The Other Side of Model-Driven Development

Grady Booch
IBM Fellow and Free Radical

Disclaimer The following is not necessarily the opinion of IBM or of any other person living or dead or yet to be born.

No models were injured in the creation of this presentation.

Credits

Ruth Malan on visualization

- traceinthesand.com

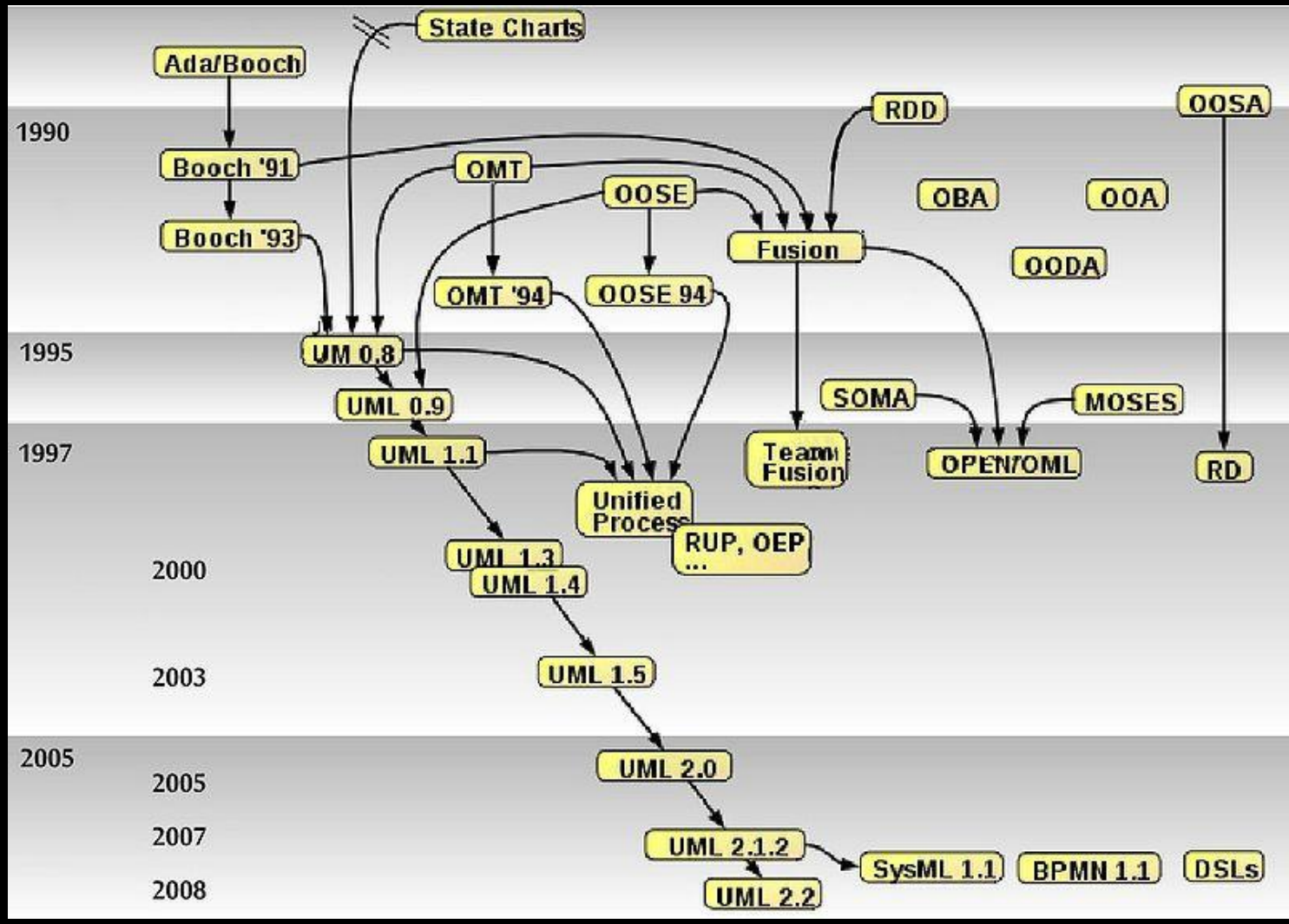
Tilak Mitra on views

- IBM Global Services

UML Origins

The Unified Modeling Language is the industry-standard language for specifying, visualizing, constructing, and documenting the artifacts of a software-intensive system.

UML Origins



http://en.wikipedia.org/wiki/Unified_Modeling_Language

UML Origins Deconstructed

The Unified Modeling Language is the industry-standard language for specifying, visualizing, constructing, and documenting the artifacts of a software-intensive system.

