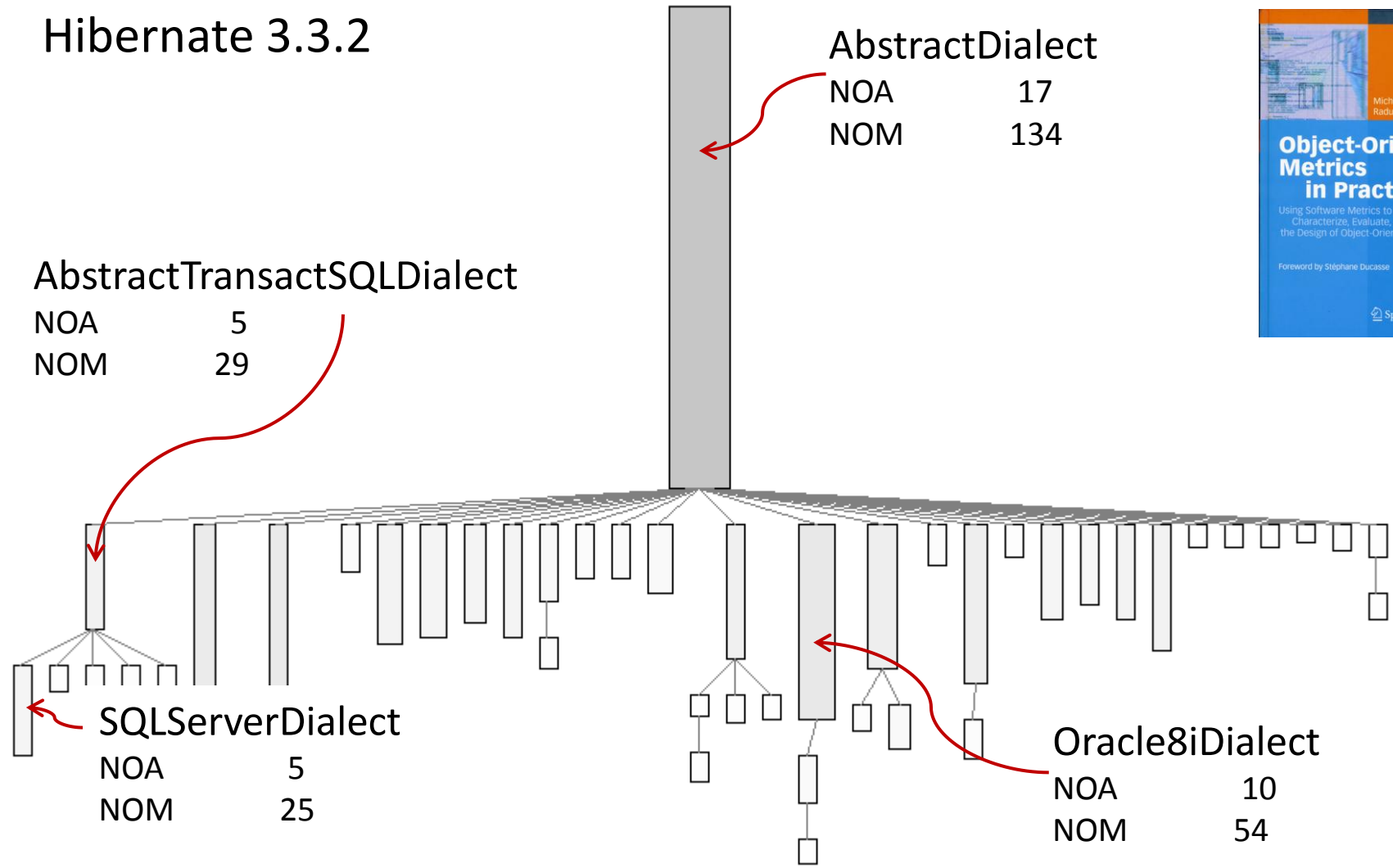
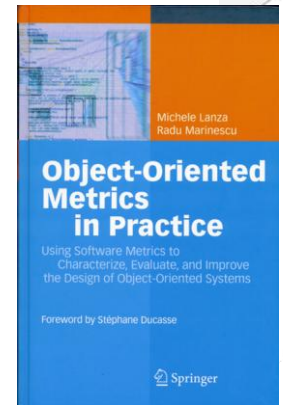
NOA 5
NOM 29

