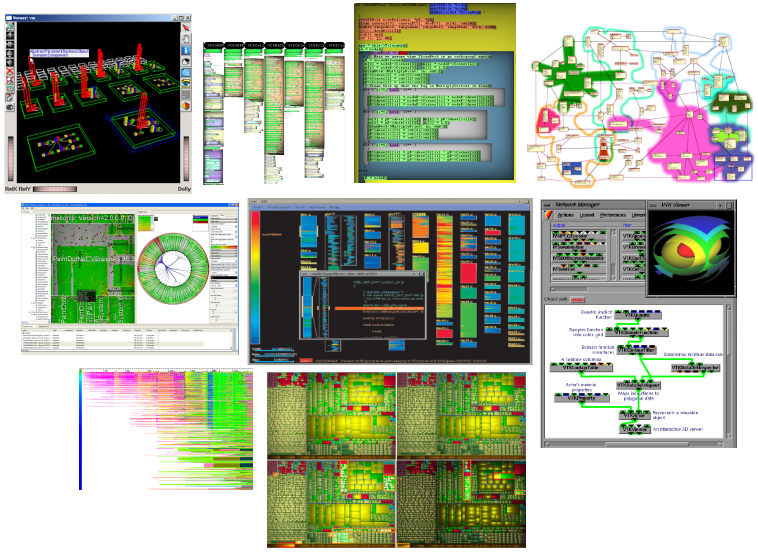
Making sense of software big data
with visual analytics



prof. dr. Alexandru (Alex) Telea

Department of Mathematics and Computer Science
University of Groningen, the Netherlands

Introduction

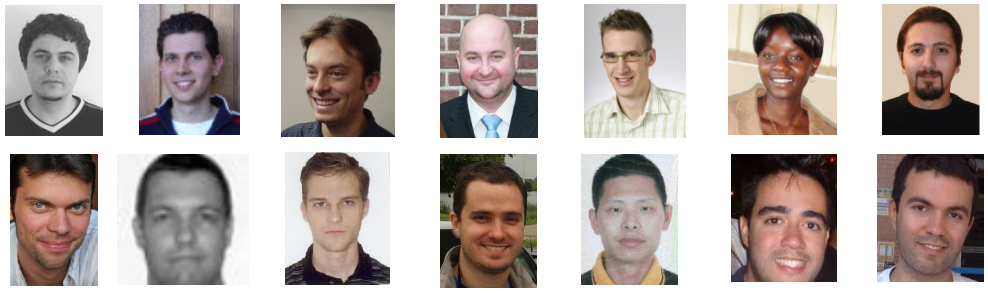
Who am I?

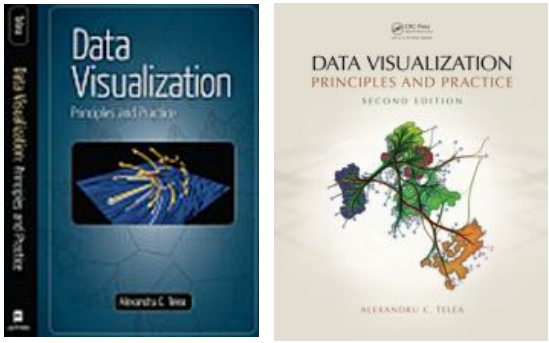
www.cs.rug.nl/~alext

- professor in computer science / multiscale analytics @ RuG (since 2007)
- steering committee chair ACM SOFTVIS / IEEE VISSOFT (since 2007)
- 14 PhD students, 60+ MSc students
- 200 international publications in visual data analytics
- co-founder SolidSource BV

SolidSource IT

www.solidsourceit.com





Data Visualization: Principles and Practice
A. K. Peters, 2008 / 2014

Collaborations & Customers

The Industry…

PHILIPS
sense and simplicity
CREDIT SUISSE
ASML
Continental
PERVASIVE
NXP
making a success of life
momentum
NOKIA
Connecting People
LELY
innovators in agriculture
SBO
SOFTWARE BENCHMARKING
SolidSource IT
software diagnostics
RDW

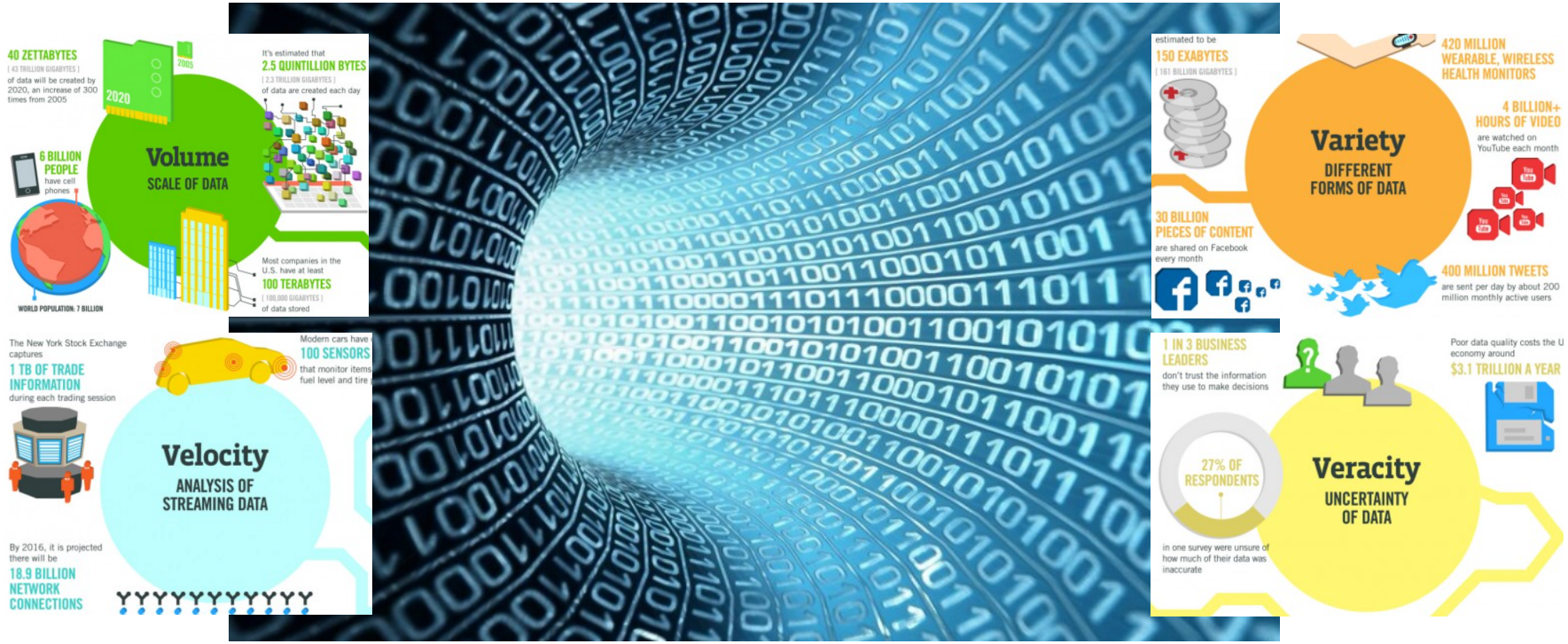
Research/Academia…

HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam
TU/e Technische Universiteit Eindhoven University of Technology
USP Universidade de São Paulo
BRASIL
UNICAMP
UNIVERSIDADE ESTADUAL DE CAMPINAS
IBM Research
dgac
DSNA
DTI Direction de la Technique et de l'Innovation
Laboratoire Bordelais de Recherche en Informatique
UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

…Open Source

OpenCV
KOFFICE
mozilla
Qt

The ‘Big Data’ Challenge – The Four ‘V’ Story

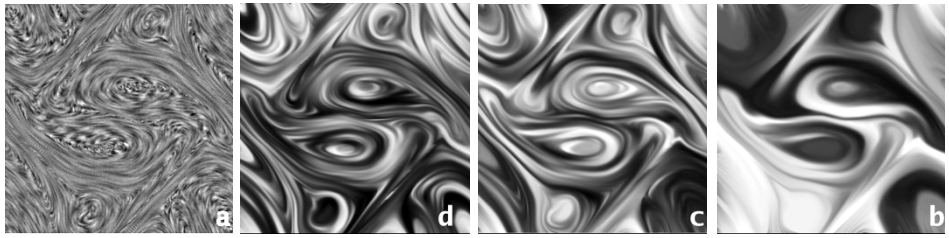


Engineering & Science
FE simulations: 1B elements,
15 DoF/element, 10K timesteps

Information Systems
Data warehouses: 10K tables,
100K rows/table,
200 columns/table

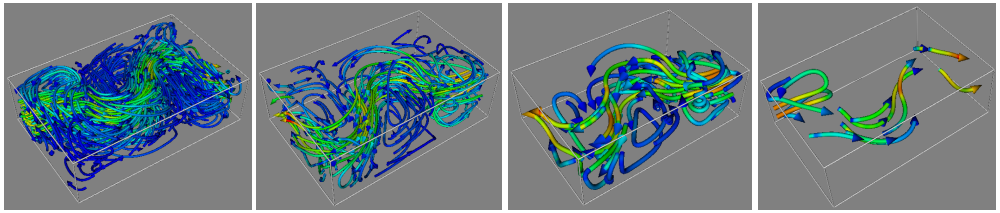
Software Engineering
Code repositories: 10K files,
10K lines/file, 1K versions

Continuous data: The multiscale **solution**

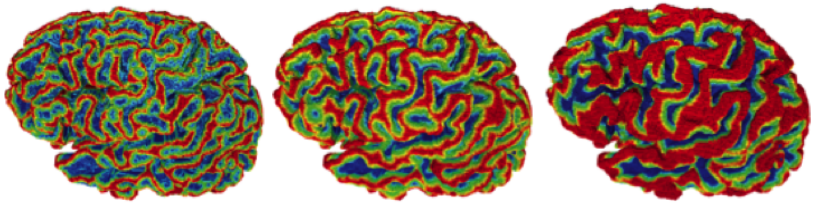


computational flow dynamics

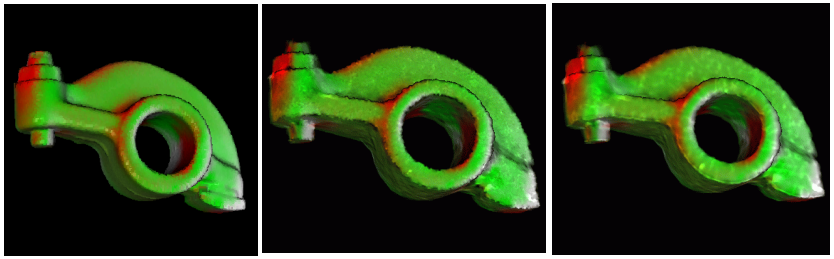
shape segmentation



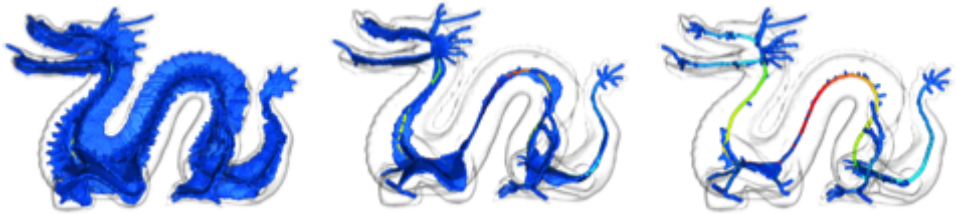
3D vector field analysis



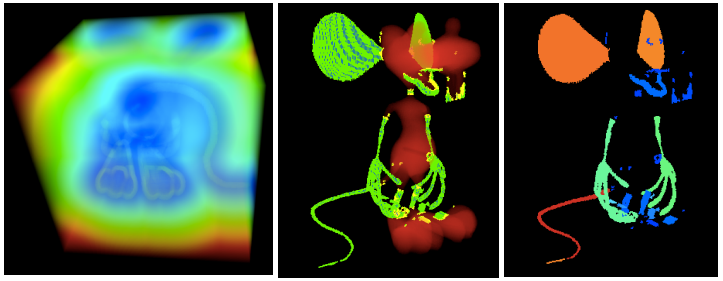
medical shape classification



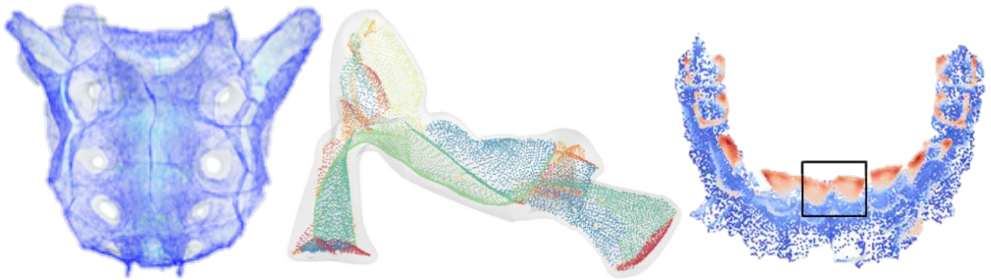
extreme shape simplification



medial descriptors / skeletonization

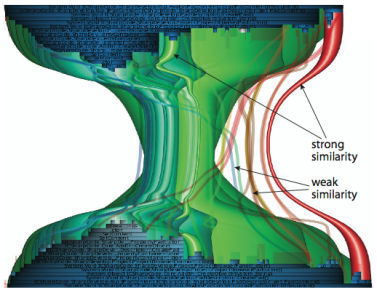


3D metrology

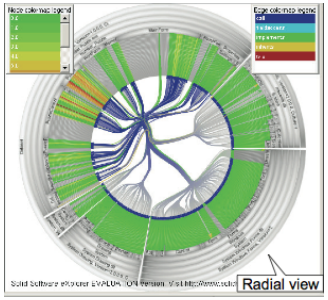


point cloud processing

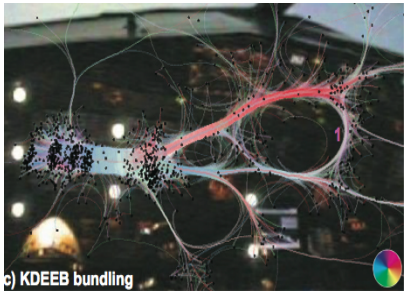
Discrete data: The multiscale **challenge**