Specifying Abstraction with freedom but without ambiguity

- Define
- Designate
- Delineate

Visualizing Abstraction with focus

- Picture
- Project
- Present

Constructing Artifacts at a moment in time

- Make manifest
- Manufacture
- Metamorphose

Documenting Artifacts across time and space

- Record
- Represent
- Reserve

Artifacts

Artifacts for many stakeholders

- Executables
- Scaffolding
- Legacy

Software-Intensive Systems

Abstractions made manifest

- Hardware
- Software
- Wetware
- Systems of systems

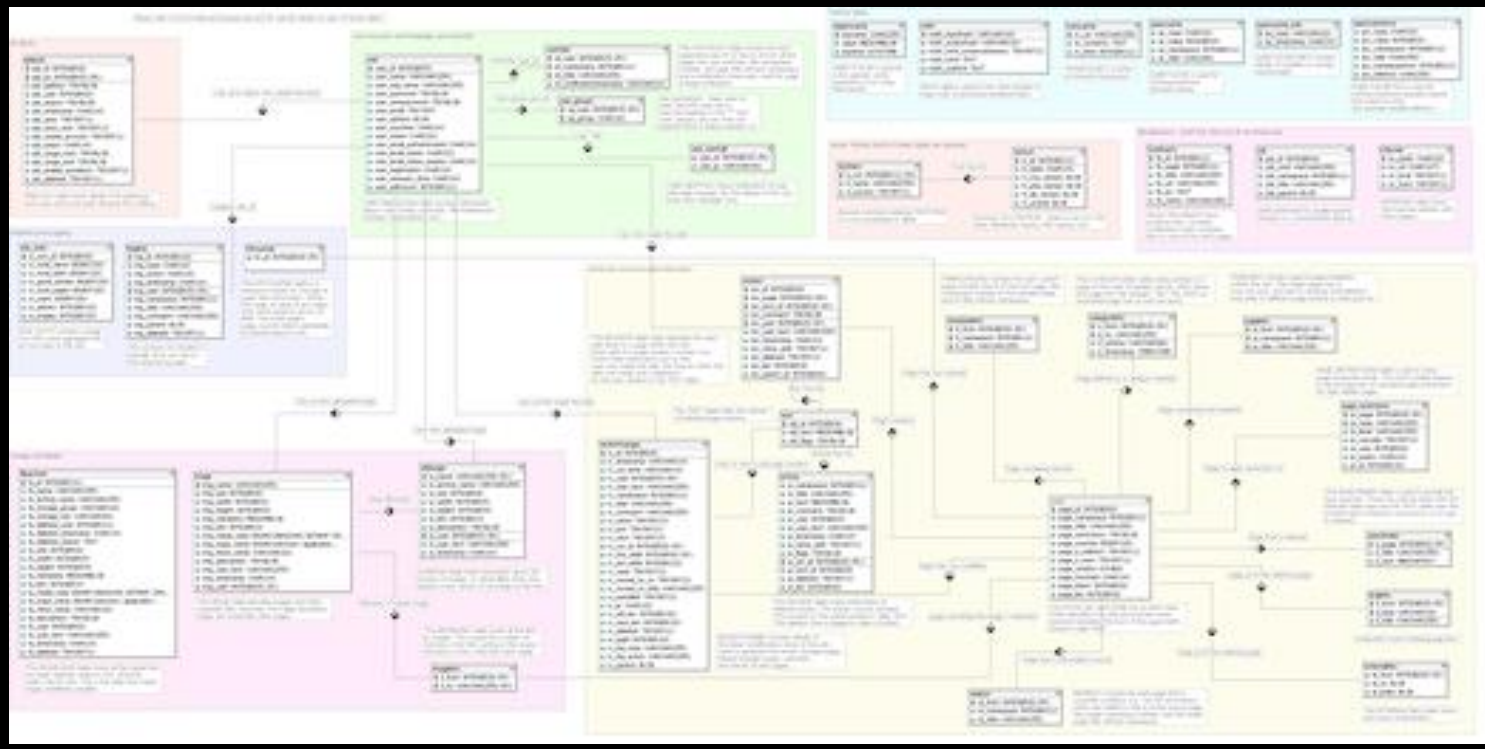
UML Strengths

- Expressive
- Extensible and adaptable
- Reasonably well-defined
- Widely supported

UML In The Wild

Aerospace
Artificial intelligence
Commercial
Content authoring
Development
Devices
Entertainment and sports
Financial
Game and virtual worlds
Governmental
Industrial
Legal
Medical
Military
Operating systems
Platforms
Scientific
Telecommunications
Transportation
Utilities

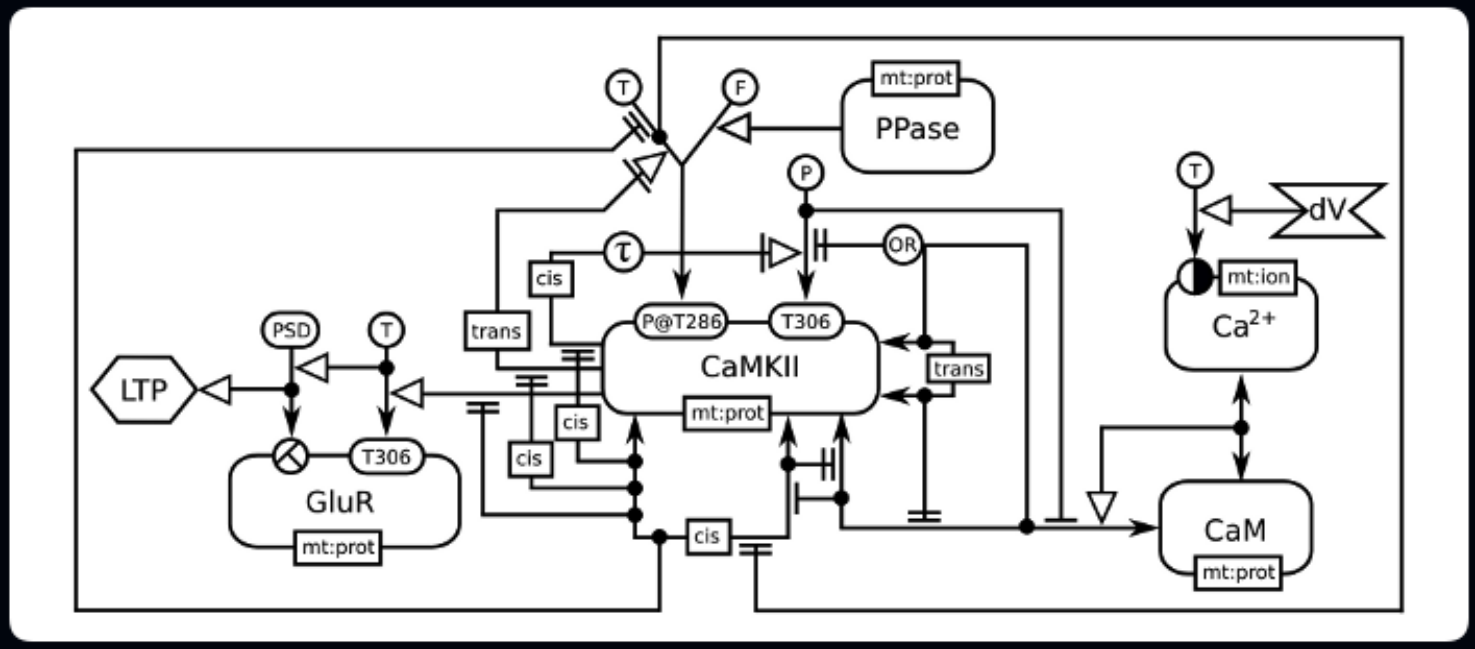
Wikimedia



http://gadgets.boingboing.net/2009/01/21/chart-mediawiki-wiki.html

Systems Biology Graphical Notation

IBM Research



http://sbgn.org/

UML Weaknesses

- Language bloat
- Problems in learning and adopting
- Cumulative impedance mismatching
- Dysfunctional interchange format

http://en.wikipedia.org/wiki/Unified_Modeling_Language#Criticisms

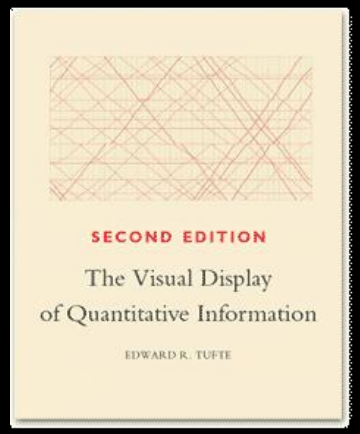
Personal Observation

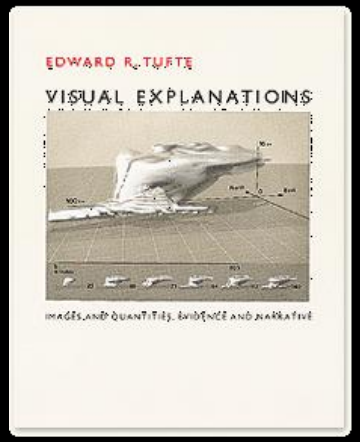
Abstractions are not reality.

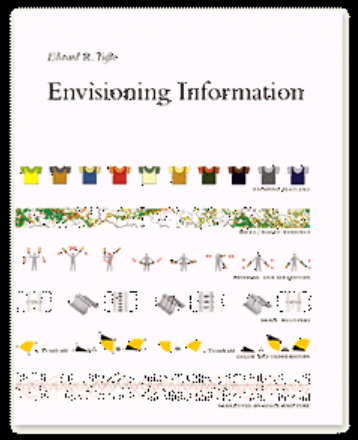
The original authors of the UML never intended it to be a programming language.

We've lost our way.