SQLServerDialect

NOA 5
NOM 25

Oracle8iDialect

NOA 10
NOM 54

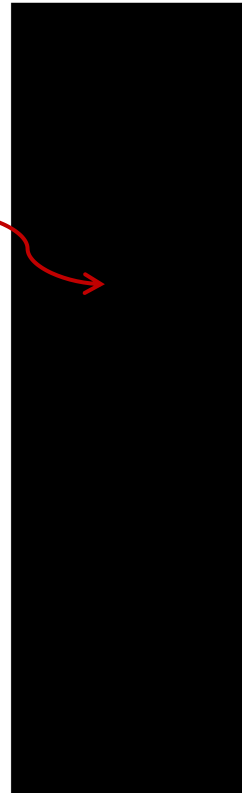


Metriken kombinieren – System Complexity View

Hibernate 3.3.2

AbstractEntityPersister

NOA 90
NOM 299



JoinedSubclassEntityPersister

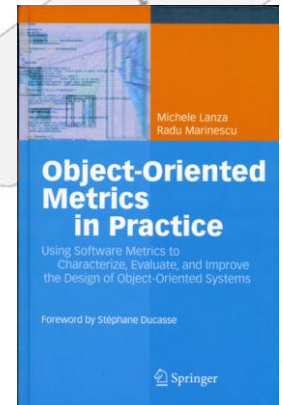
NOA 28
NOM 44

UnionSubclassEntityPersister

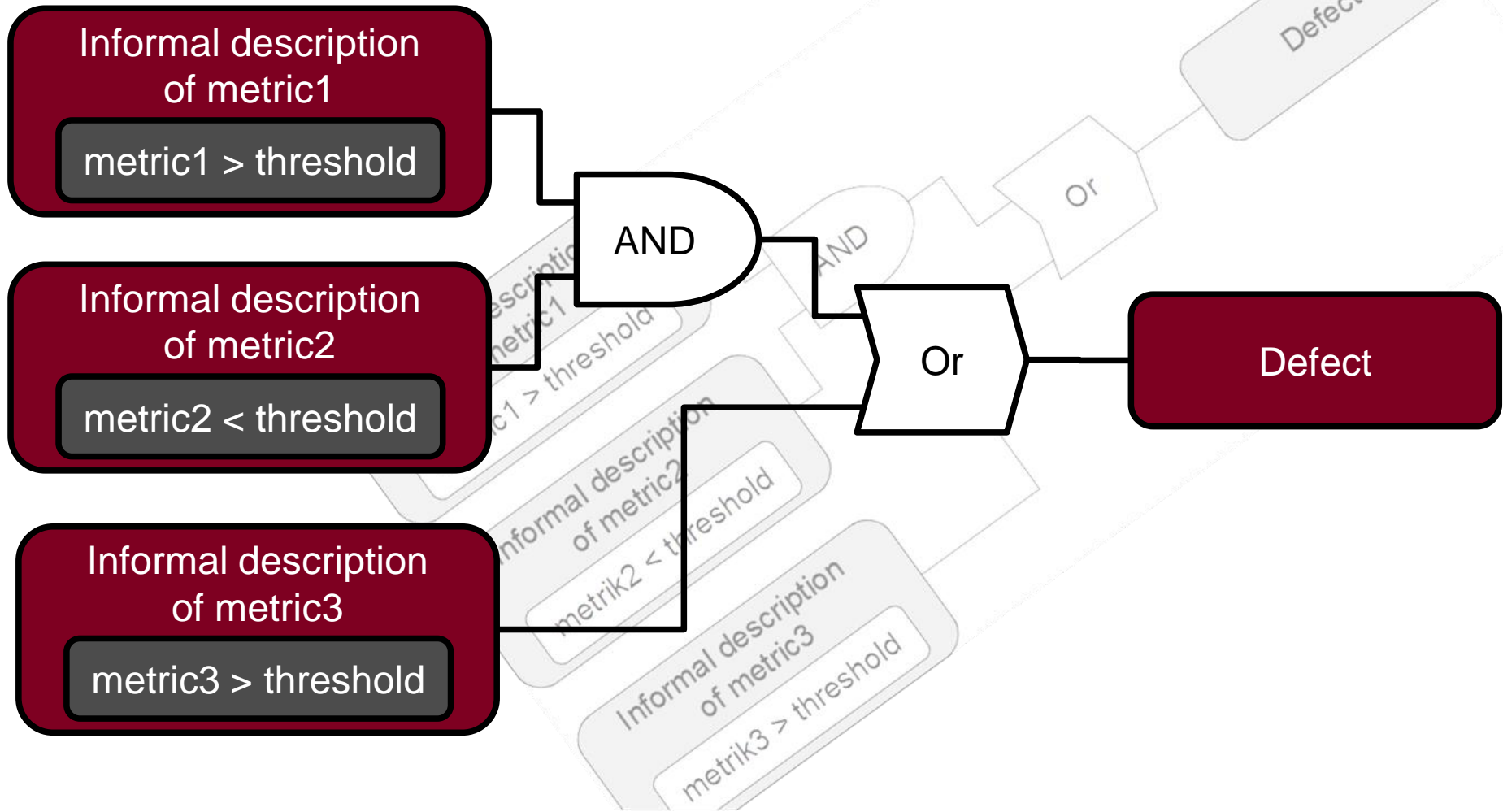
NOA 14
NOM 41

SingleTableEntityPersister

NOA 40
NOM 55



Metriken kombinieren - Detection Strategies



Disharmonies

Lanza und Marienscu

- > Beschreiben drei Kategorien von Designschwächen
 - Identity disharmonies
 - Collaboration disharmonies
 - Classification disharmonies
- > Basieren teilweise auf den Code-Smells von Martin Fowler
- > Disharmonies können voneinander abhängig sein

identity disharmony

collaboration disharmony

classification disharmony

Identity Disharmonies

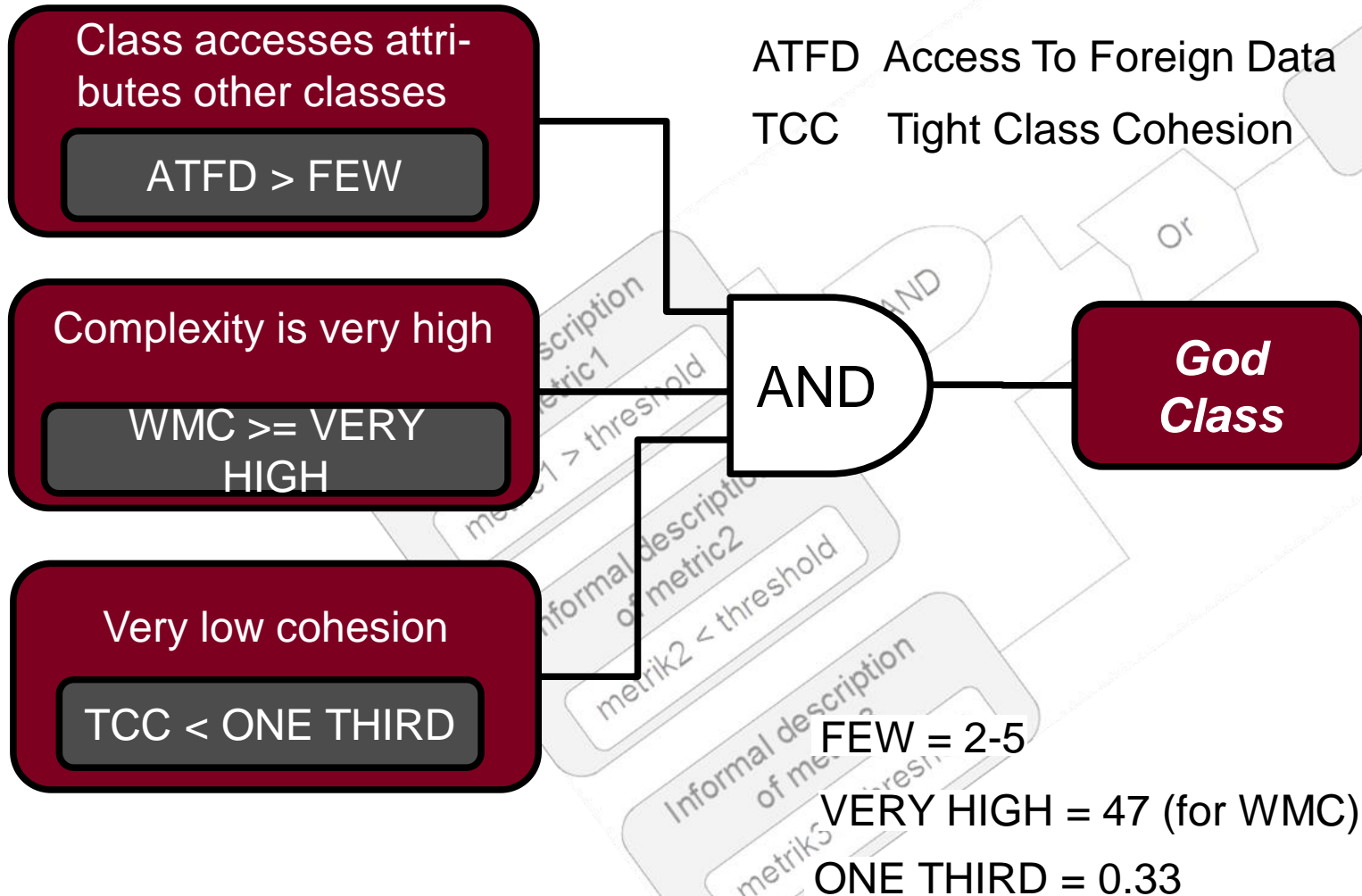
- > *Feature Envy* (Fowler)
 - Methoden sind mehr an Attributen fremder Klassen interessiert
- > *Data Class* (Fowler)
 - Enthalten Daten, aber kein bzw. kaum Verhalten
- > *God class* (Fowler)
 - Interagieren mit vielen anderen Klassen und sind sehr komplex
- > *Brain Method*
 - Methode ist sehr komplex und beinhalten die meiste Funktionalität der Klasse
- > *Brain Class*
 - Enthält viele Brain Methods
- > *Significant Duplication* (Fowler)
 - Klasse enthält einen hohen Anteil an Code Duplizierung

identity
disharmony

collaboration
disharmony

classification
disharmony

Identity Disharmony - *God Class*



identity disharmony

collaboration disharmony

classification disharmony

Identity Disharmony God Class - Hibernate 3.3.2

Defect

identity disharmony

collaboration disharmony

classification disharmony

The screenshot shows the iPlasma 6.1 interface. The top part displays a table of God Class filters for the class group 'org.hibernate.persister.entity.AbstractEntityPersister'. The table has columns for Name, WMC, TCC, ATFD, and LOCC. Below the table, the source code for the class `org.hibernate.persister.entity.AbstractEntityPersister` is shown, highlighting its status as a God Class with 9 BrainMethods and 8 IntensiveCouplings. The code lists methods and attributes, including `addDiscriminatorToInsert`, `addDiscriminatorToSelect`, `getSubclassColumnTableNumberClosure`, `private Logger log`, `public String ENTITY_CLASS`, and `private SessionFactoryImplementor factory`.