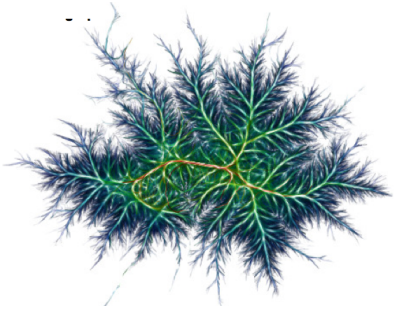
software profiling



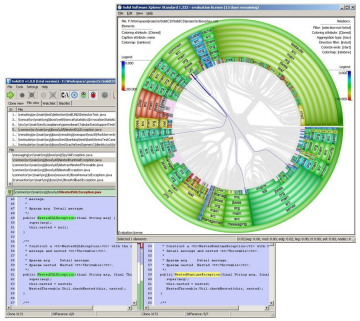
multivariate networks



eye tracking



huge graphs

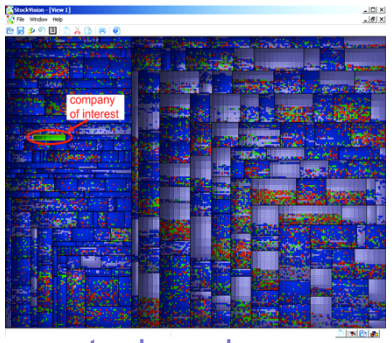


text analysis

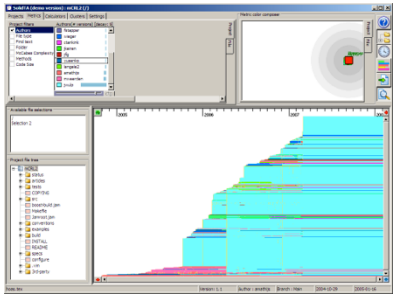
How should we deal with **scale**?

- continuous simplification? How?
- what to simplify exactly?
- how to show simplified results?
- reinvent wheel for each data type?

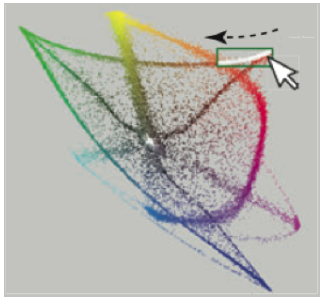
P2P networks



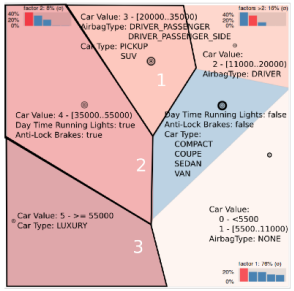
stock exchange



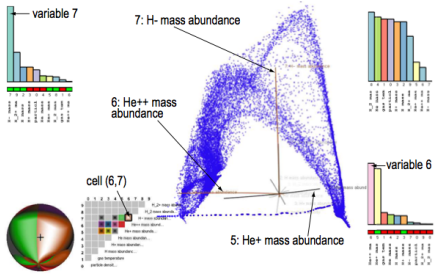
repository analysis



data cubes



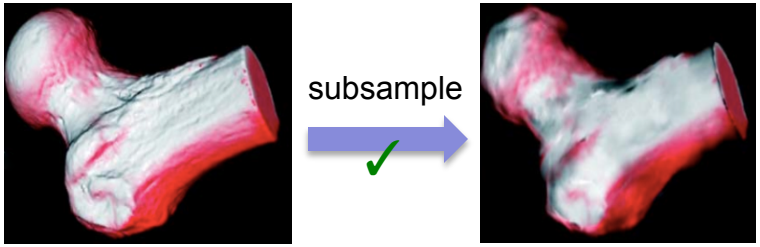
decision support



classifier design

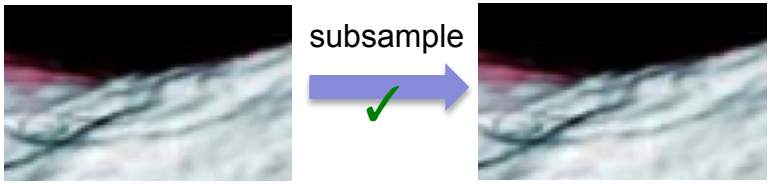
The cause of the problem

Continuous data



bone dataset, 80K points

bone dataset, 20K points



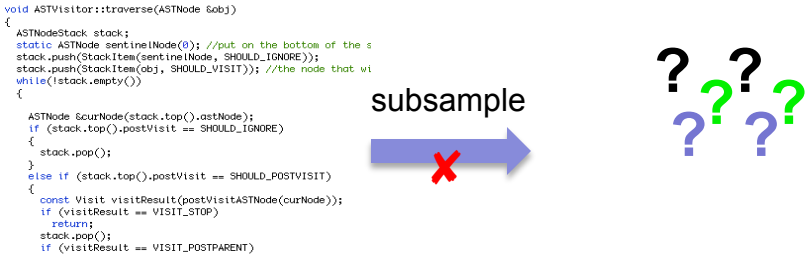
bone detail, 88 polygons

bone detail, 87 polygons

- we throw away 75% of the data
- the semantics stays the same (a bone)

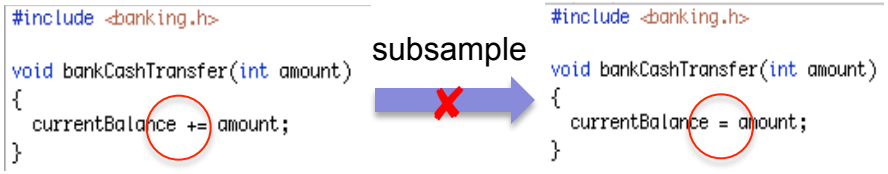
- continuous data: spatial
- attributes: numerical
- interpolation: simple
- resampling: Cauchy-continuous ☺

Discrete data



C++ text, 80K lines

C++ text, 20K lines



C++ text, 88 chars

C++ text, 87 chars

- we throw away one single character
- the semantics becomes completely different

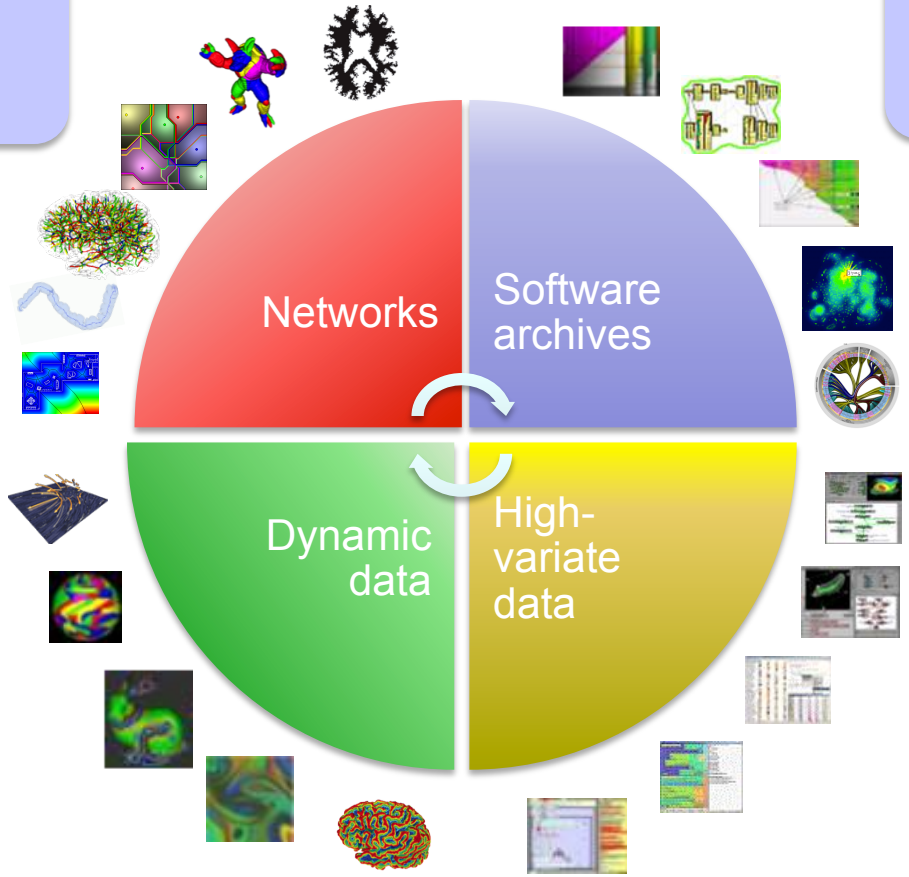
- discrete data: non-spatial
- attributes: any type
- interpolation: often not possible
- resampling: **not Cauchy continuous** ☹

How to treat volume and variety for all data types?

Our answer: Multiscale Visual Data Analytics

GPU Computing

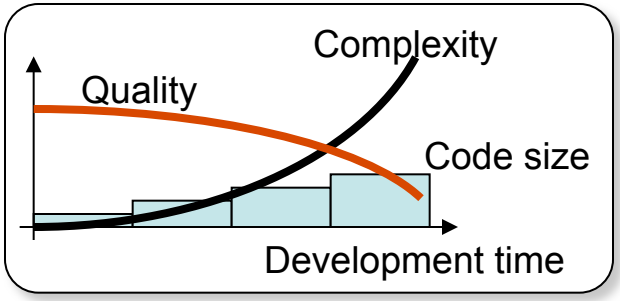
Multiscale methods

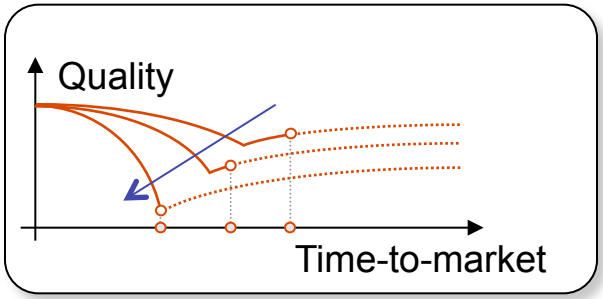


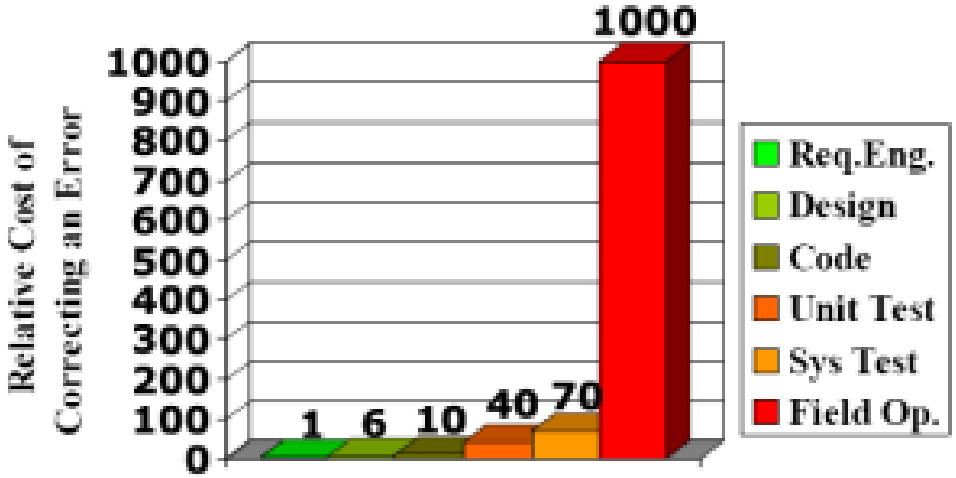
Information visualization

Why Software Engineering?

Software size, team size, and complexity increase → quality decreases
Time-to-market decreases → quality decreases







"The planet has invested so far $2300 billion in maintenance of COBOL code" (Foster, 1991)

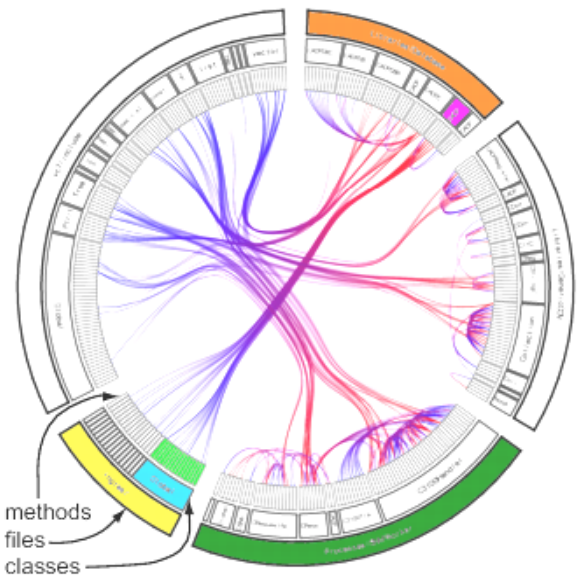
Comparison: Iraq/Afghanistan US war costs: $2700 billion (Wikipedia, 2015)

1. Multivariate dependency networks

How to measure a system's **modularity**?