Edward Tufte

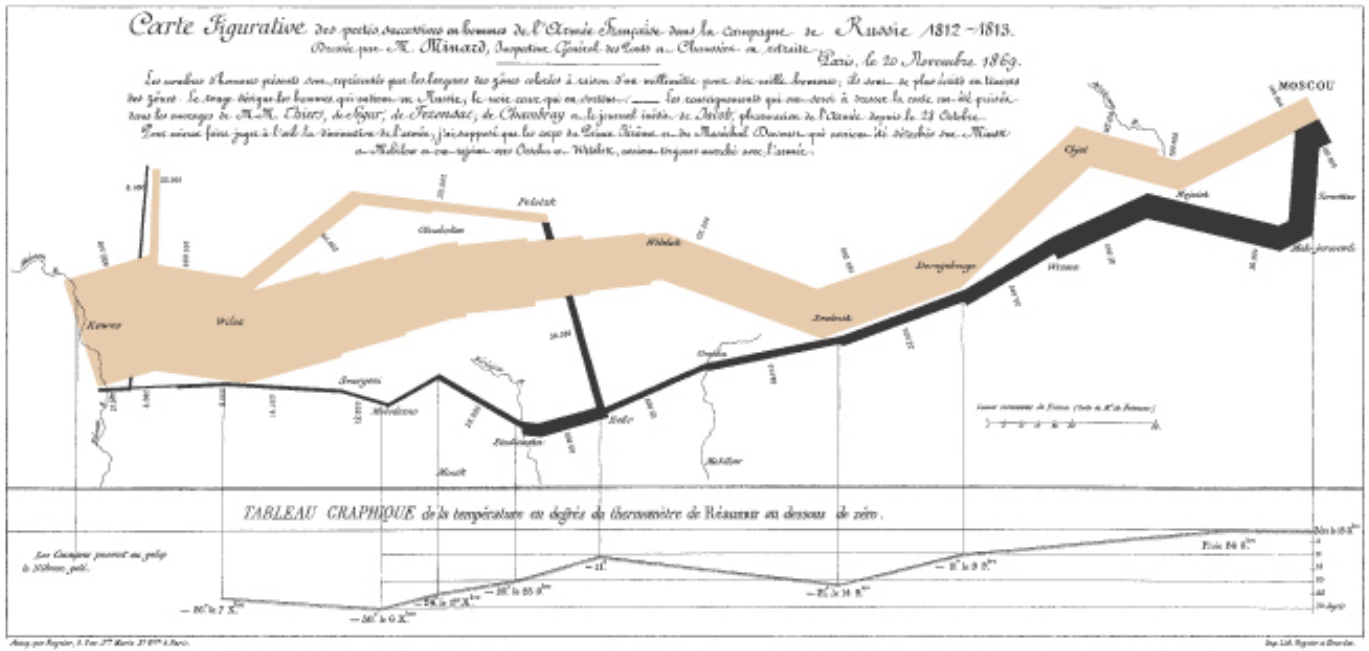






http://www.edwardtufte.com

Edward Tufte


Carte Figurative des pertes successives en hommes de l'Armée Française dans la campagne de Russie 1812–1813.
Dressée par M. Minard, Inspecteur Général des Ponts et Chaussées en retraite.
Paris, le 20 Novembre 1869.
MOSCOU
Kowno
Wilna
Smorgoni
Molodetchno
Studienka
Zodoi
Moiaisk
Dorogobouge
Witebsk
Smolensk
Polotsk
Orscha
Mohilow
Gloubokoe
Minsk
Wiazma
Gjatsk
Mojaisk
Tarantino
Malo-Jaroslawetz
TABLEAU GRAPHIQUE de la température en degrés du thermomètre de Réaumur au dessous de zéro.
Les Cavaliers passèrent au galop le Niémen gelé.
Fait le 9 X^bre
Fait le 14 X^bre
Fait le 24 X^bre
Fait le 6 X^bre
Fait le 1 X^bre

Edward Tufte

The minimum we should hope for with any display technology is that it should do no harm.

It's not that PowerPoint brought the Columbia down, but the method of presentation broke up the argument into tiny fragments, and it's intensely hierarchical - no sentences, just little phrases.

We've drifted into this presentation mode without realizing the cost to the content and the audience in the process.

The point is that analytical designs are not to be decided on their convenience to the user or necessarily their readability or what psychologists or decorators think about them; rather, design architectures should be decided on how the architecture assists analytical thinking about evidence.

What Modeling Is

Abstraction of reality

What Modeling Is Not

Abstraction is not reality

What Modeling Should Be

Abstraction with freedom but without ambiguity
Abstraction with focus
Artifacts at a moment in time
Artifacts across time and space
Artifacts for many stakeholders
Artifacts made manifest

Why We Model

To abstract
To reason about
To document
To transform

To Abstract

Amplification through simplification

- Abstractions are not reality
- Abstractions are intentionally incomplete

To Reason About

Compare, synthesize, analyze, generate abstractions

- The brain has its own language for testing the structure and consistency of the world.

Different models attend to the needs of

- Different stakeholders with
- Different concerns and thus
- Different viewpoints

http://www.youtube.com/watch?v=zSgiXGELjbc

To Document The code is the truth, but not the whole truth

- Architecture as a collection of significant design decisions
- Patterns as the themes
- Cross-cutting concerns as the traces
- Rationale as the back story
- Tribal memory as the human story

To Transform Abstractions made manifest as the executable system itself

- Some transformations are tedious
- Some transformations are noisy
- Some transformations are lossy
- Some transformations are somewhat reversible

Issues of Modeling

What do we model?
What do we leave out?
How do we represent our abstractions?

What Do We Model?

Different models attend to the needs of

- Different stakeholders with
- Different concerns and thus
- Different viewpoints

Specify only that which is essential

- Everything else may be loose
- Everything else must be consistent
- Everything must be interconnected

What Do We Leave Out?