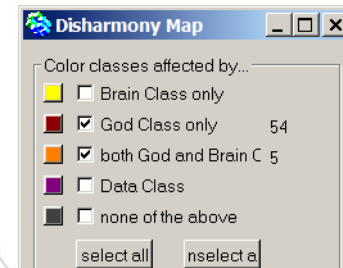
Name	WMC	TCC	ATFD	LOCC
AbstractEntityPersister	653	0,08	306	3752
HbmBinder	483	0,02	185	2971
SessionImpl	299	0,05	47	1854
Configuration	297	0,11	129	2055
FooBarTest	293	0	171	4700
AbstractCollectionPersister	271	0,01	34	1709
Loader	253	0,07	25	2274
QueryTranslatorImpl	203	0,02	8	1103
StatefulPersistenceContext	199	0,08	91	1414
HQLTest	197	0	4	1305
AbstractQueryImpl	158	0,03	7	828
Table	152	0,02	25	747

```
class public org.hibernate.persister.entity . AbstractEntityPersister
GodClass 9 BrainMethod 8 IntensiveCoupling

Object
AbstractEntityPersister

methods (295)
attributes (86)
protected addDiscriminatorToInsert
protected addDiscriminatorToSelect
protected getSubclassColumnTableNumberClosure
private Logger log
public String ENTITY_CLASS
private SessionFactoryImplementor factory
```

Identity Disharmony *God Class* - Hibernate 3.3.2

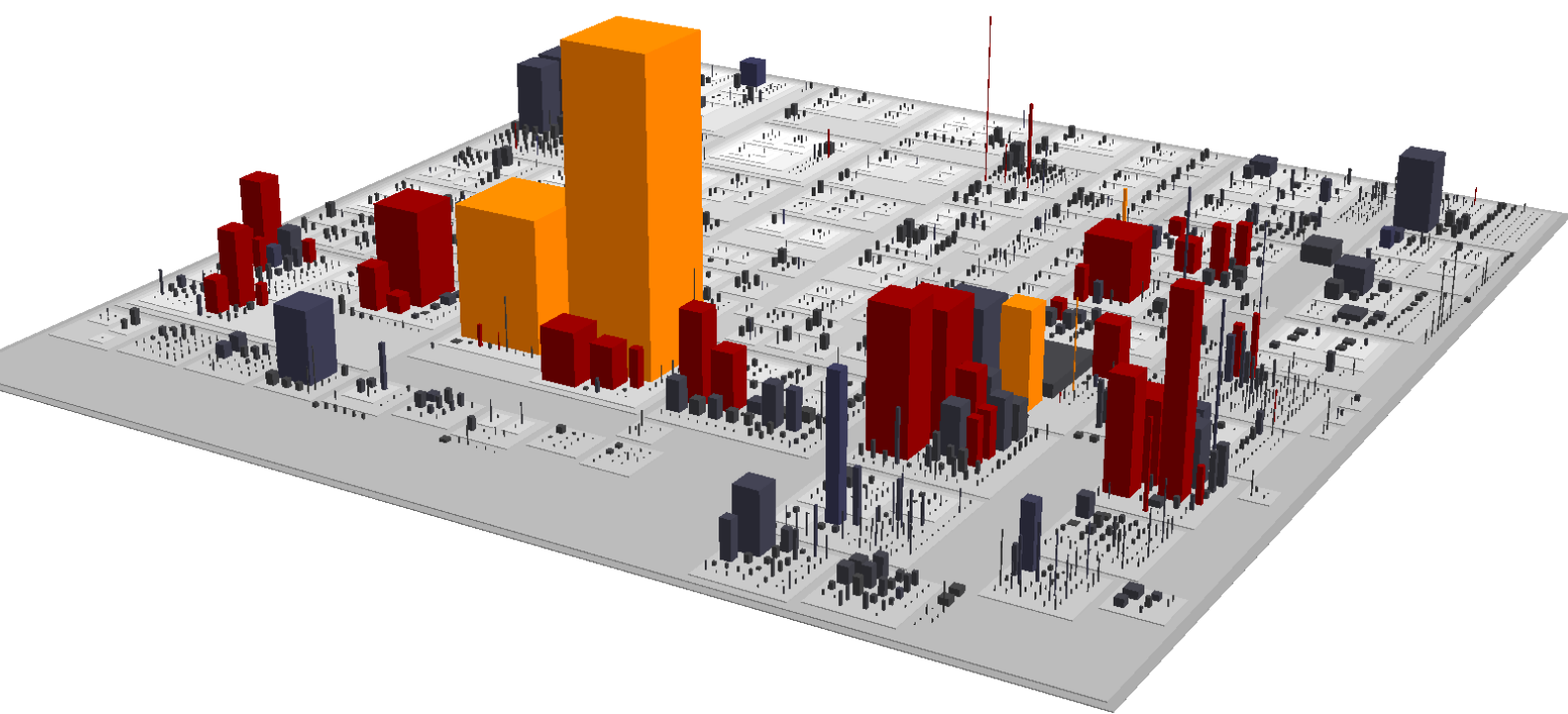


Defect

identity disharmony

collaboration disharmony

classification disharmony



Identity Disharmony God Class - Hibernate 3.3.2

AbstractEntityPersister

Disharmony Map

Color classes affected by...

- Brain Class only
- God Class only 54
- both God and Brain C 5
- Data Class
- none of the above

select all nselect a

identity
disharmony

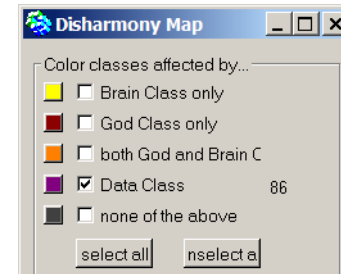
collaboration
disharmony

classification
disharmony

SingleTableEntityPersister

JoinedSubclassEntityPersister

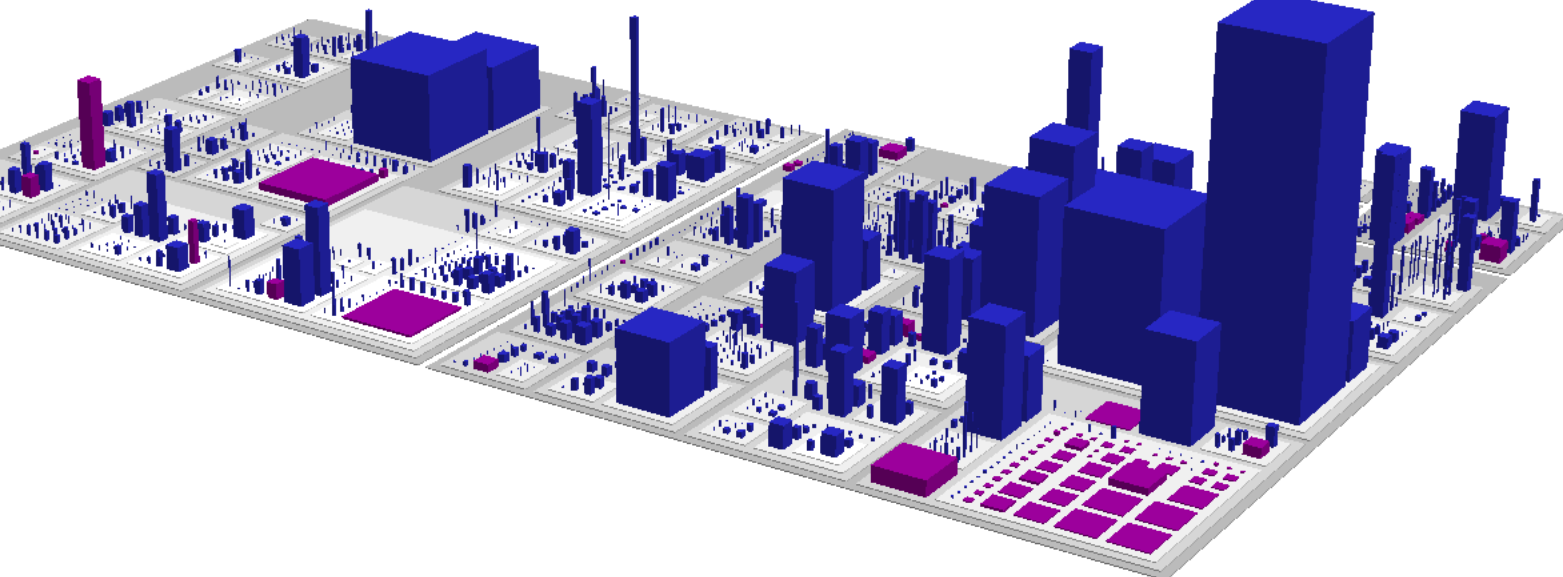
Identity Disharmony *Data Class* – in .Net