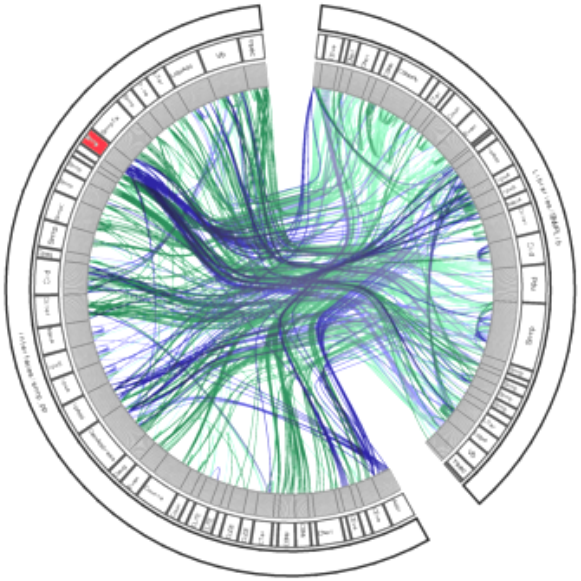
- hard to quantify in a metric…
- …but extremely easy to **see!**

Modular system



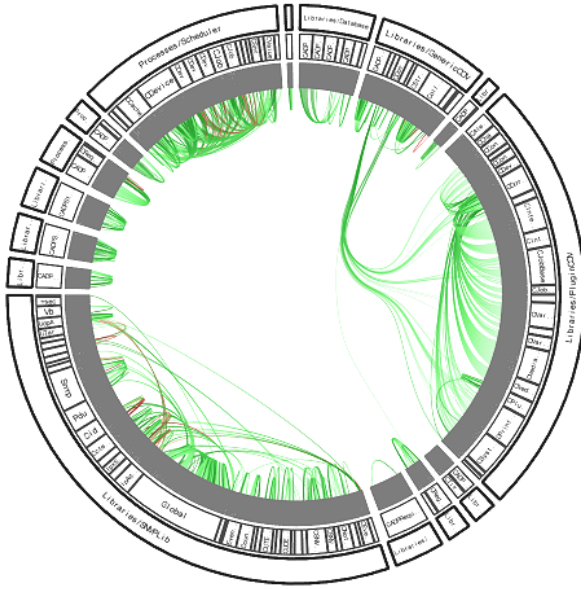
- blue = caller, red = called
- all functions in the yellow file call the purple class
- green file has many self-calls

Monolithic system



- blue = virtual, green = static functions
- red class has many virtual calls (possible interface class)

Decoupled system



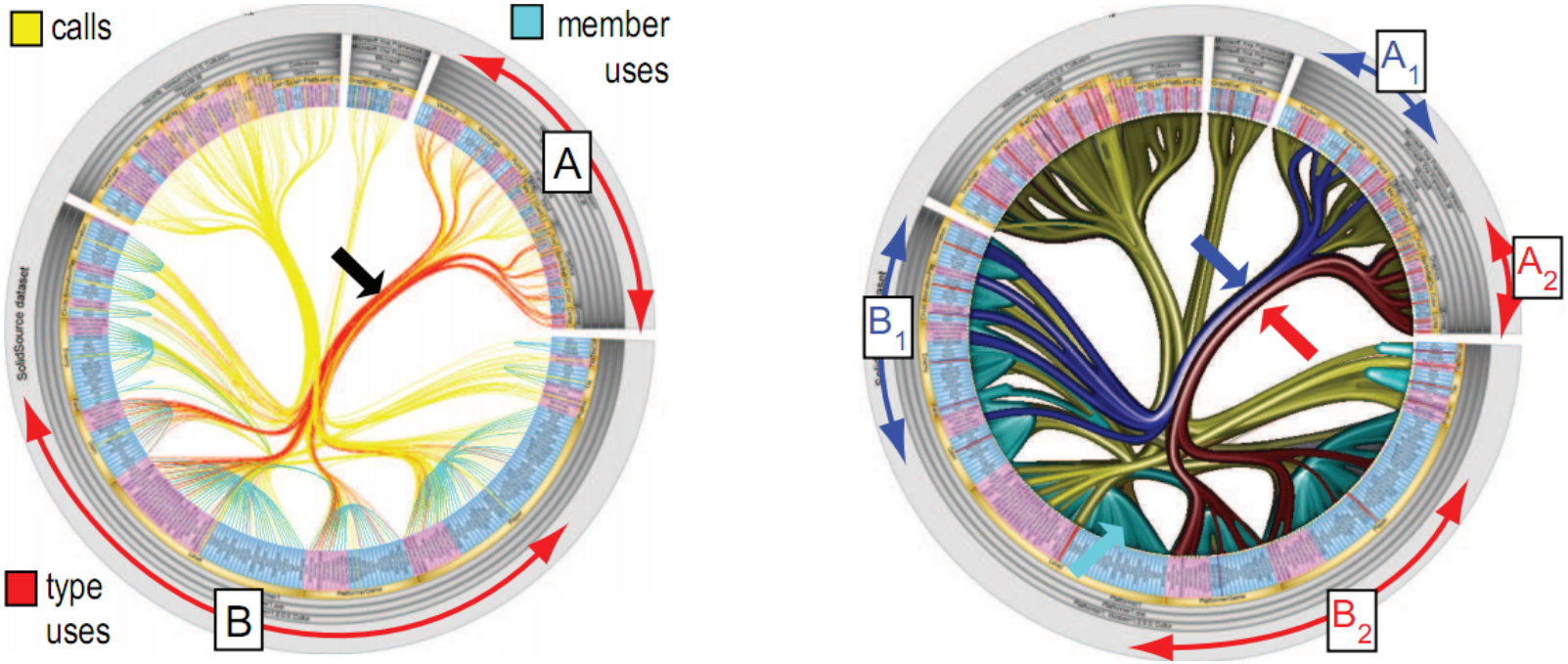
- many intra-module calls
- few inter-module calls
- typical for library software

Images generated with **SolidSX tool** (www.solidsourceit.com)

2. Simplifying network understanding

How to show the **fine-grained** subsystem structure in a network?

• use image-based edge bundles (IBEB)



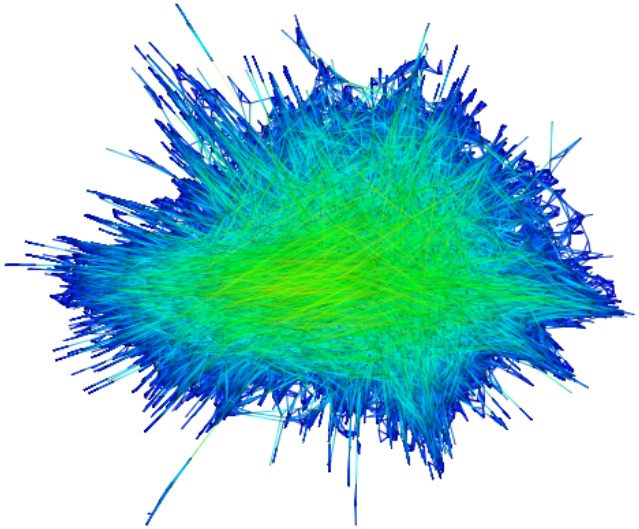
Software system (relations = hierarchy + dependencies)

- classical method: shows coarse system-system relations only
  - A depends on B, but *how*?
- IBEB shows finer-scale subsystem relations
  - A1 depends on B1, A2 depends on B2

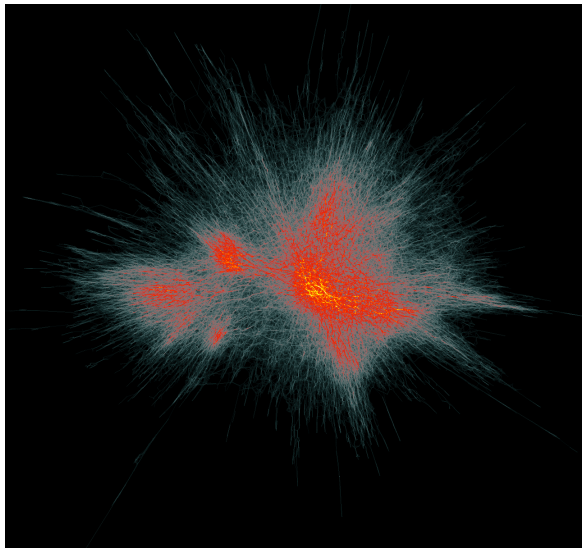
3. Understanding very large networks

Given a very large relational dataset, how to show **structure**?

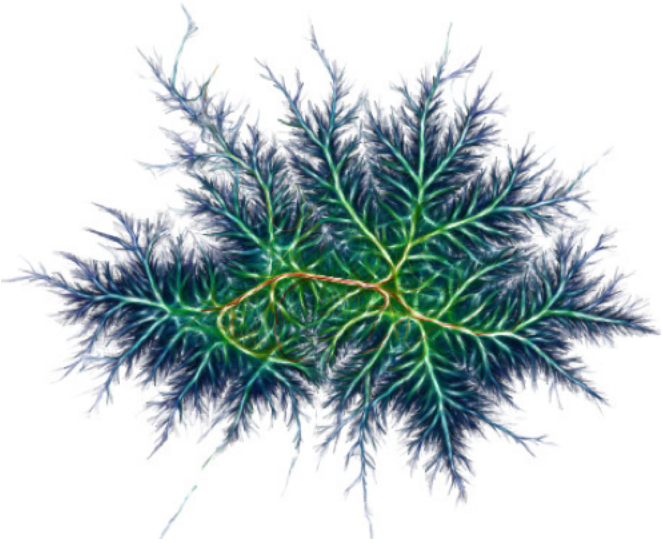
- use-case: Amazon transactions (~1M nodes/edges)
- extract/show the main connection patterns in **real time**
- graph **skeleton:** main groups of related transactions



standard graph drawing



MINGLE bundling (state-of-the-art, 2012; several minutes on a standard PC)

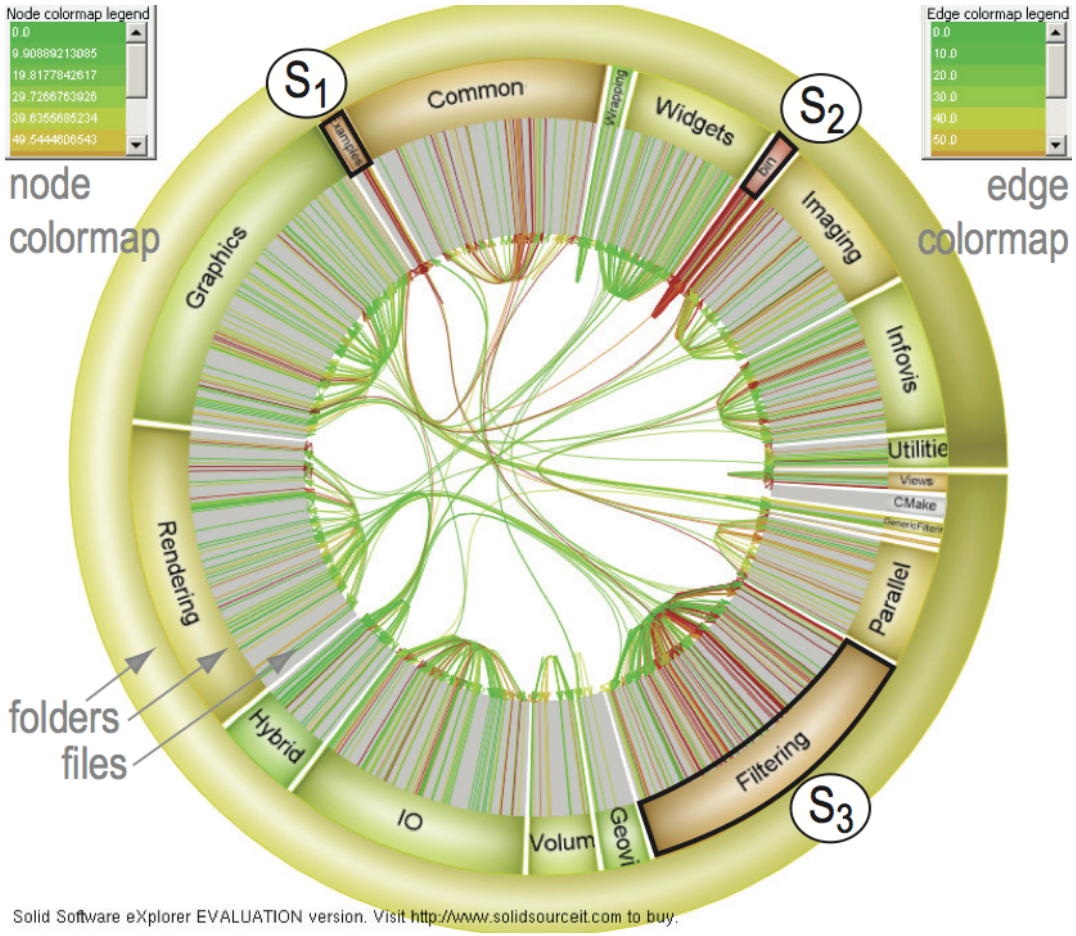


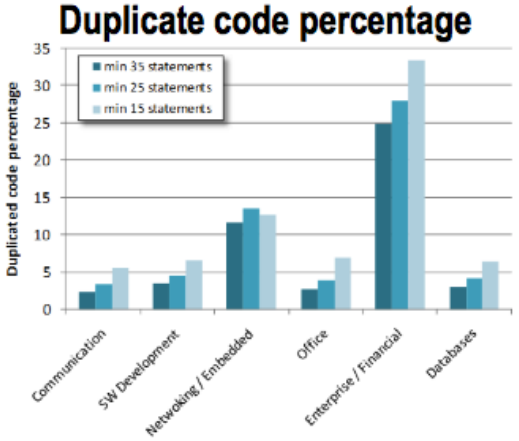
our CUBu bundling (2014): real-time (<1 second on a standard PC)

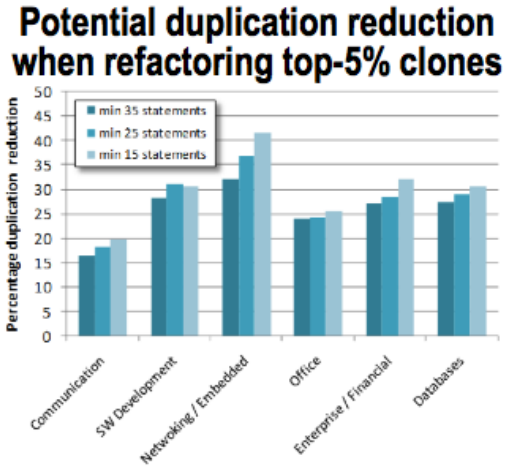
4. Duplication detection

How to show **duplication** in a 5M lines code archive?