Expunge details that

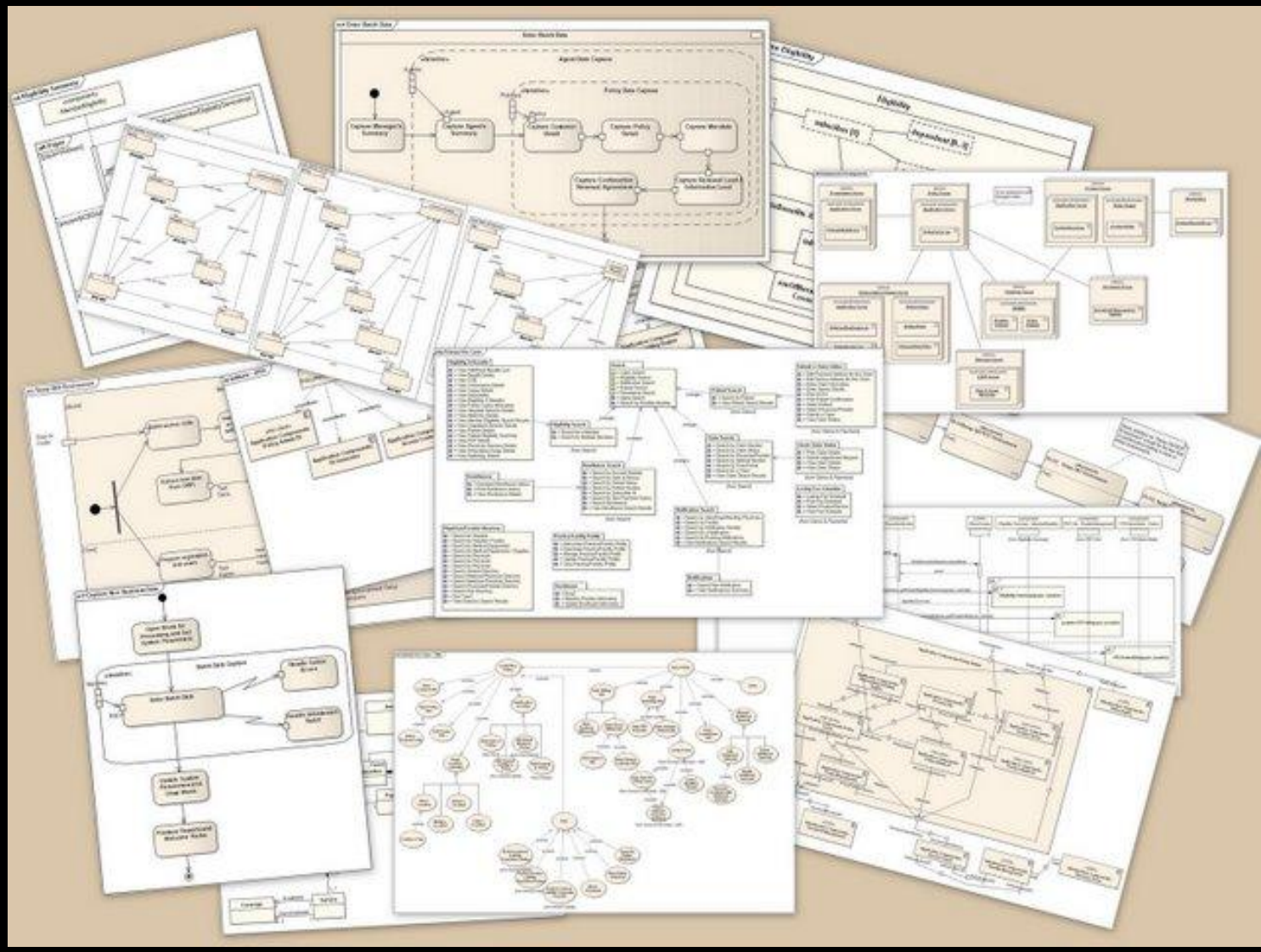
- Do not advance understanding
- Obscure the essential abstraction

How Do We Represent Our Abstractions?

Use the representation best suited to the semantics of the abstraction

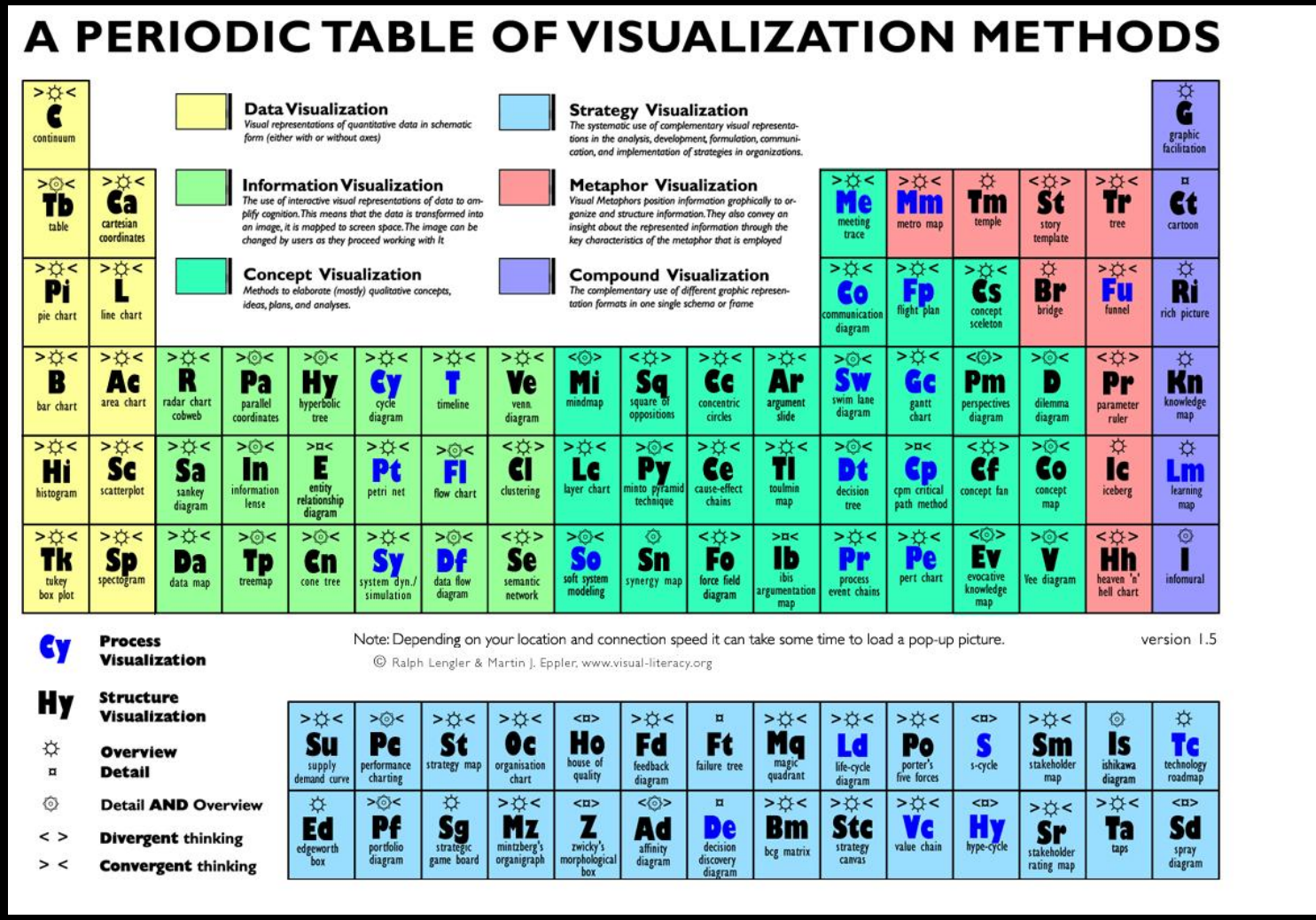
- Some graphical, some textual
- Some static, some dynamic

How Do We Represent Our Artifacts?



http://en.wikipedia.org/wiki/File:UML_Diagrams.jpg

Visualization Methods



http://www.visual-literacy.org/periodic_table/periodic_table.html

Visualization Methods

Data visualization
- Representations of quantitative data

Information visualization
- Representations of data to amplify cognition

Concept visualization
- Methods to elaborate qualitative concepts

Strategy visualization
- Complementary visual representations of strategy

Metaphor visualization
- Insights about information

Compound visualization
- Multiple formats in one frame

http://www.visual-literacy.org/periodic_table/periodic_table.html

Visualization Methods

Conferences

- IEEE Workshop on Visualizing Software for Understanding and Analysis
- IEEE Conference on Program Comprehension
- Working Conference on Reverse Engineering