Defect

identity disharmony

Spring.Net

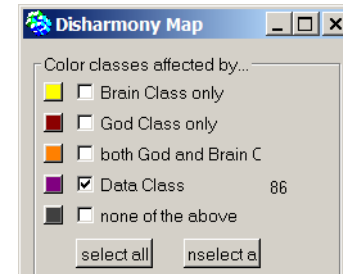


collaboration disharmony

classification disharmony

— NHibernate

Identity Disharmony *Data Class* – in .Net

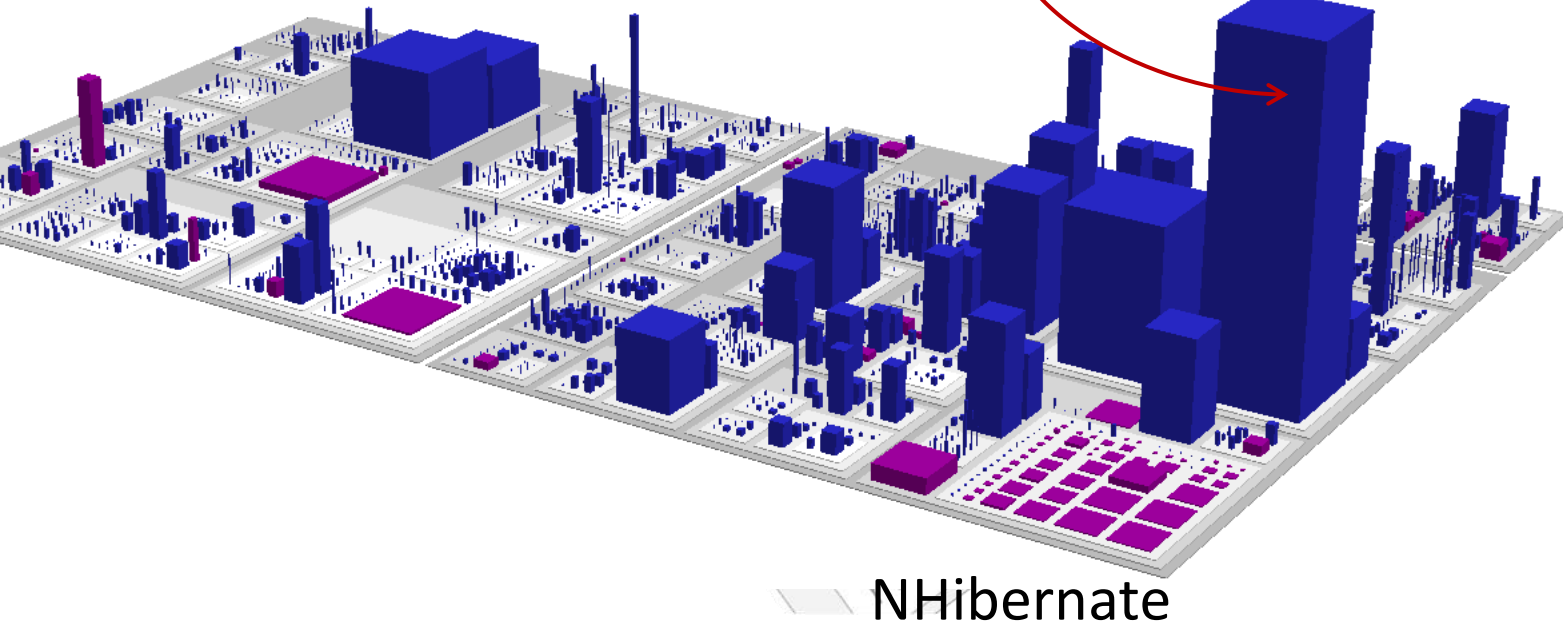


Defect

identity disharmony

AbstractEntityPersister

Spring.Net

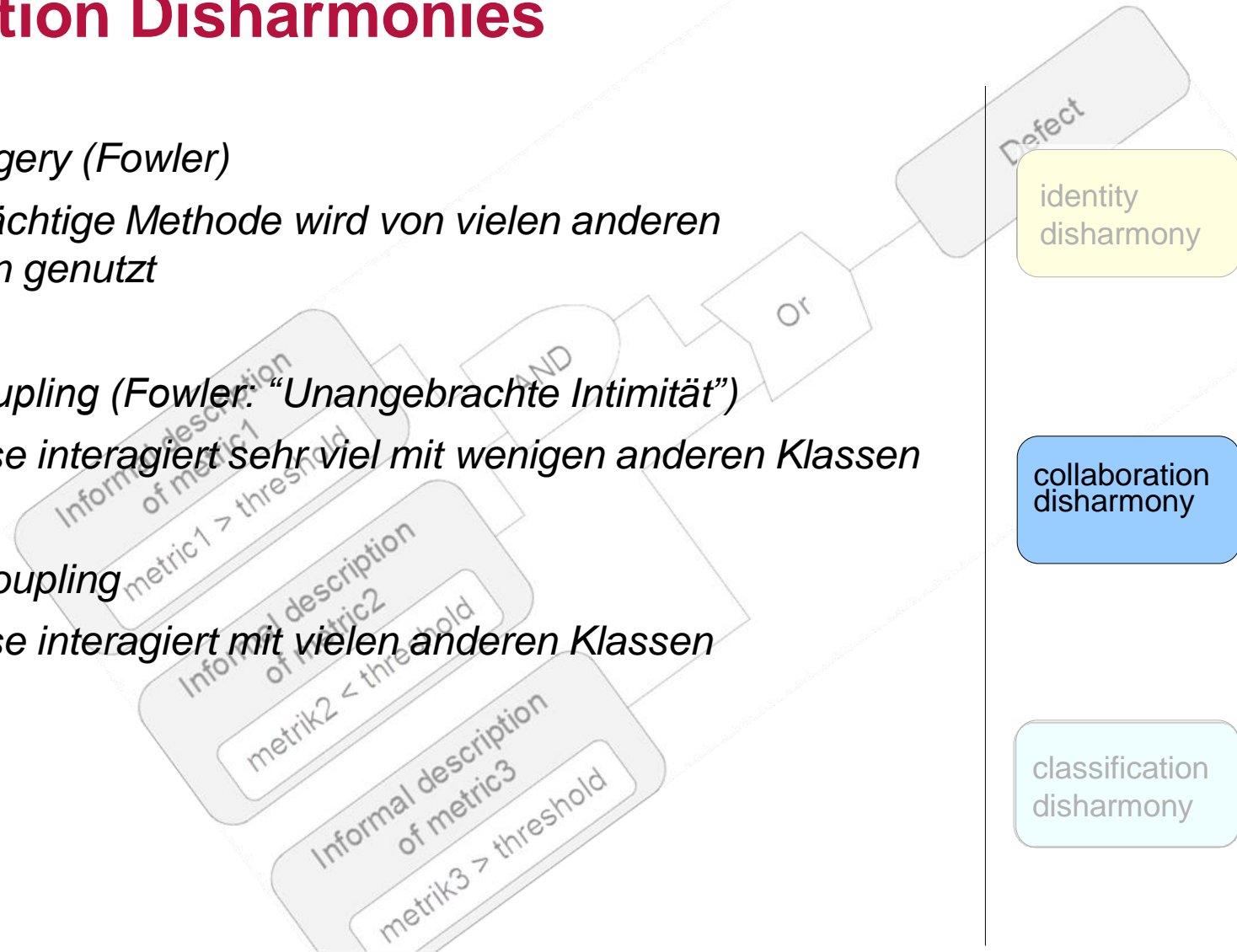


collaboration disharmony

classification disharmony

Collaboration Disharmonies

- > *Shotgun Surgery (Fowler)*
 - *Die verdächtige Methode wird von vielen anderen Methoden genutzt*
- > *Intensive Coupling (Fowler: “Unangebrachte Intimität”)*
 - *Die Klasse interagiert sehr viel mit wenigen anderen Klassen*
- > *Dispersed Coupling*
 - *Die Klasse interagiert mit vielen anderen Klassen*



Collaboration Disharmony - *Shotgun Surgery*

Method is called by too many methods

CM > Short Memory Cap

[CM = Changing Methods]

CM: Anzahl der Methoden, die die betrachtete Methode aufrufen

Short Memory Capacity = 7-8

AND

Shotgun Surgery

Incoming calls are from many other classes

CC > Many

[CC = Changing Classes]

CC : Anzahl der Klassen, deren Methoden die betrachtete Methode aufrufen.

Many > 7

identity disharmony

collaboration disharmony

classification disharmony

Defect

Or

Informal description of metric1
metric1 > threshold

Informal description of metric2
metric2 < threshold

Informal description of metric3
metric3 > threshold

Collaboration - Shotgun Surgery

The screenshot shows the iPlasma 6.1 IDE interface. The top window displays a class hierarchy for 'org.hibernate.Session' with a 'Shotgun Surgery filter' applied. The filter table lists various methods and their associated counts for different categories (CC, CM).