- useful to detect wasted effort, lack of modularity, copy-paste
- but also recurrent patterns (e.g. in text archives)





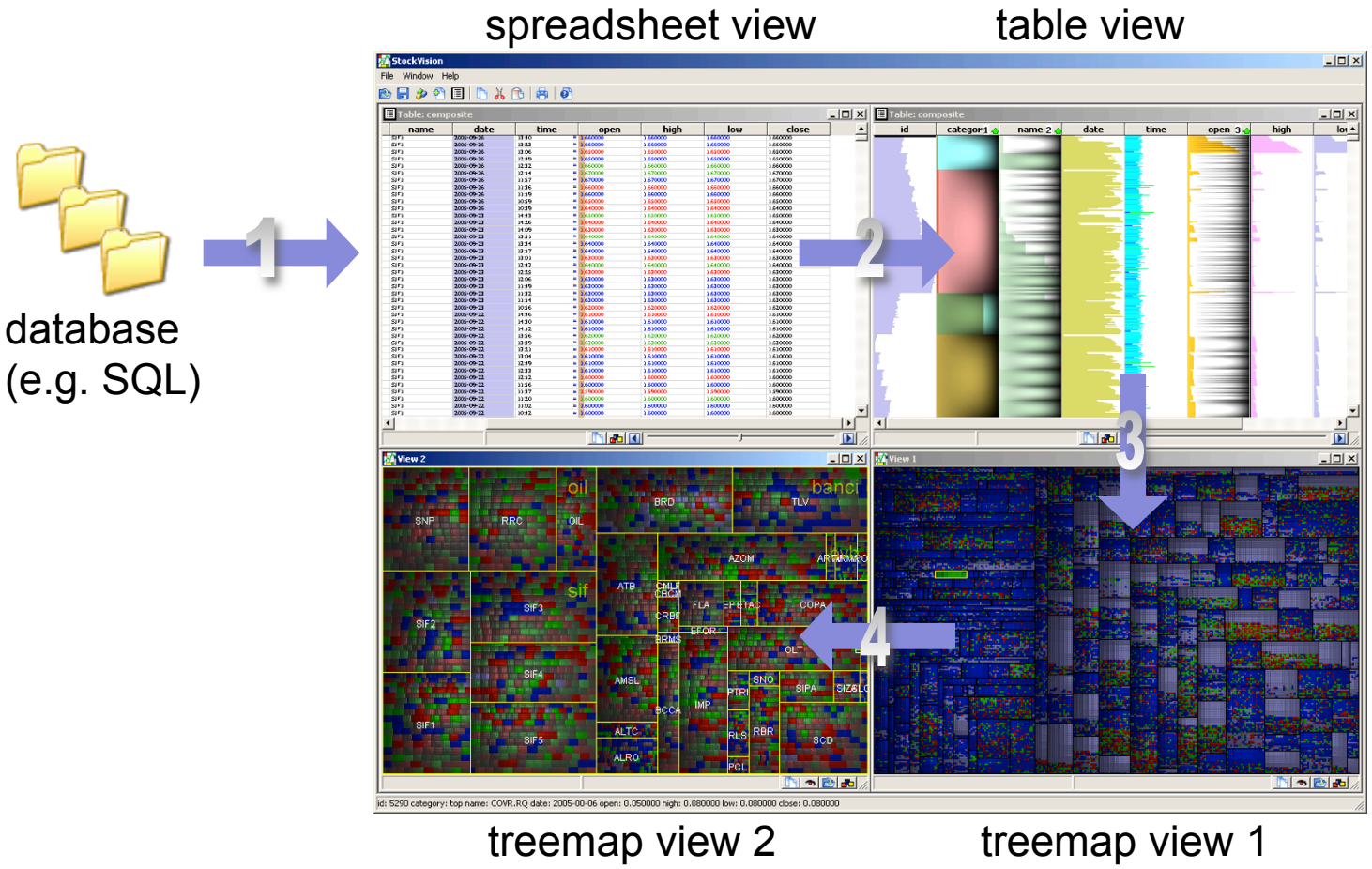


Images generated with **SolidSDD tool** (www.solidsourceit.com)

5. Discovering data outliers

Given a 1M record database, how to find **outliers**?

- use **TableLens** to find trends
- use **treemaps** to find correlation of trends with categories

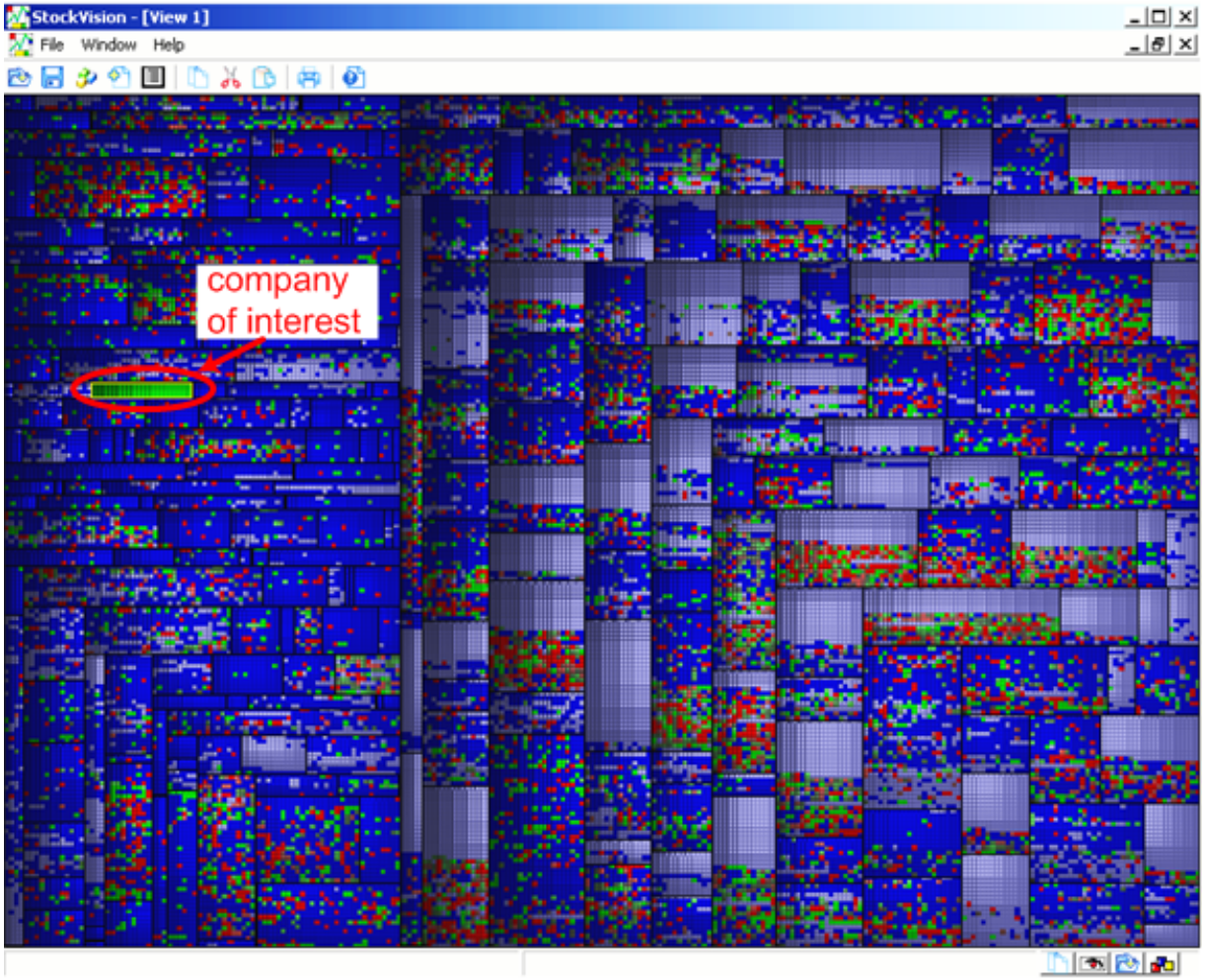


Images generated with **TableVision tool** (www.cs.rug.nl/svcg)

6. Discovering outliers (cont'd)

Example: Find **outliers** in a 4M stock-transaction database

- RASDAQ emerging stock market (customer: InterCapital SRL)
- discover small-cap growing company (good investment)



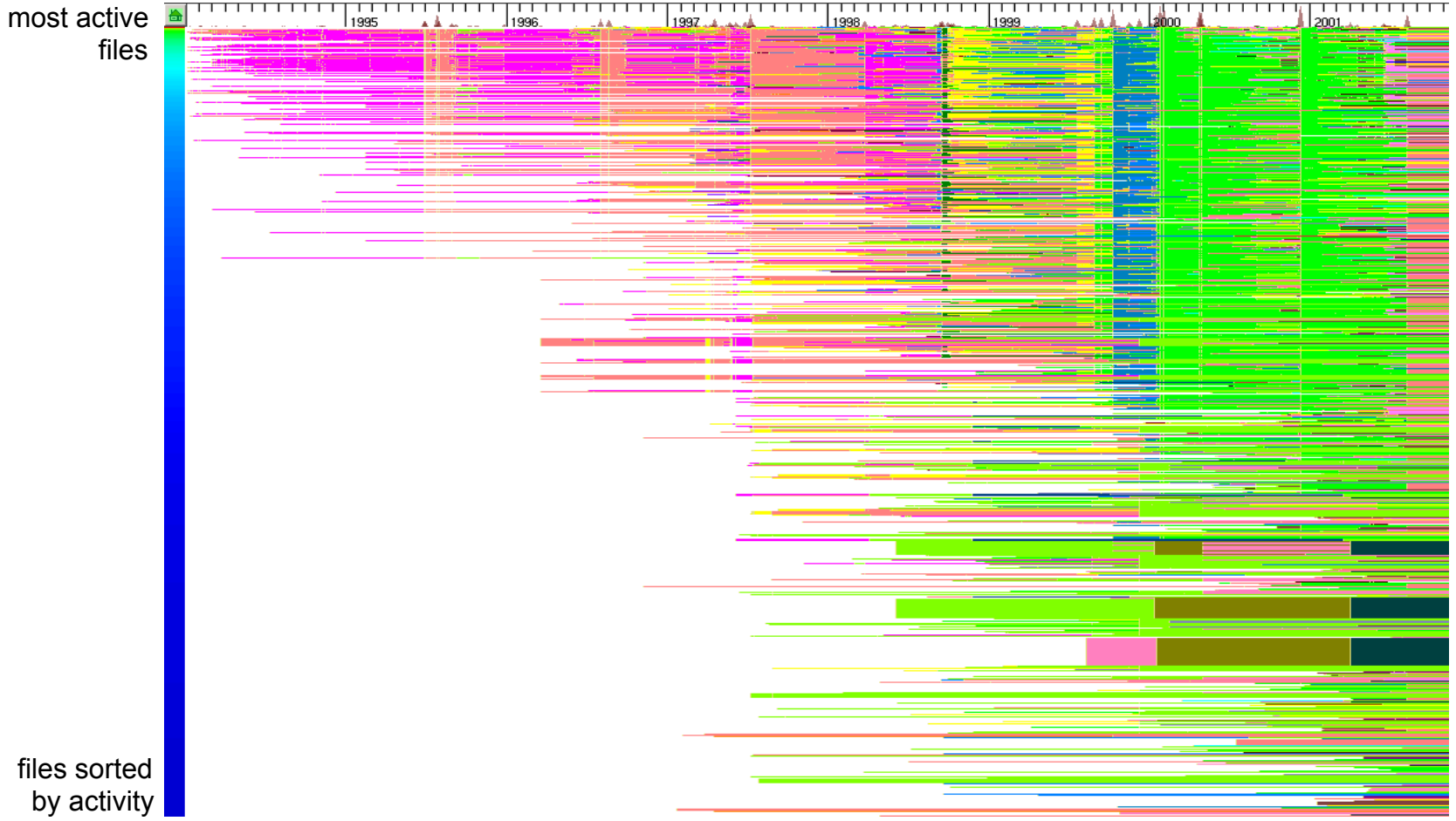
Images generated with **TableVision tool** (www.cs.rug.nl/svcg)

7. Understanding large data archives

Given a 10-year archive (50K files, 200K changes), how to spot **trends**?