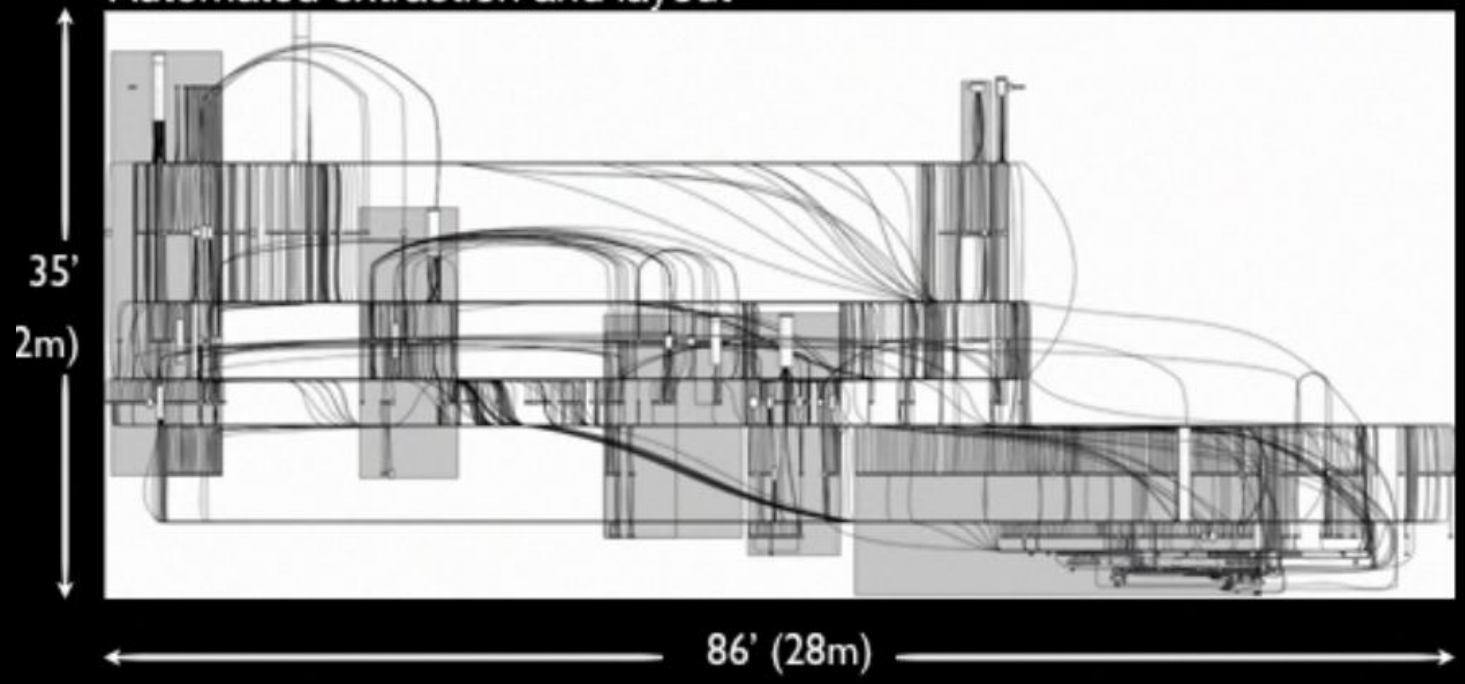
Sites

- http://www.visualcomplexity.com/vc
- http://www.nsf.gov/news/special_reports/scivis/winners_2008.jsp
- http://www.ruthmalan.com/Journal/JournalCurrent.htm