scope name	Name	CC	CM
Session	close	198	774
Session	beginTransaction	185	593
Transaction	commit	183	585
Session	delete	144	454
Session	save	109	351
Session	createQuery	121	324
Session	flush	81	255
Query	list	84	222
Session	persist	84	202
SessionImplementor	getFactory	102	189
SessionImplementor	getEntityMode	74	172
Session	get	96	168
Session	connection	27	161
Session	getTransaction	71	150
Session	load	44	150
SessionImplementor	getPersistenceContext	61	127
Query	executeUpdate	39	97
SessionFactory	getStatistics	55	93
Session	clear	59	86
Session	createCriteria	45	85
EntityPersister	getEntityName	53	83
Query	uniqueResult	58	78

The bottom window shows the source code for the `org.hibernate.Session` interface, listing 65 methods and 0 attributes. The methods listed include:

- public `getEntityMode`
- public `getSession`
- public `flush ShotgunSurgery`
- public `setFlushMode`
- public `getFlushMode`
- public `setCacheMode`
- public `getCacheMode`
- public `getSessionFactory ShotgunSurgery`
- public `connection ShotgunSurgery`
- public `close ShotgunSurgery`
- public `cancelQuery`
- public `isOpen ShotgunSurgery`
- public `isConnected`
- public `isDirty`
- public `getIdentifier`
- public `contains ShotgunSurgery`
- public `evict ShotgunSurgery`

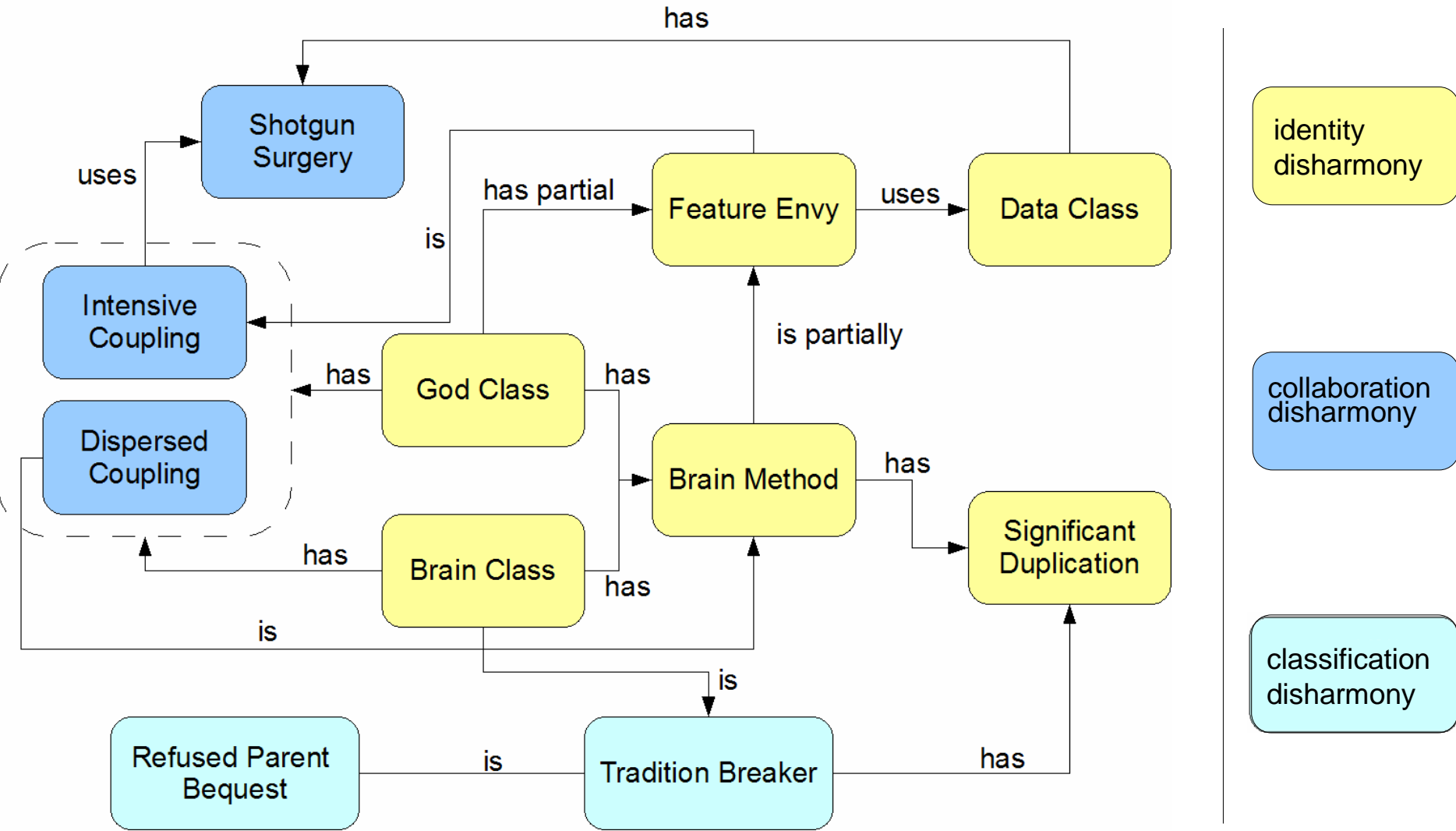
Defect

identity disharmony

collaboration disharmony

classification disharmony

Abhängigkeiten der *Disharmonies*



Viele Disharmonies ...