- software maintenance (Philips, Bosch, Continental, ASML)
- financial/healthcare data (Pervasive, Momentum, Credit Suisse)

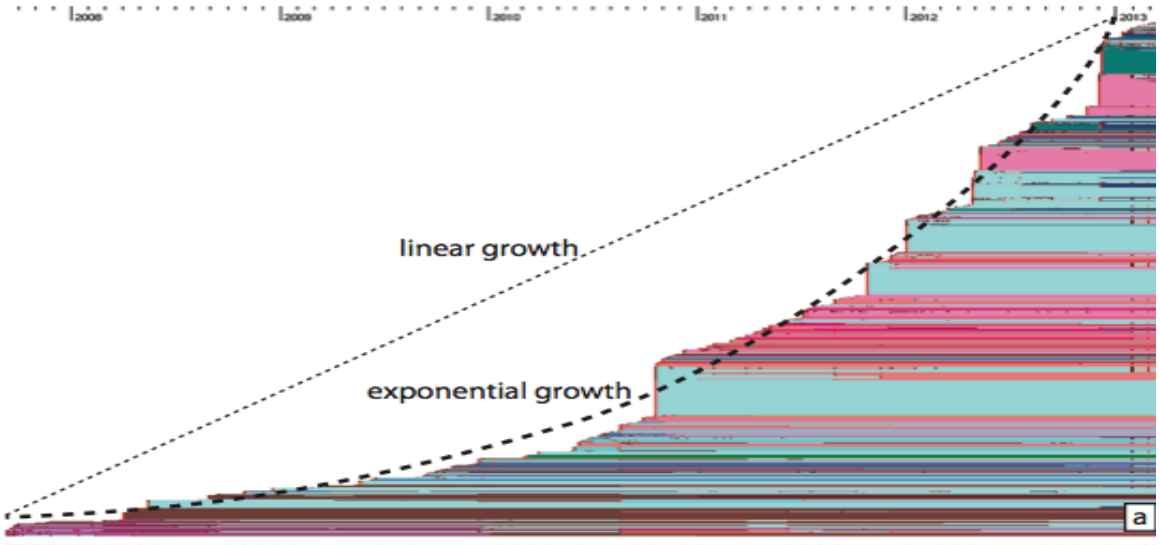
Trend Analyzer:

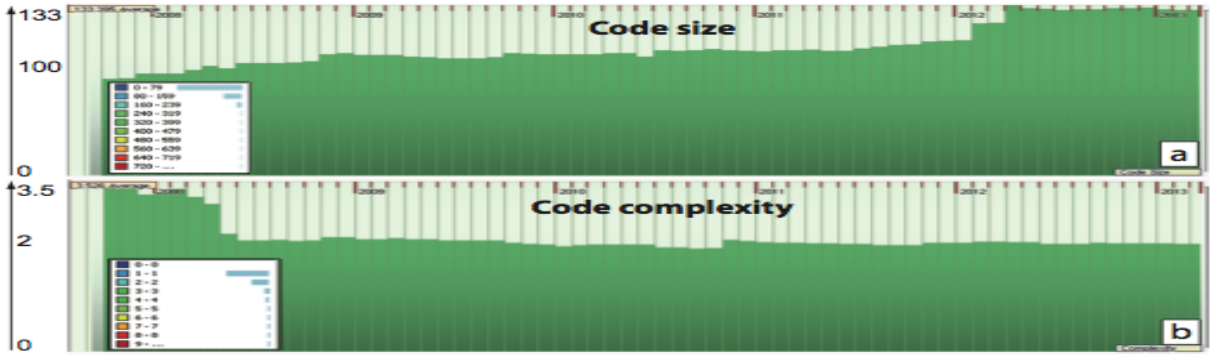


Images generated with **SolidSTA tool** (www.solidsourceit.com)

8. Understanding large data archives (cont'd)

Use-case: Does software increase cause quality problems?



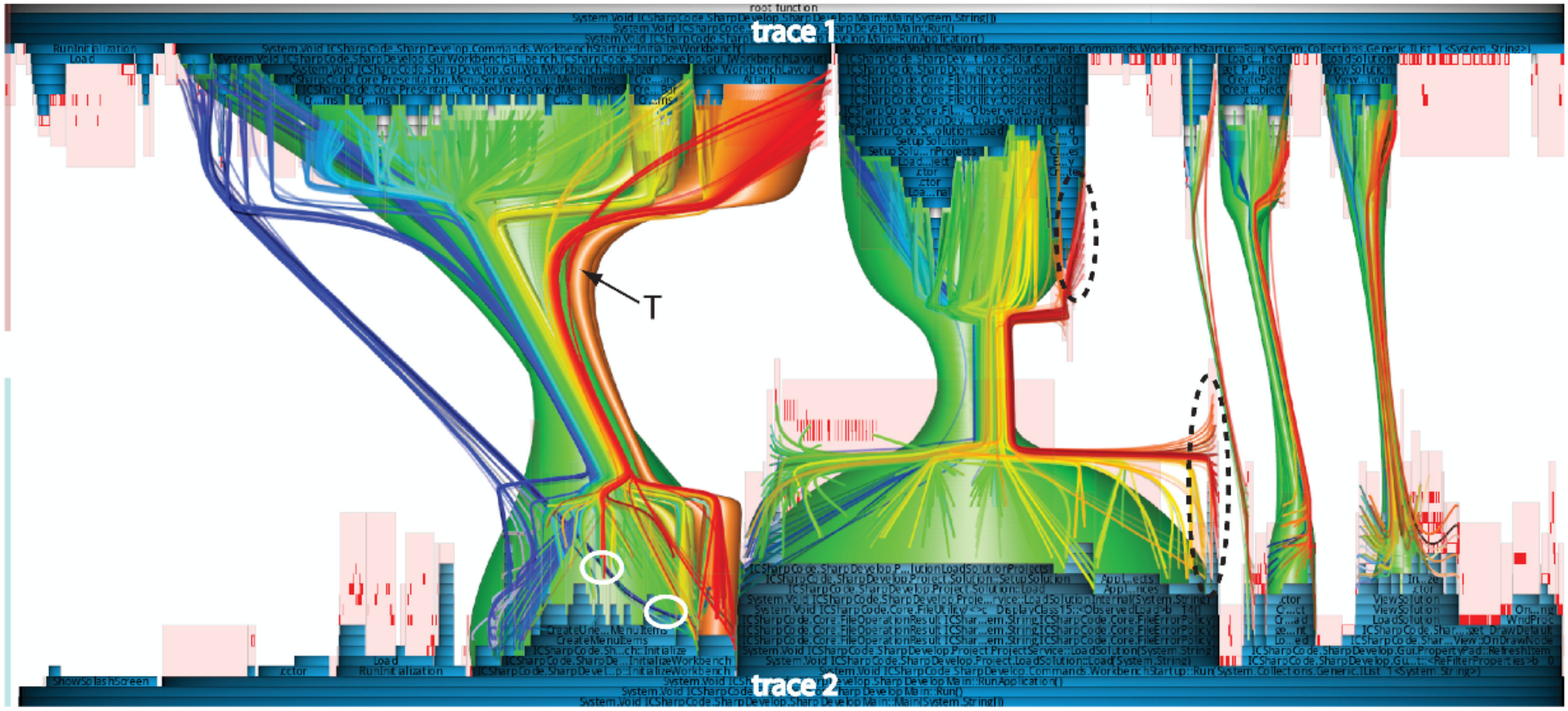


Images generated with **SolidSTA tool** (www.solidsourceit.com); customer: RDW

9. Dynamic traces

Given several executions of a program, how to spot **differences**?

- used for finding performance/quality problems (**Deutsche Post**)
- trace differences = potential problems

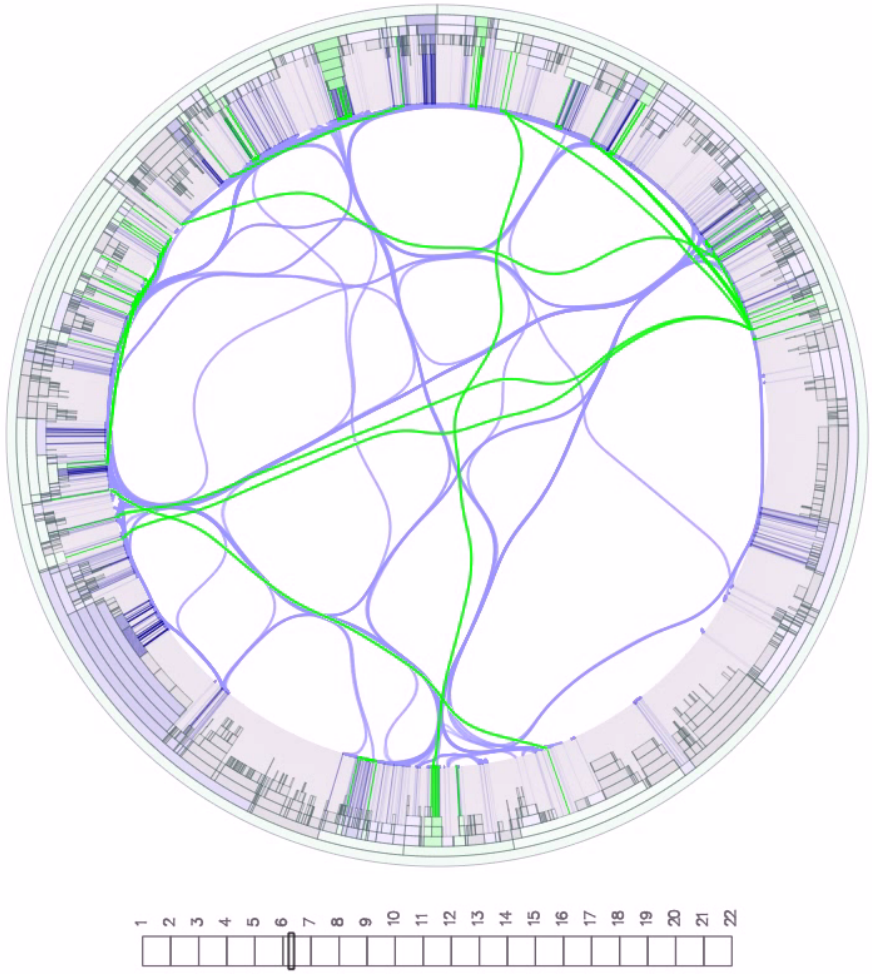


Images generated with **TraceDiff tool** (http://www.hpi.uni-potsdam.de)

10. Dynamic networks

How to show **changes** in a network?

• software dependencies, customer interactions, transport networks, …

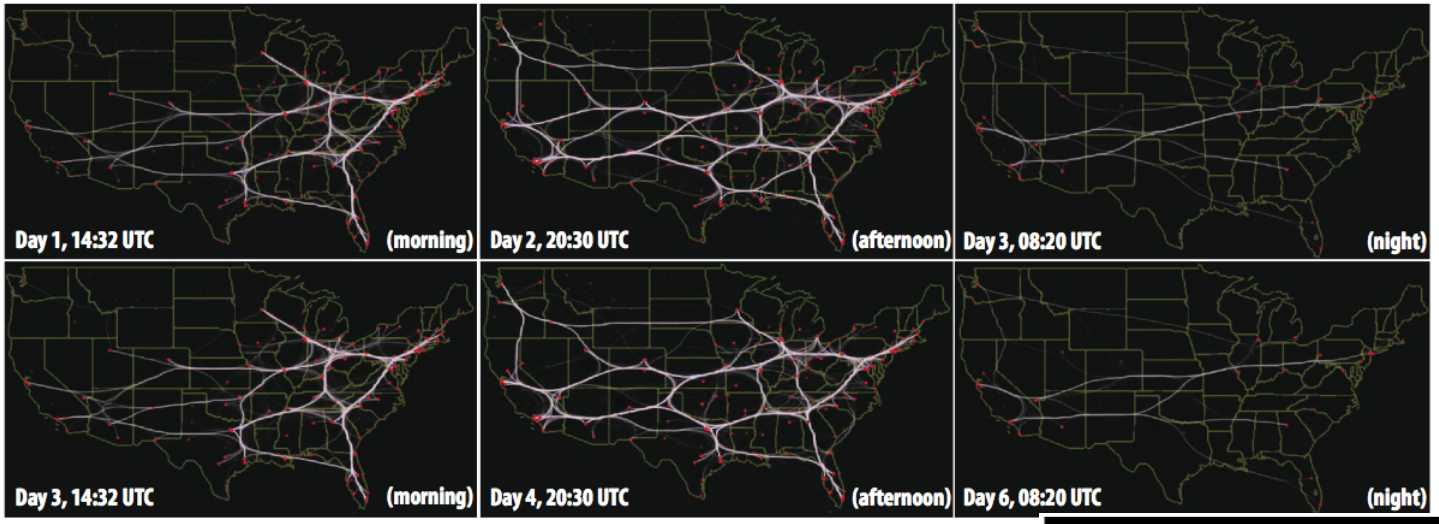


Evolution of clones
in Mozilla Firefox (2004-2011)

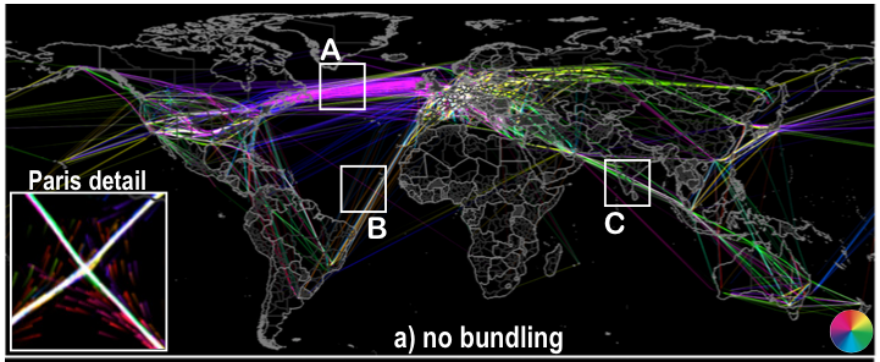
10. Dynamic networks (cont'd)

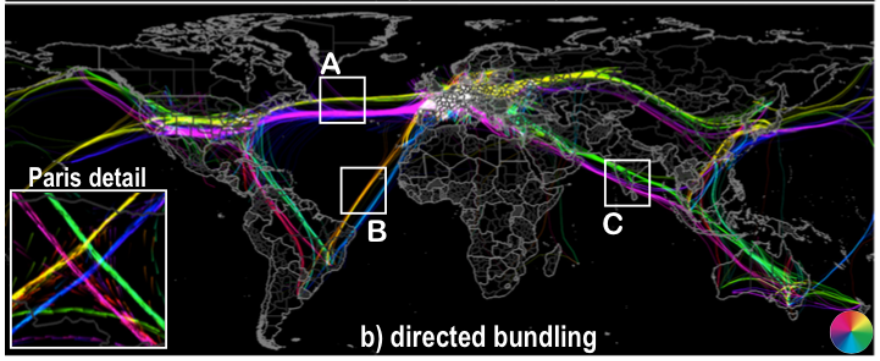
How to show changes in a transportation network?