SPQR

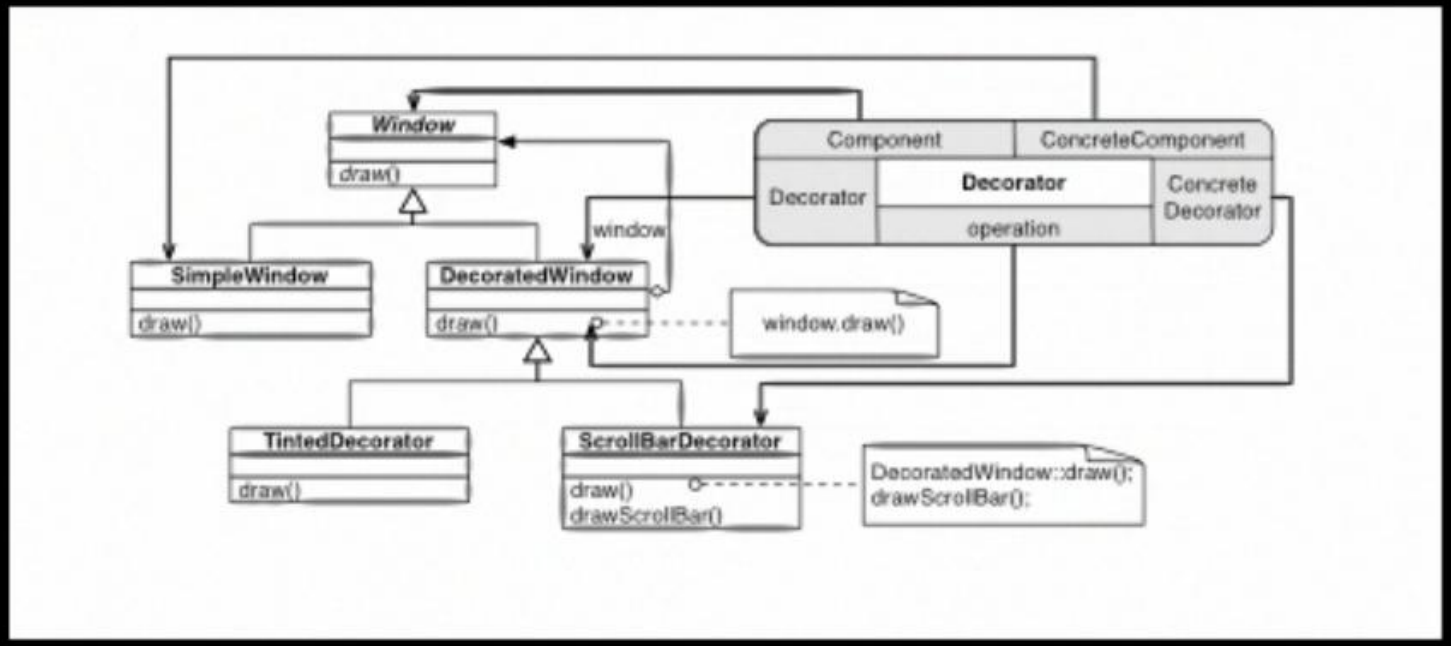
300kLOC of C firmware as UML

Automated extraction and layout



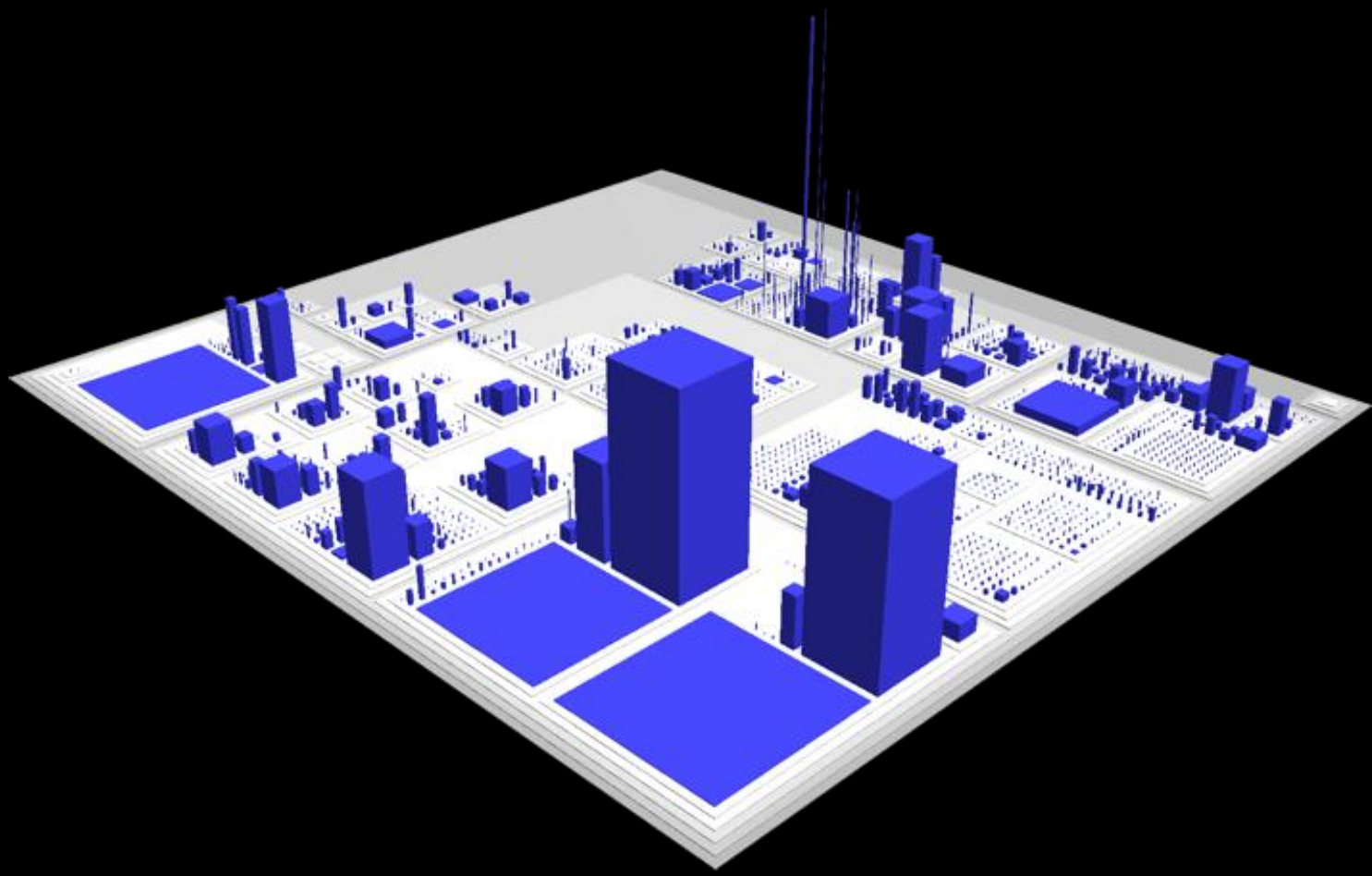
http://www.cs.unc.edu/~smithja/Jason_McColm_Smith.html

SQPR



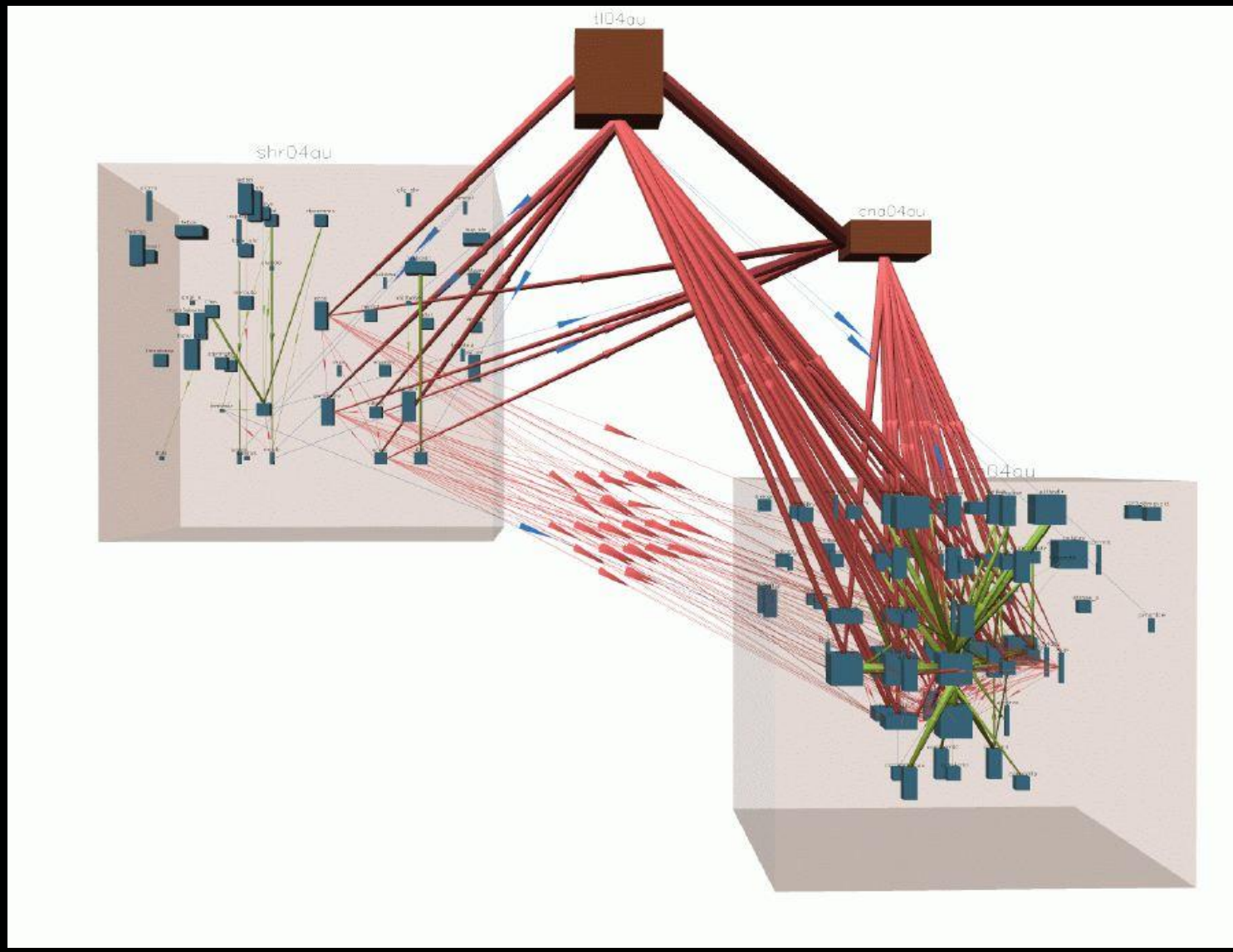
http://www.cs.unc.edu/~smithja/Jason_McColm_Smith.html