iPlasma 6.1

Name	WMC	TCC	ATFD	LOCC
AbstractEntityPersister	653	0,08	306	3752

org.hibernate.persister.entity.AbstractEntityPersister

```
class public org.hibernate.persister.entity . AbstractEntityPersister
GodClass 9 BrainMethod 8 IntensiveCoupling

Object
AbstractEntityPersister

This is a God Class because:
• some of its methods access directly (or via getter/setters) 306.0 attributes from 73 external classes:
  ○ isSubclassEntityName
    □ GodClass 2 FeatureEnvy 1 BrainMethod 1 IntensiveCoupling EntityMetamodel :: getSubclassEntityNames

  ○ AbstractEntityPersisterBrainMethod IntensiveCoupling
    □ GodClass 1 FeatureEnvy 1 IntensiveCoupling PersistentClass :: getFilterMap
    □ GodClass 2 FeatureEnvy 1 BrainMethod 1 IntensiveCoupling EntityMetamodel :: getPropertySpan
    □ GodClass 3 FeatureEnvy 2 BrainMethod 3 IntensiveCoupling Table :: getRowId
    □ Property :: getPersistentClass
    □ Property :: getValue
    □ Settings :: getDefaultBatchFetchSize
    □ GodClass 1 FeatureEnvy 1 IntensiveCoupling PersistentClass :: getTemporaryIdTableName
    □ Property :: getName
    □ GodClass 1 FeatureEnvy 1 IntensiveCoupling PersistentClass :: getTemporaryIdTableDDL
    □ GodClass 1 FeatureEnvy 1 IntensiveCoupling PersistentClass :: getEntityName
    □ GodClass 1 FeatureEnvy 1 IntensiveCoupling PersistentClass :: getBatchSize
    □ GodClass 1 FeatureEnvy 1 IntensiveCoupling PersistentClass :: getLoaderName

  ○ generateLazySelectString
    □ ArrayHelper :: toArray
    □ GodClass 2 FeatureEnvy 1 BrainMethod 1 IntensiveCoupling EntityMetamodel :: hasLazyProperties
```

identity disharmony

collaboration disharmony

classification disharmony

AGENDA

- > Motivation
- > Metriken
- > Metriken kombinieren
- > Zusammenfassung, Literatur und Werkzeuge

Zusammenfassung

Metriken messen

- > Größe und Komplexität
- > Vererbung
- > Kopplung

- > Metriken können zur Aufdeckung von potenziellen Refaktorisierungen genutzt werden

- > Durch geeignete Kombination können (Fowlersche) *Code Smells* aufgedeckt werden

- > **Metriken müssen grundsätzlich bewertet und interpretiert werden**

- > Komplementär zu Werkzeugen wie PMD, JDepend

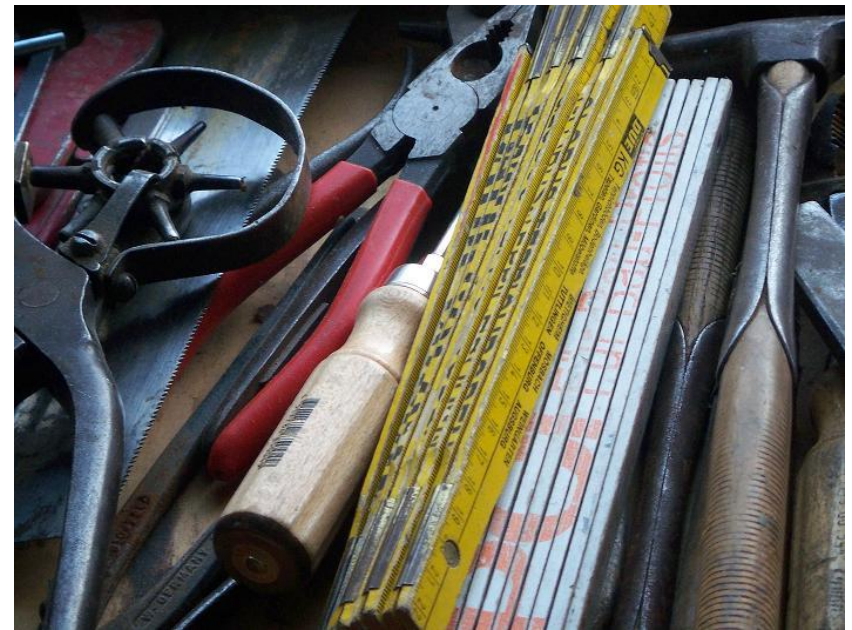
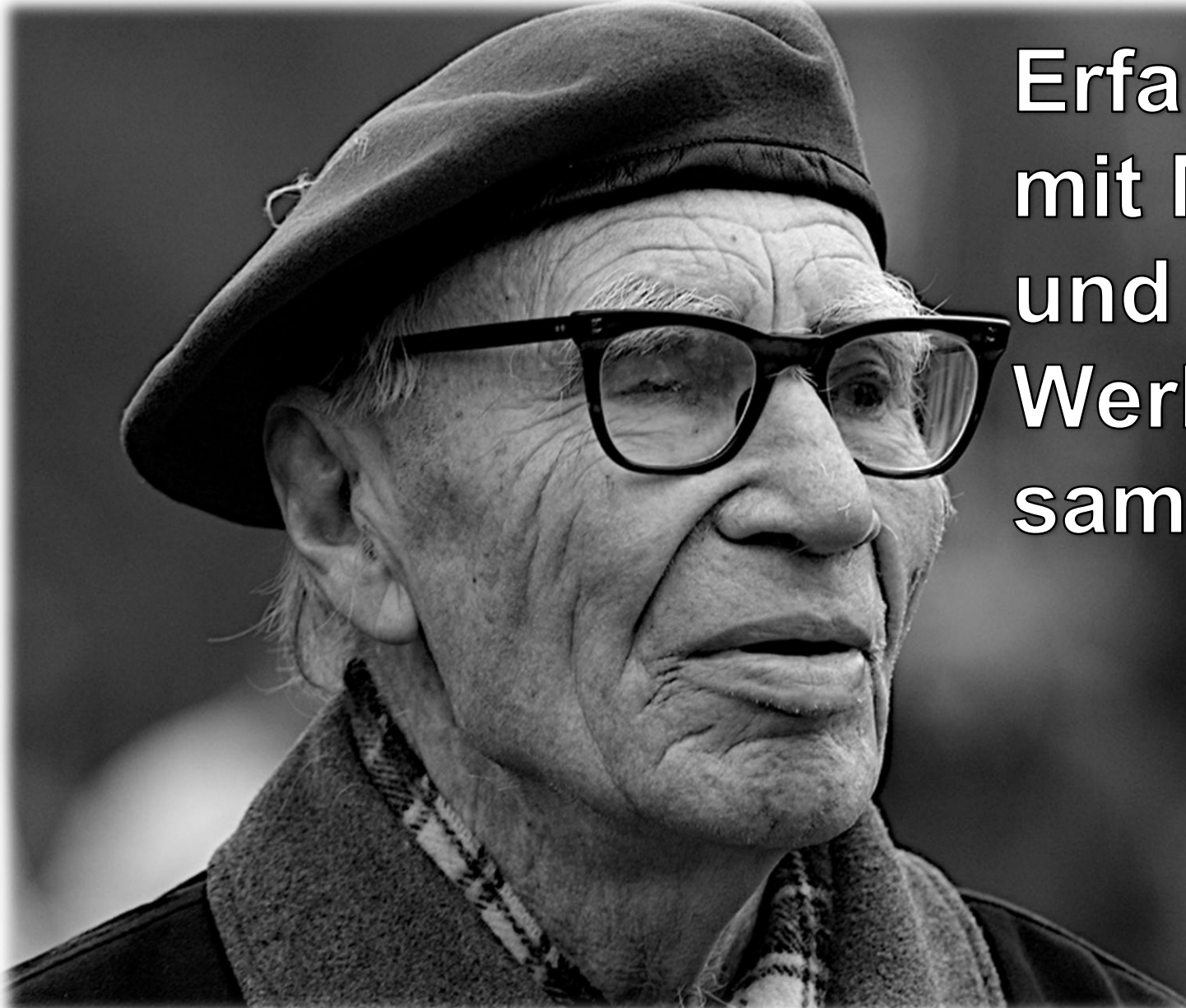