• use kernel-density bundling (*aka* mean-shift clustering in image segmentation)



US flights (Aug 2008)
(~20K flights)



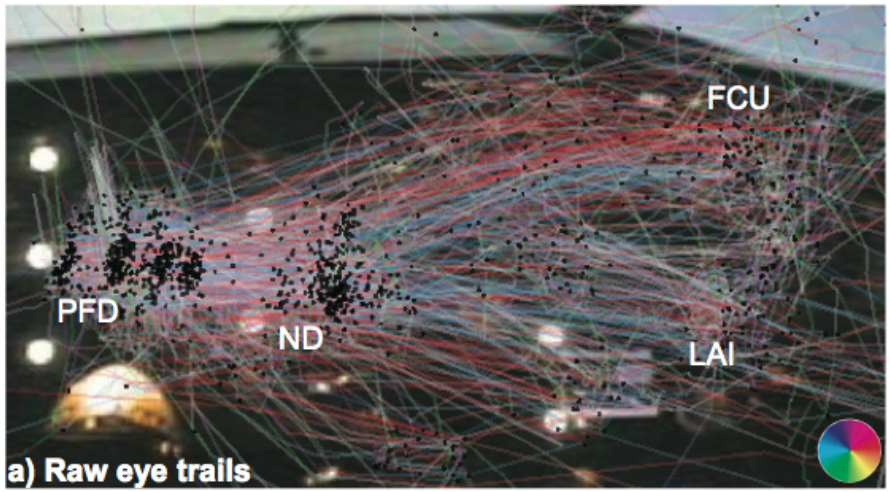


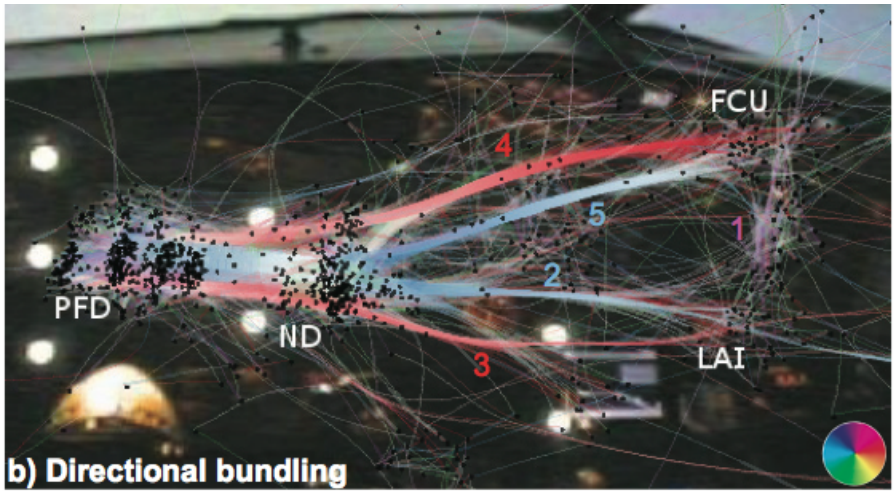
World flights (June 2013)
(~1M flights)

11. Eye-gaze trails

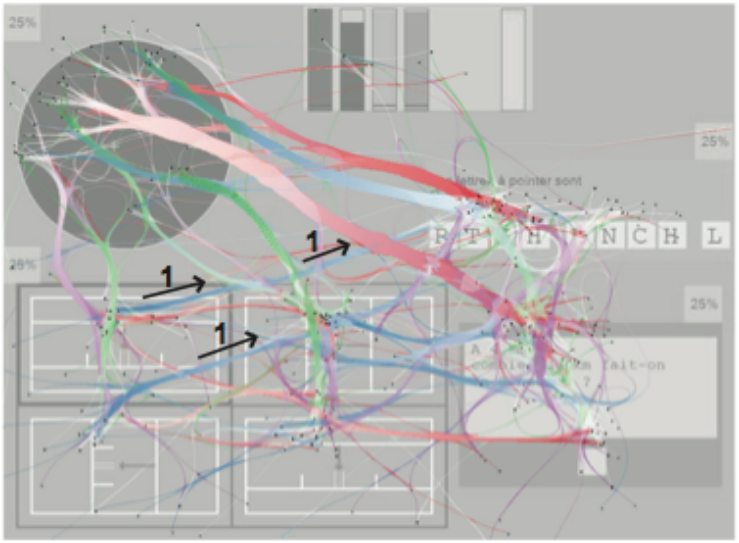
How to analyze the **behavior** of a human subject looking at visual stimuli?

- evaluate/optimize user-interface design for highly-critical devices (e.g. aircraft, surgery)
- use **trail bundling** of the eye-gaze tracks (recorded by an eye tracker)





Data source: ENAC pilot training academy, Toulouse, France



Data source: ATC national authority, Paris, France

Application: Post-Mortem IT System Assessment

Context

- client: top-three automotive IT producer (Germany)
- product: 8 years evolution (2002-2008)
  - 3.5 MLOC code, C166 dialect (1881 files)
  - 1 MLOC headers (2454 files)
  - 15 releases
  - 3 teams = ~60 people (2 x EU, 1 x India)
- product failed to meet requests, at end

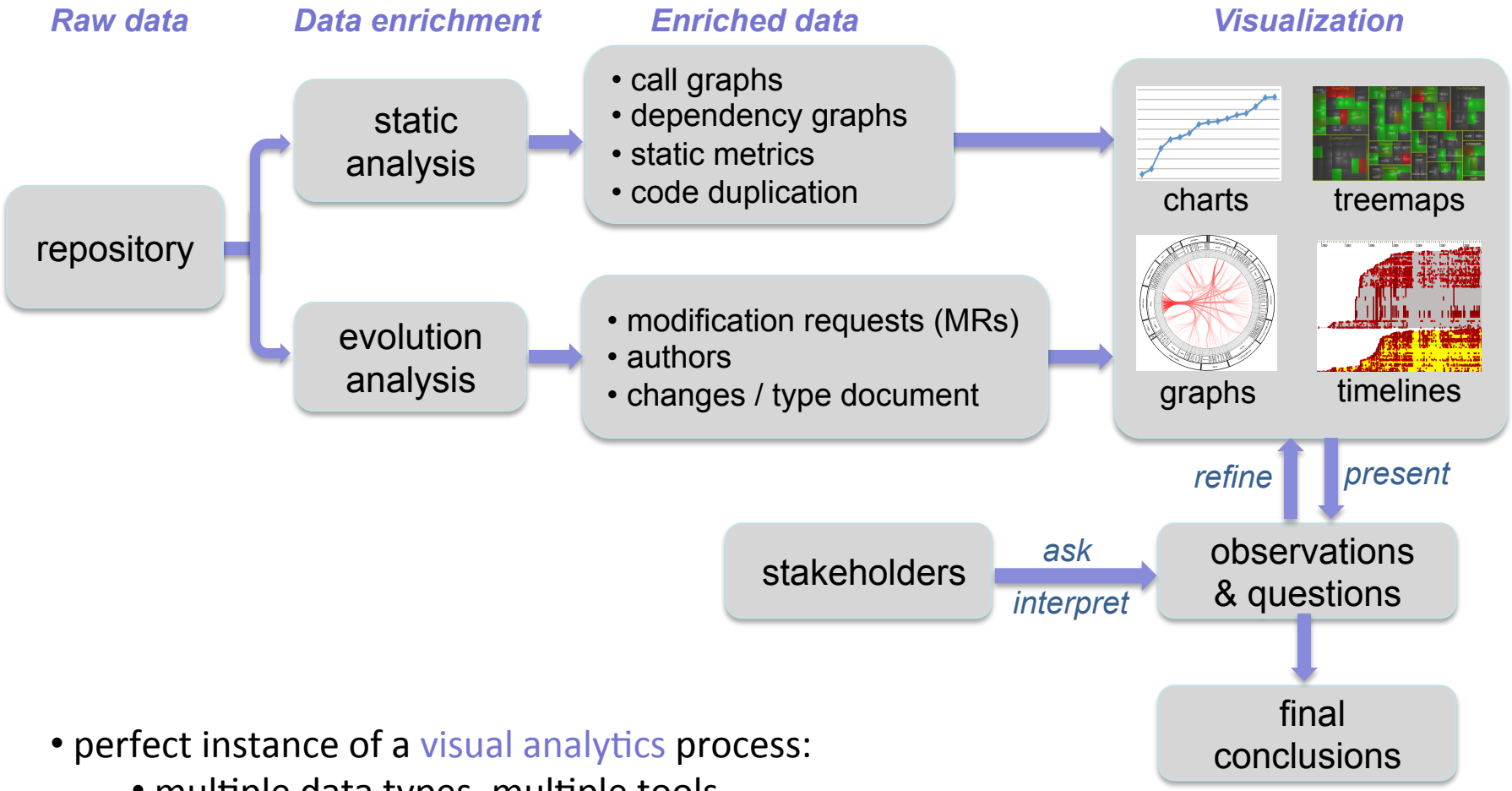
Questions

- what happened right/wrong?
- how to prevent such errors in the future?

Constraints

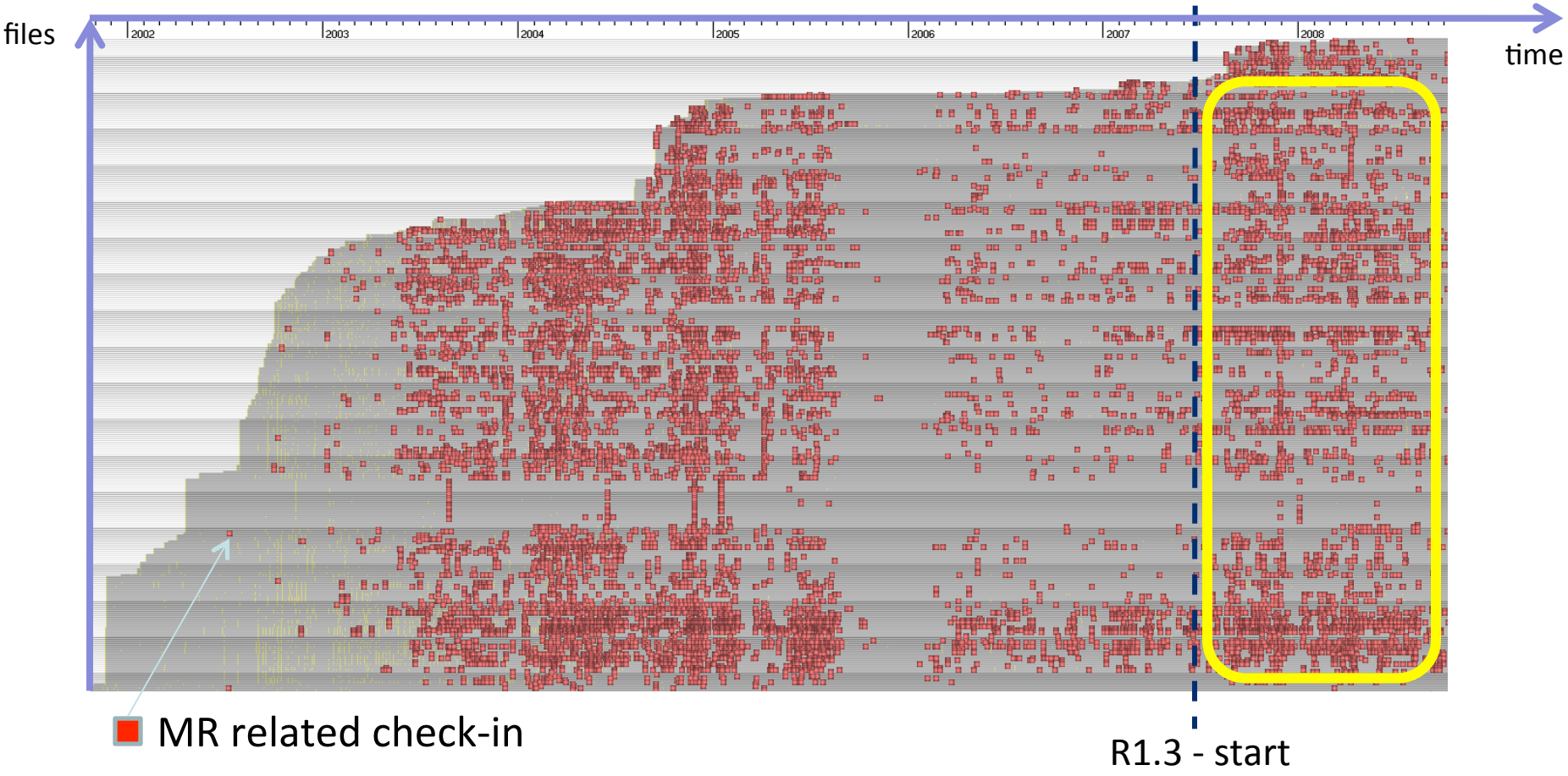
- data: source code repository only
- time: answers needed in **max. 1 week**
- we were unfamiliar with the application