CodeCity



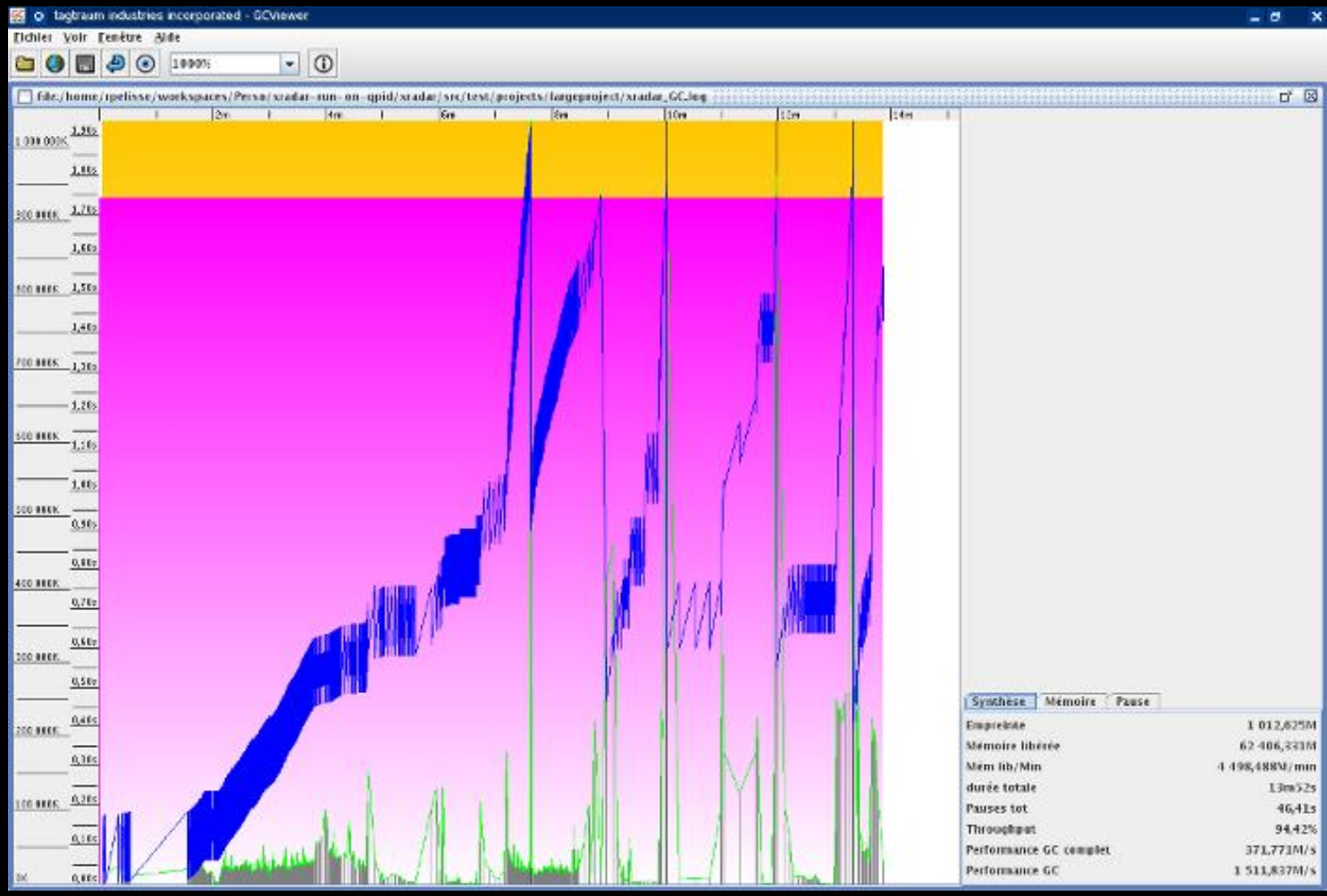
http://www.inf.unisi.ch/phd/wettel/codecity.html

NV3D



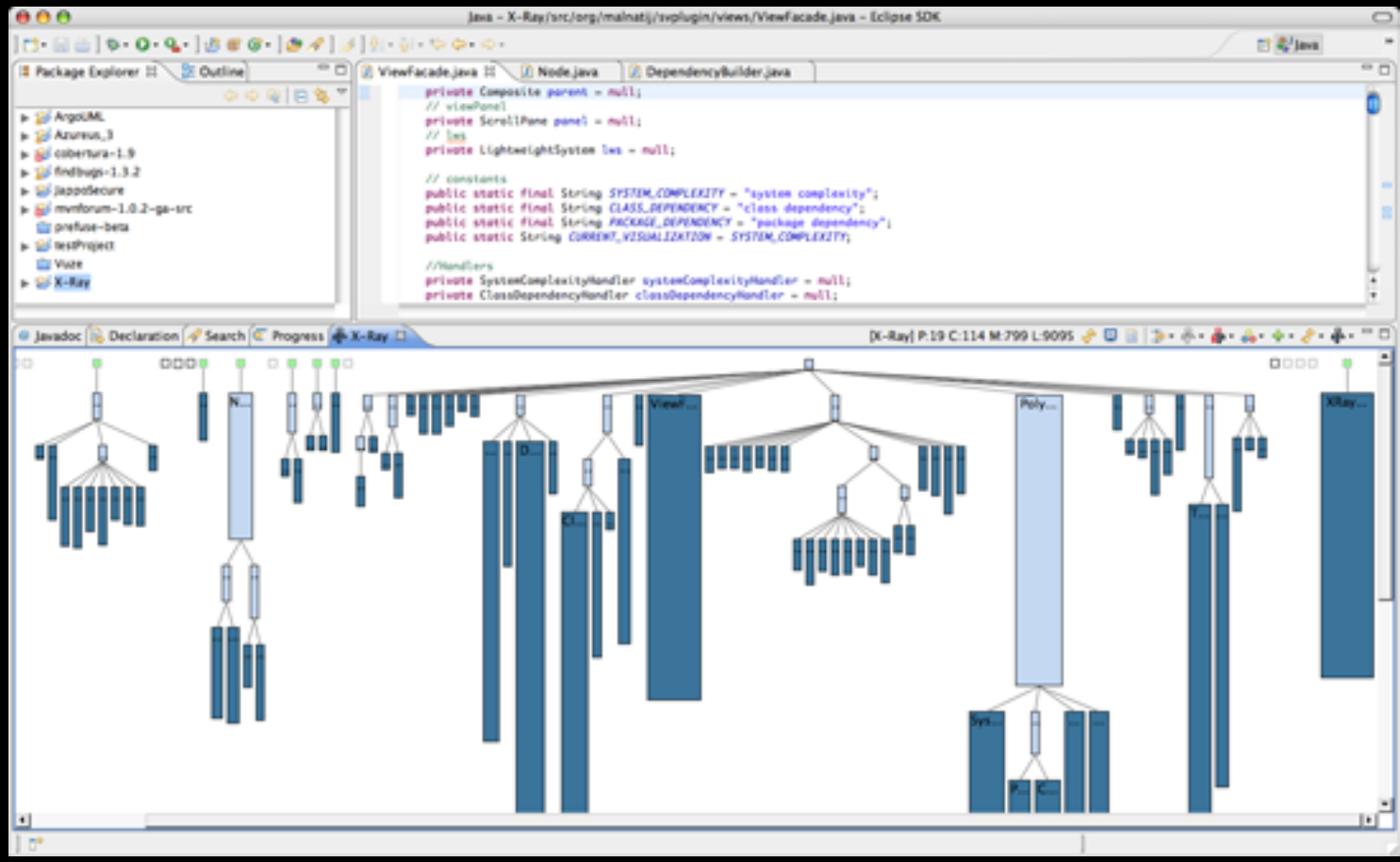
http://ccom.unh.edu/vislab/projects/InfoNets.html