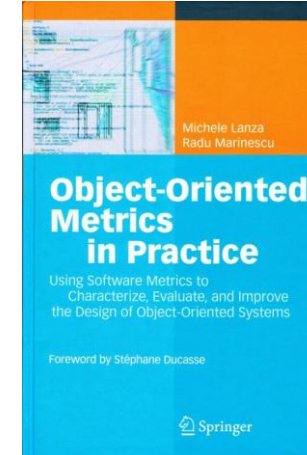
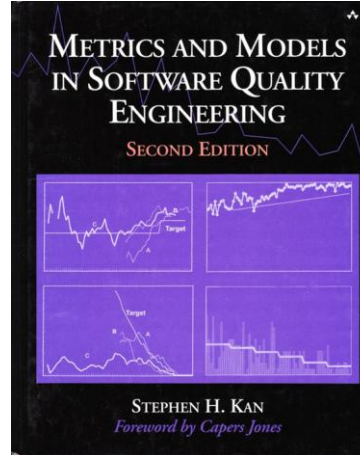
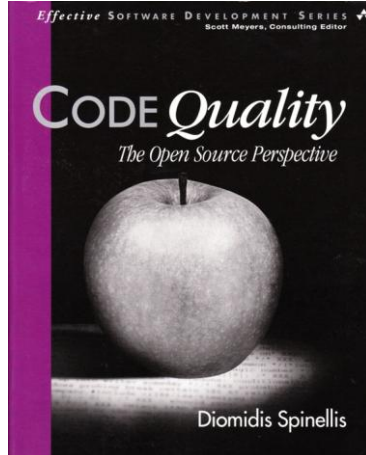
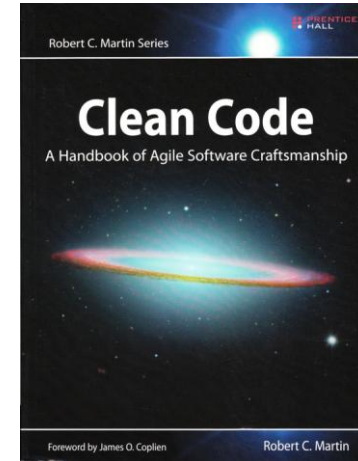
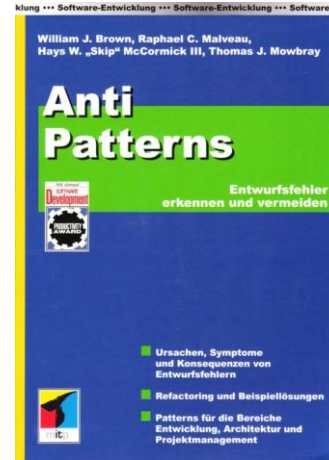
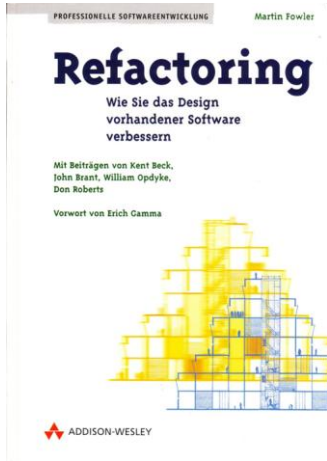
Foto: pixelio.de - Olaf Rendler



Erfahrung mit Metriken und Werkzeugen sammeln

© by Steve Punter at flickr

Literatur



(Some) Tools

- > JavaNCSS <http://www.kclee.de/clemens/java/javancss/>
- > ckjm <http://www.spinellis.gr/sw/ckjm/>
- > PMD <http://pmd.sourceforge.net/>
- > IPlasma <http://loose.upt.ro/iplasma/index.html>
- > InFusion <http://www.intooitus.com>
- > InCode <http://loose.upt.ro/incode/pmwiki.php/Main/Incode?from=Main.InCode>
- > CodeCity <http://www.inf.unisi.ch/phd/wettel/codecity.html>
- > MOOSE <http://moose.unibe.ch/>
- > Metrics <http://metrics.sourceforge.net/>
- > SonarJ <http://www.hello2morrow.com/products/sonarj>

Vielen Dank - Thomas.Haug@mathema.de

„However, a metric is not a god; it is merely a measurement against an arbitrary standard“

Robert C. Martin

$$\begin{aligned} MI = & 171 - 5,2 * l(\text{average}(V)) \\ & - 0,23 * \text{average}(g') \\ & - 16,2 * \ln(\text{average}(LOC)) \\ & + 50 * \sin\left(\sqrt{(2,4 * PerCM)}\right) \end{aligned}$$