Methodology



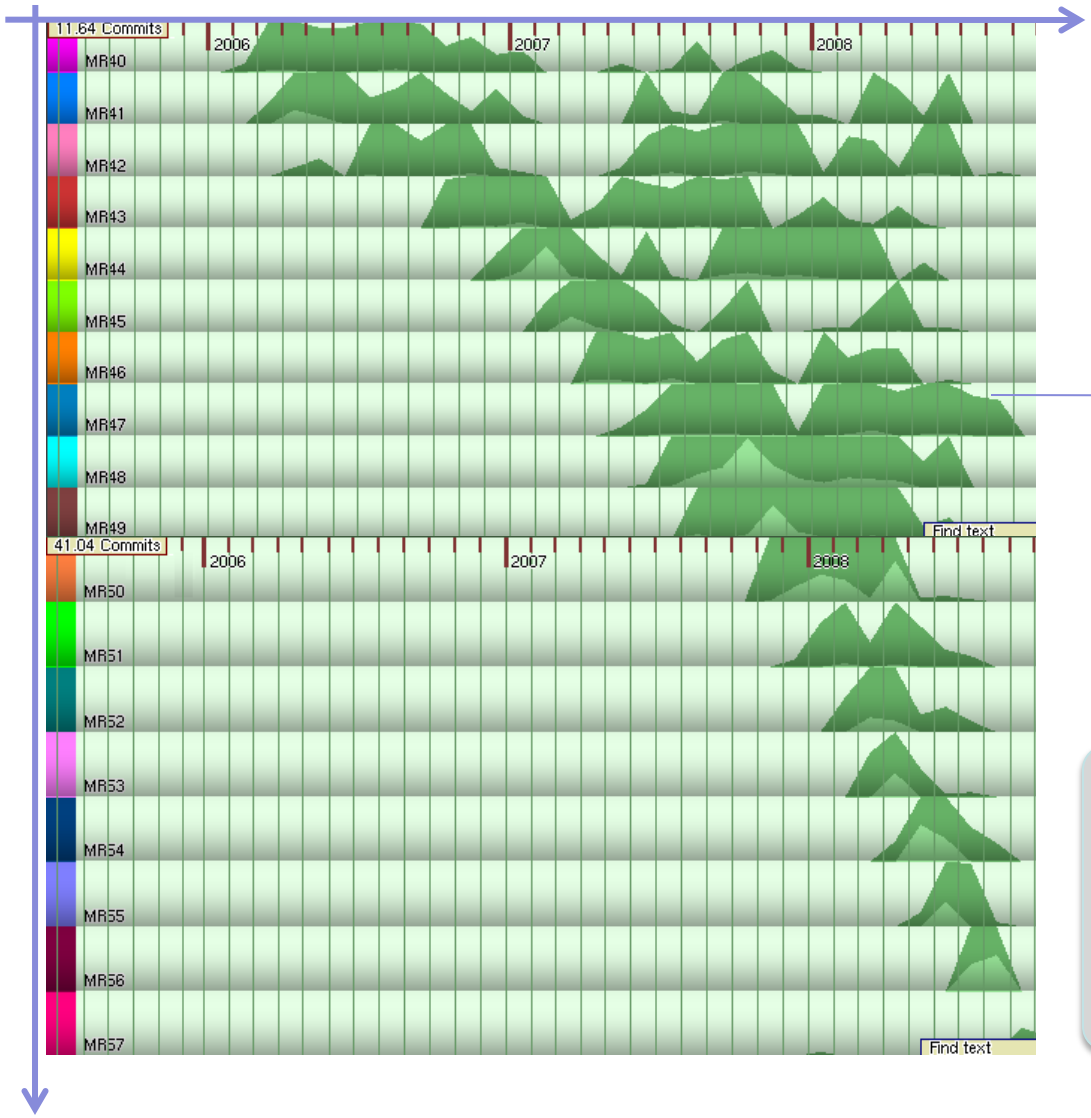
- perfect instance of a visual analytics process:
  - multiple data types, multiple tools
  - tight combination of data extraction, processing, visualization
  - incremental *hypothesis refinement*

Requirements: MR Duration



little increase in the file curve – most activity in *old* files suggests too long maintenance & closure of requirements

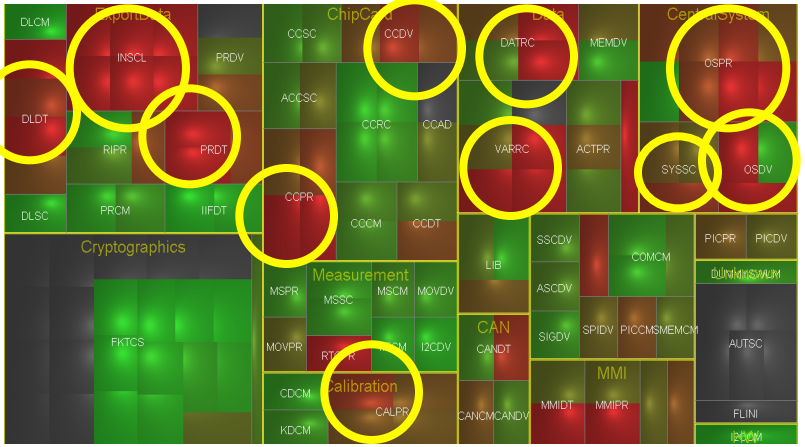
Requirements: MR Duration

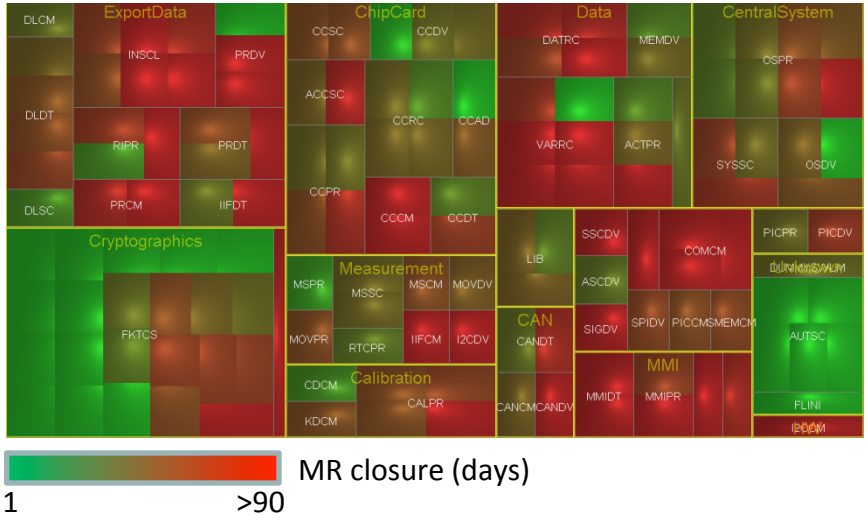


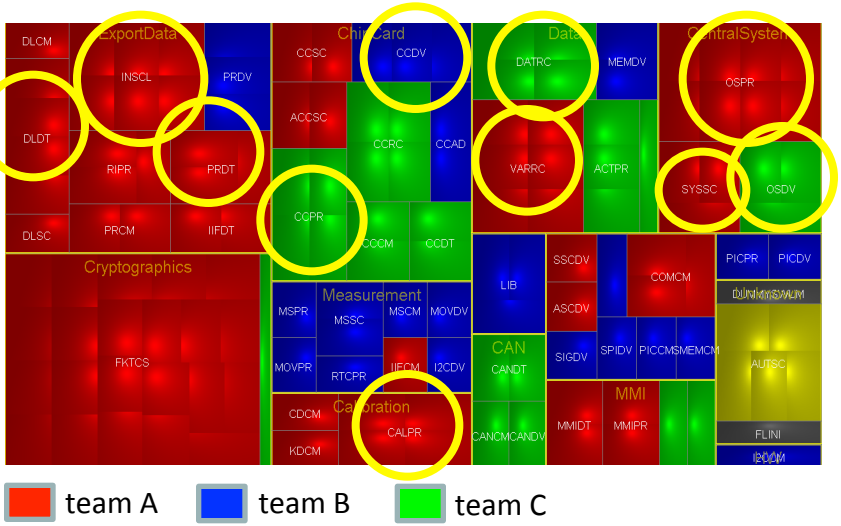
graph: # commits referring to MRs within a given id range

in mid 2008, activity related to MRs from 2006 still takes place

Team: Code Ownership

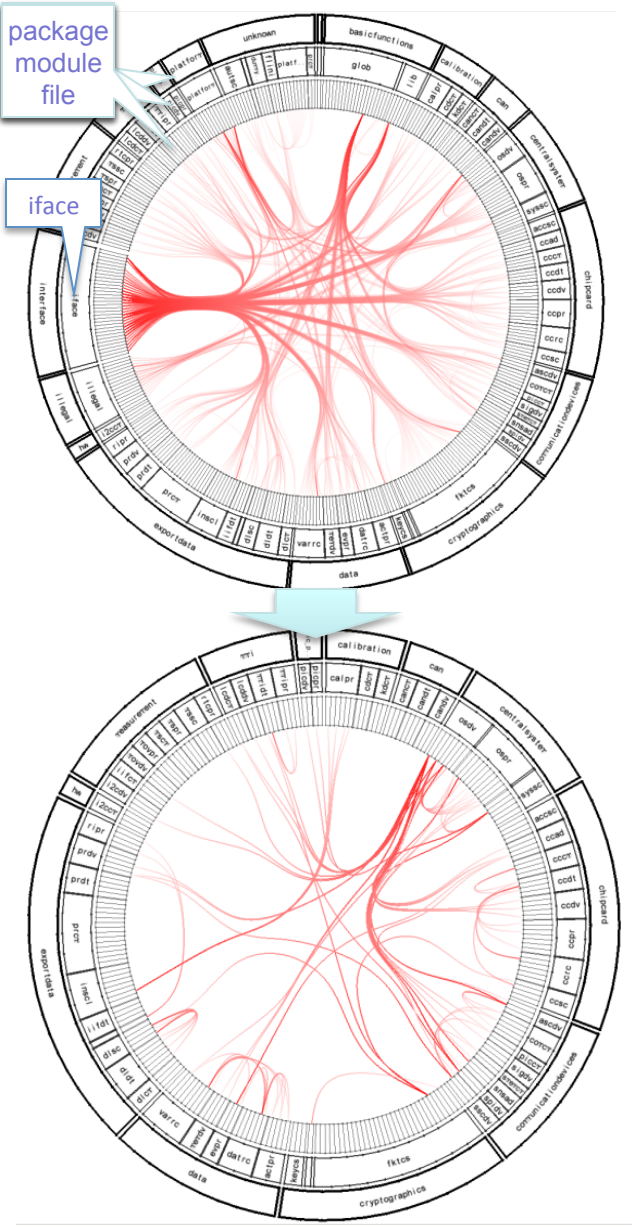






large part of software affected by long open-standing MRs
Most of these are assigned to team A (largest team)…
…and this team was reported to have communication problems!

Code: Dependencies



uses = call, type, variable, macro, …

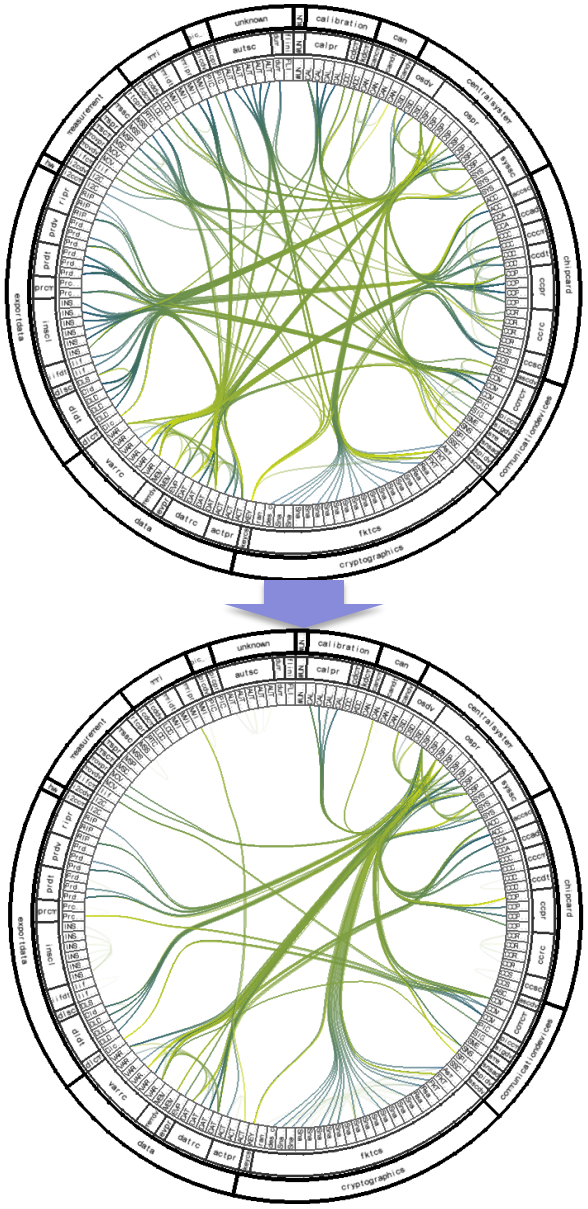
is used

Most dependencies occur via the iface, basicfunctions and platform packages

Filter out these allowed dependencies…
…to discover *unwanted* dependencies

These are accesses that bypass established interfaces

Code: Call graph

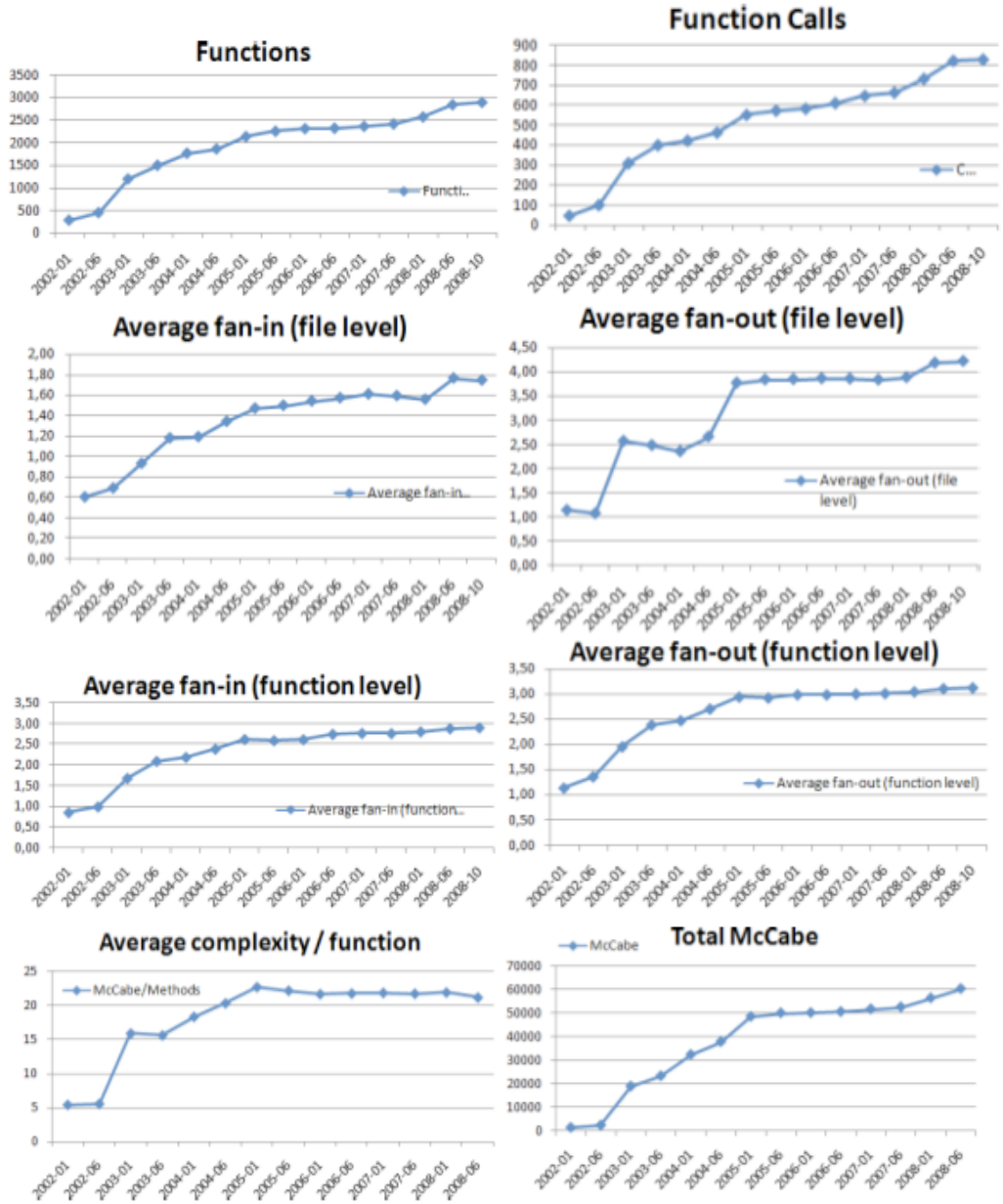


High coupling at package level
This image does not tell us very much

Select only modules which are *mutually call dependent*…
…to discover *layering violations*