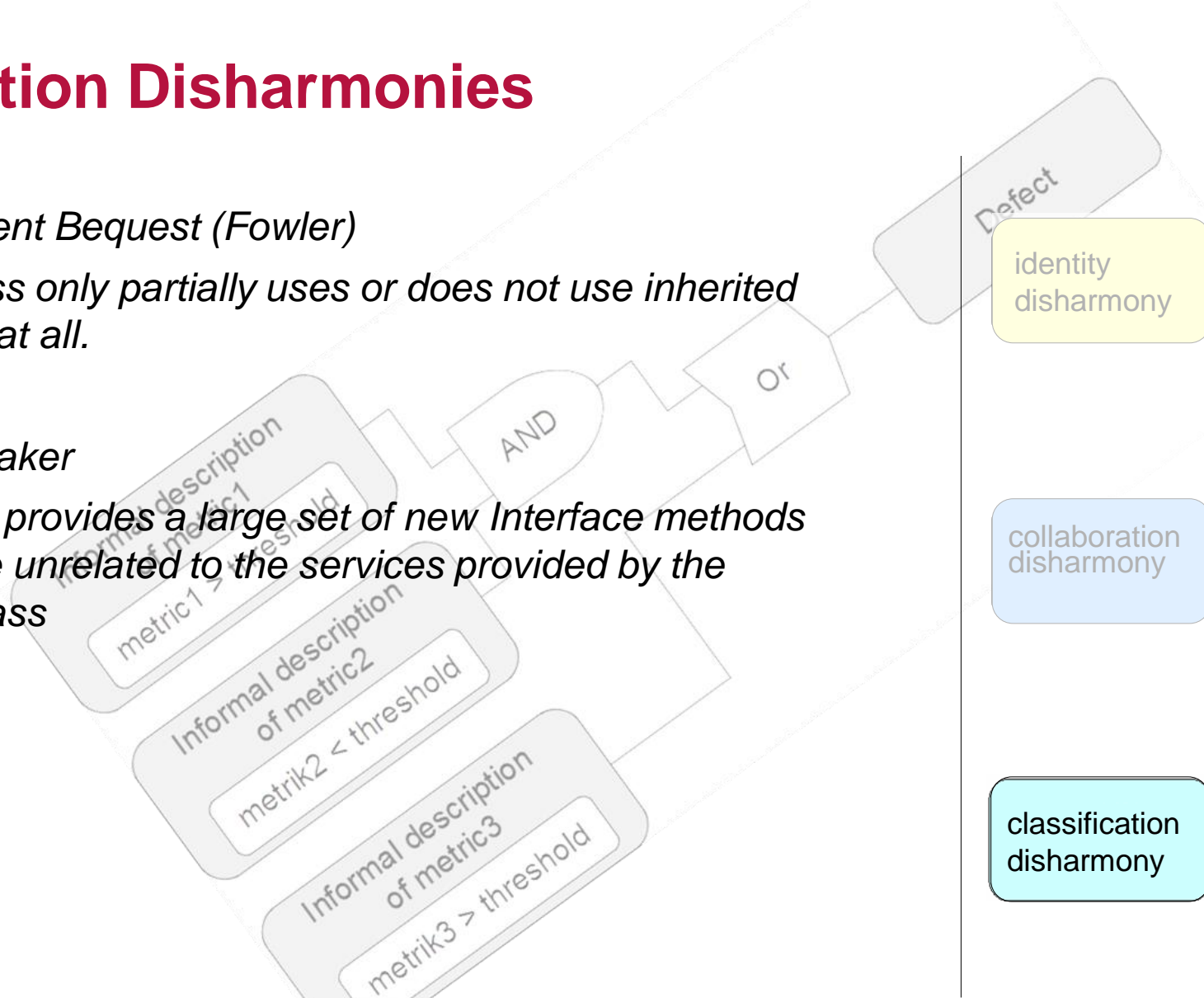
Classification Disharmonies

> *Refused Parent Bequest (Fowler)*

- *Child class only partially uses or does not use inherited methods at all.*

> *Tradition Breaker*

- *Subclass provides a large set of new Interface methods which are unrelated to the services provided by the parent class*



Classification Disharmonies

- > *Refused Parent Bequest (Fowler)*
- > *Tradition Breaker*

The screenshot shows the inFusion 7.2.11 IDE interface. The main window displays the source code for the class `DefaultMessageListenerContainer` in the package `org.springframework.jms.listener`. The class hierarchy is shown as follows:

- Object
- JmsAccessor
- JmsDestinationAccessor
- AbstractJmsListeningContainer
- AbstractMessageListenerContainer
- AbstractPollingMessageListenerContainer
- DefaultMessageListenerContainer

The class overview shows 35 methods and 19 attributes. The design problems section identifies two issues:

- Tradition Breaker**: The class extends substantially the interface inherited from its base class. More precisely, the class adds a significant number of public methods (7 new methods). As a result, the interface of the class has increased with 93% relative to its base-class.
- Refused Parent Bequest**: The class uses only sparsely the inheritance-specific (i.e. protected) members defined by its base class. More precisely, the class uses only 18% of the 11 protected members provided by the base class.

Additional analysis notes for both issues state: "Furthermore, the class does override only 10 methods which is 28% of all methods in the class, and it specializes 5 methods which is 14% of all methods."

Defect

identity disharmony

collaboration disharmony

classification disharmony

Identity Disharmony *God Class* - Spring 2.5.6

Name	ATFD	WMC	TCC
BeanDefinitionParserDelegate	53	215	0,13
PropertyEditorRegistrySupport	26	68	0,21
AbstractAutowireCapableBeanFactory	18	204	0,03
HibernateTransactionManager	14	100	0,11
LocalSessionFactoryBean	14	104	0,06
StaticListableBeanFactory	12	53	0,26
JndiRmiClientInterceptor	11	48	0,07
JdoTransactionManager	11	60	0,17
JpaTransactionManager	11	66	0,11
BeanWrapperImpl	11	114	0,17
ScriptFactoryPostProcessor	9	48	0,11
GenericTableMetaDataProvider	9	49	0,05
NamedParameterUtils	9	56	0
Order	9	58	0,05
AbstractBeanDefinition	9	129	0,06
AbstractPlatformTransactionManager	9	130	0,09
TopLinkTransactionManager	8	50	0,10
MutablePropertyValues	8	50	0,29
BeanDefinitionVisitor	8	58	0

class public org.springframework.beans.factory.xml.BeanDefinitionParserDelegate
GodClass 5 FeatureEnvy 4 BrainMethod 1 IntensiveCoupling

Object
BeanDefinitionParserDelegate

This is a *God Class* because:

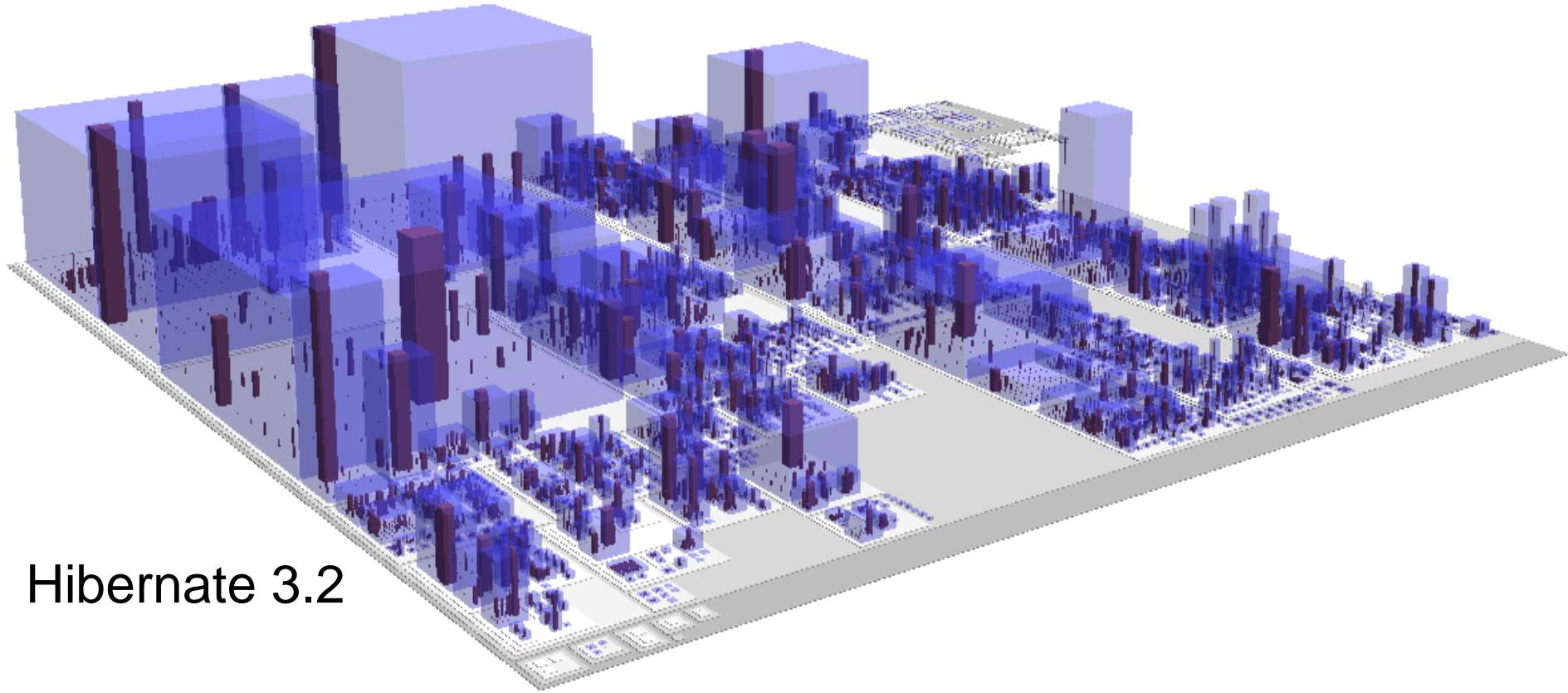
- some of its methods access directly (or via getter/setters) 53.0 attributes from 17 external classes:
 - initDefaultsFeatureEnvy

identity disharmony

collaboration disharmony

classification disharmony

Combining metrics visually



Hibernate 3.2

Combining metrics visually