Not a strict layering in the system (as it should be)

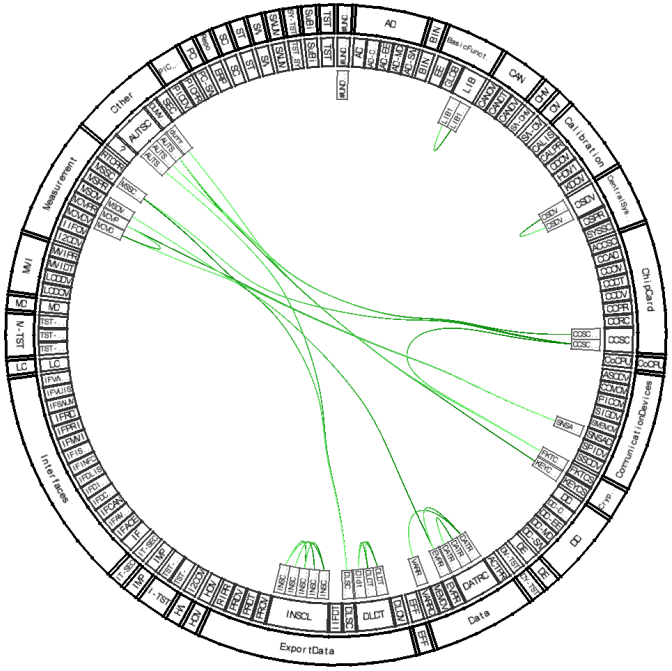
Code: Quality Metrics



Moderate code + dependency growth
- does not explain product's problems

Average complexity/function > 20
Total complexity: up 20% in R1.3
- testing can be hard!
- possible cause of product's problems

Code: Duplication



External duplication

- links: modules that contain similar code blocks of >25 LOC

duplicated blocks

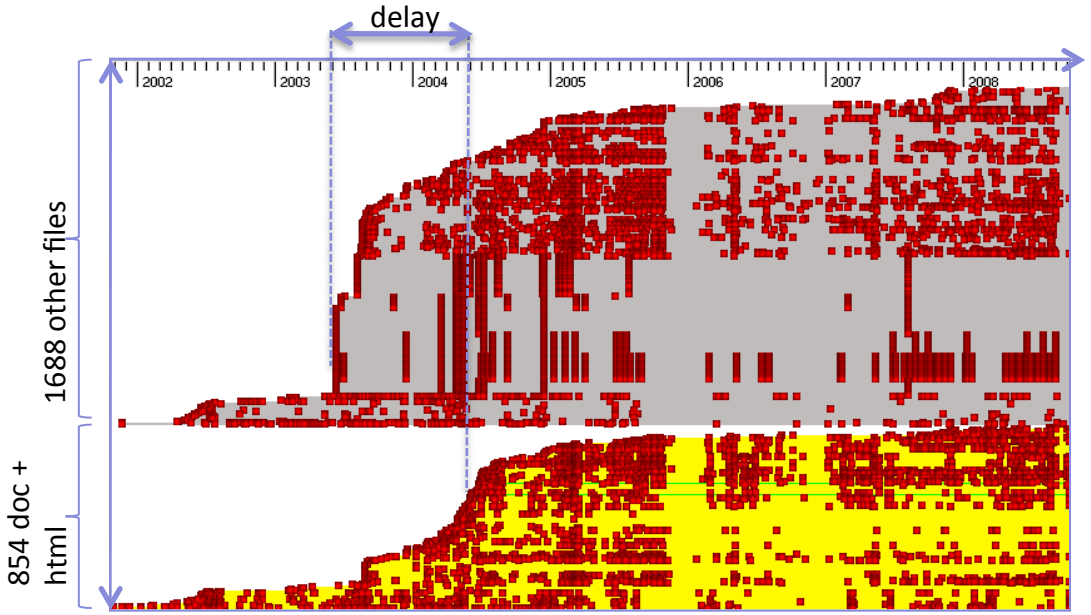
1 60

Internal duplication

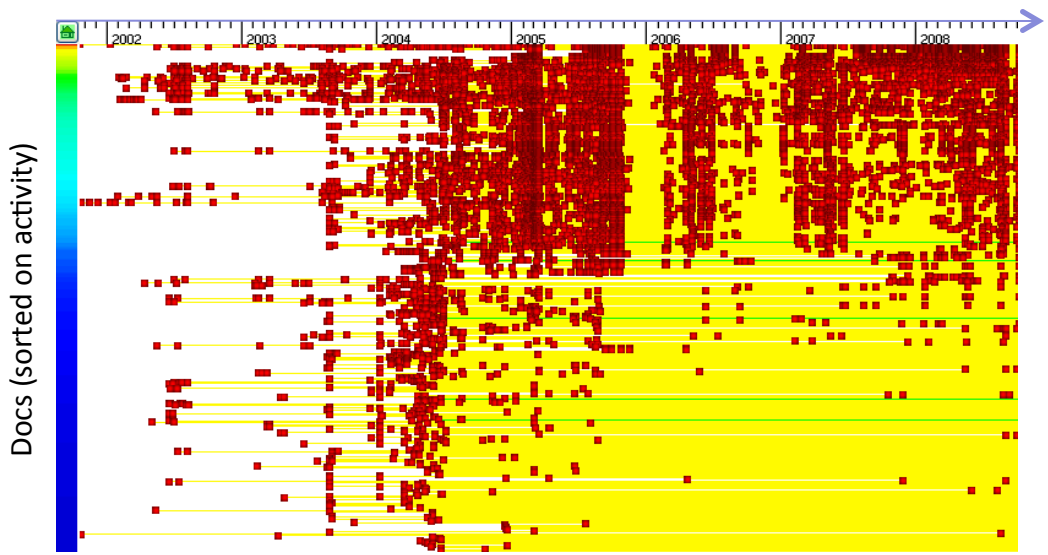
- color: # duplicated blocks within a file

Little external/internal duplication
Arguably not a problem for testing

Documentation



- 30% of files are documentation
- updated regularly
- grow in sync with rest of code base

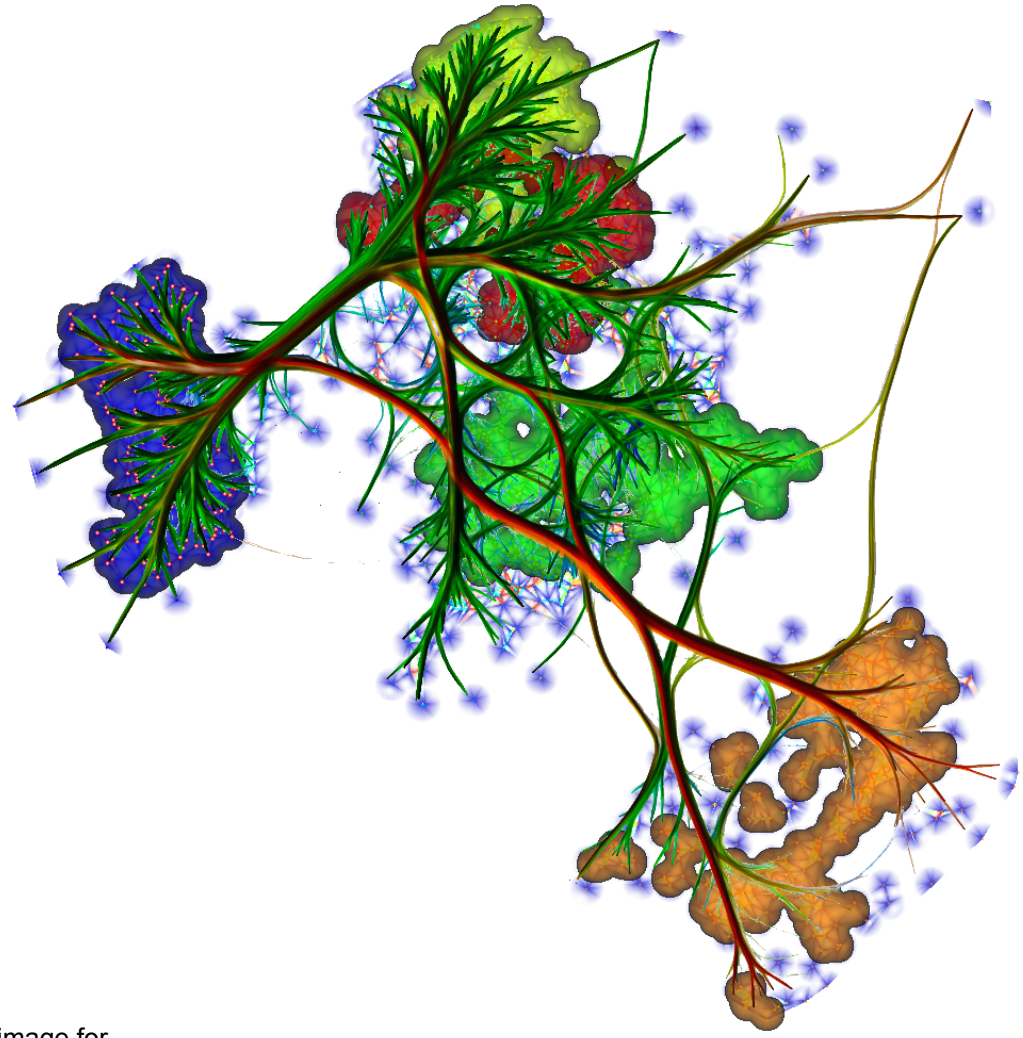


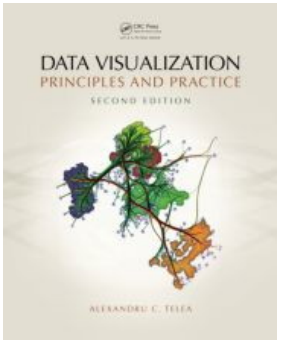
- 40% of docs frequently updated
- rest seem to be ‘stale’

Code is arguably well documented…
…so refactoring is likely to be doable

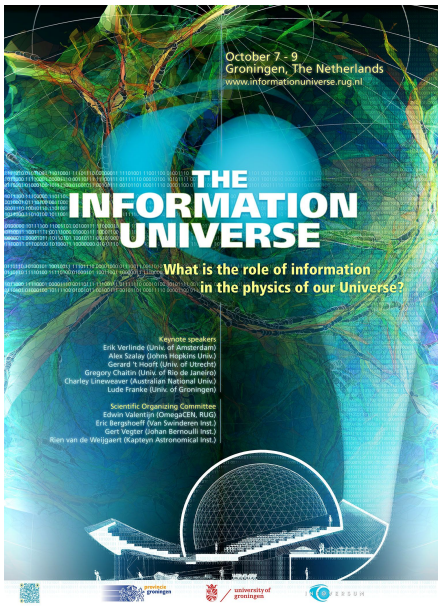
Putting it all together: Showcase example

- 19-dimensional dataset (image classification), visualized with several image-based techniques
  - points: 2D projection of 19-dimensional data, shaded by one attribute
  - bundles: point-to-point projection errors
  - cushions: clusters of similar points





Cover image for
Data Visualization: Principles and Practice, CRC Press, 2014



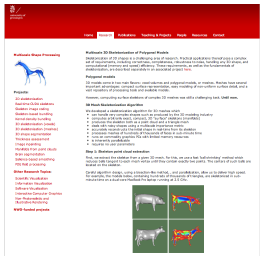
Conclusions

Multiscale big data analytics

- **synergy** of visual computing, multiscale techniques, machine learning
- **visual aspect** is crucial
  - for hypothesis forming: see complex patterns (‘discover the unknown’)
  - for communication: present complex findings intuitively to users
- **computing aspect** is crucial
  - GPU image-based techniques: solution to the volume/velocity problem
- a **real opportunity**, happening right now
- **applications**: anything (finance, healthcare, IT, traffic, engineering, …)

Thank you for your interest!

a.c.telea@rug.nl



www.cs.rug.nl/svcg/Shapes

- examples, applications
- code
- datasets
- papers