XRadar



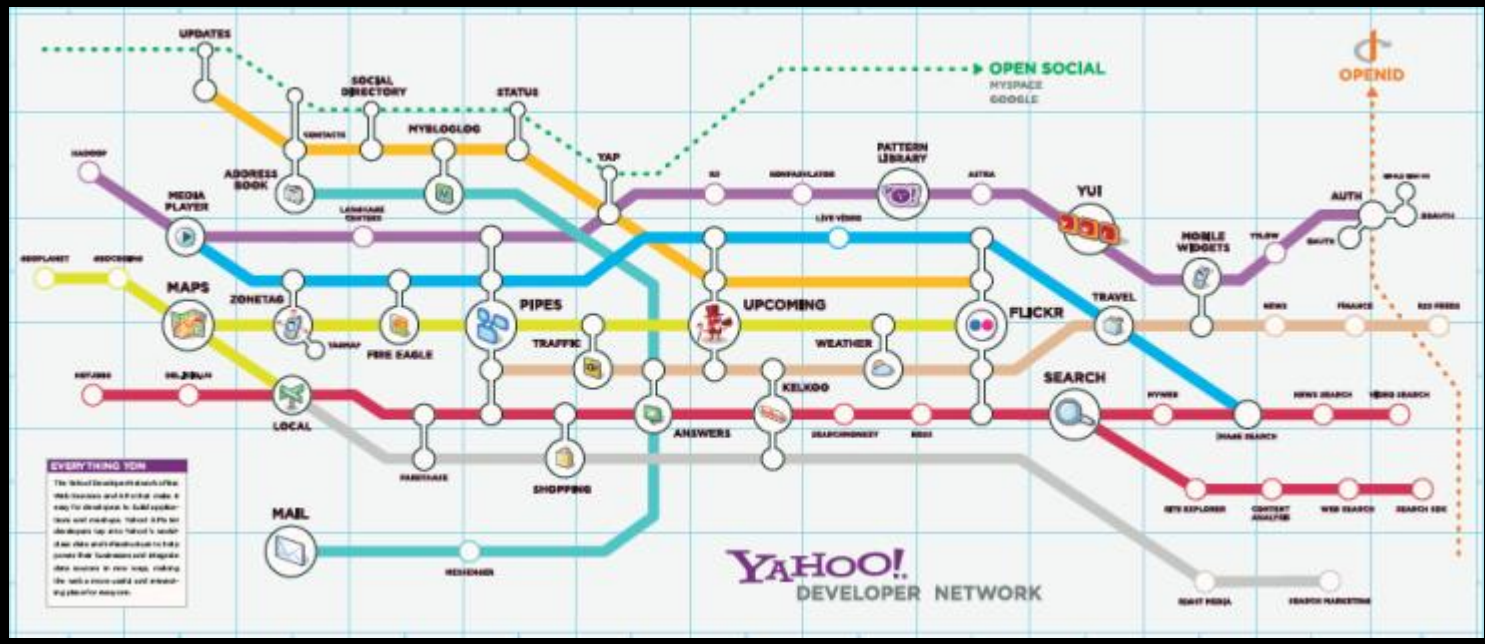
http://xradar.sourceforge.net/

X-Ray



http://atelier.inf.unisi.ch/~malnatij/xray.php

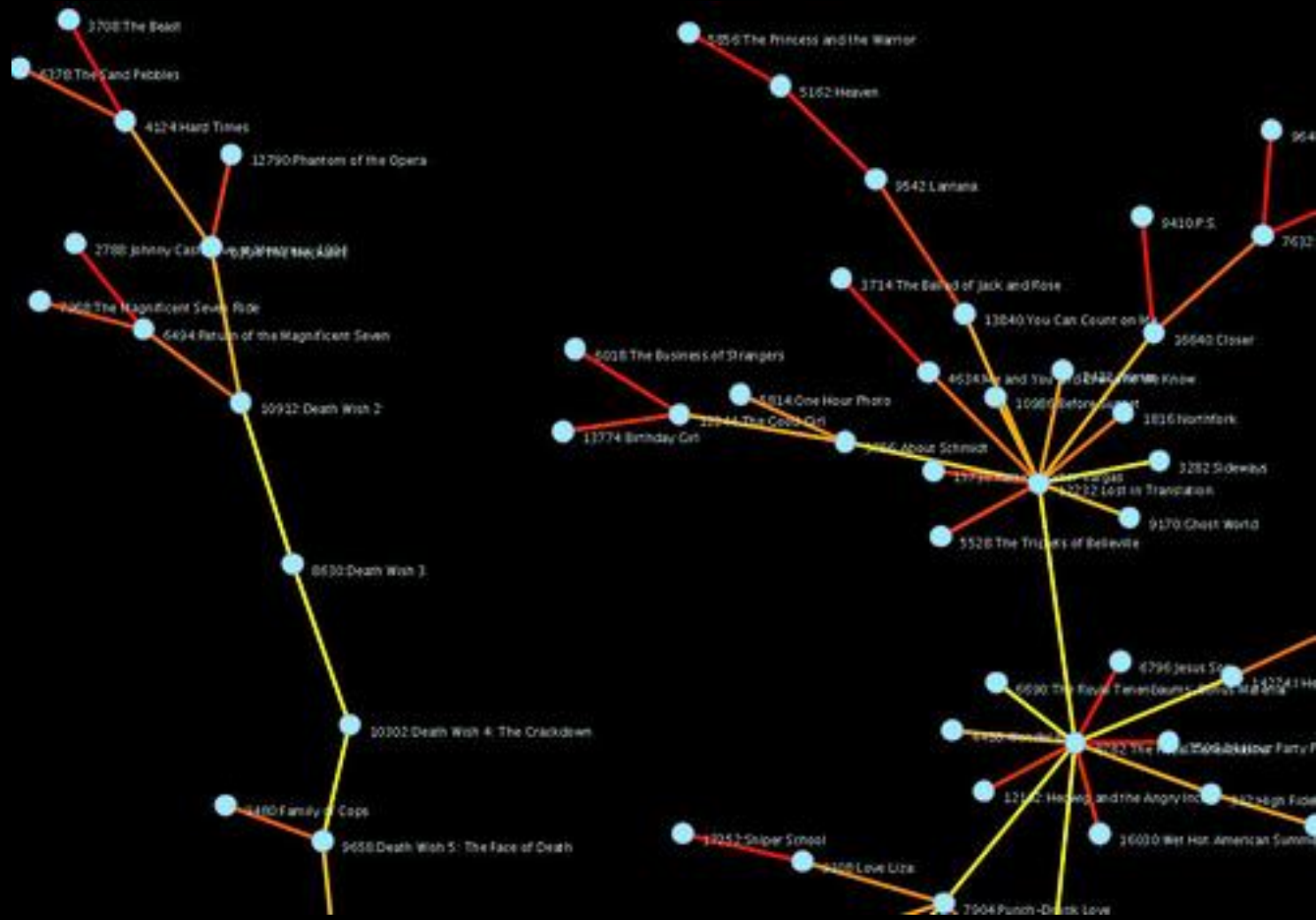
Yahoo API



http://visualthinkmap.ning.com/photo/2168552:Photo:2865?context=latest

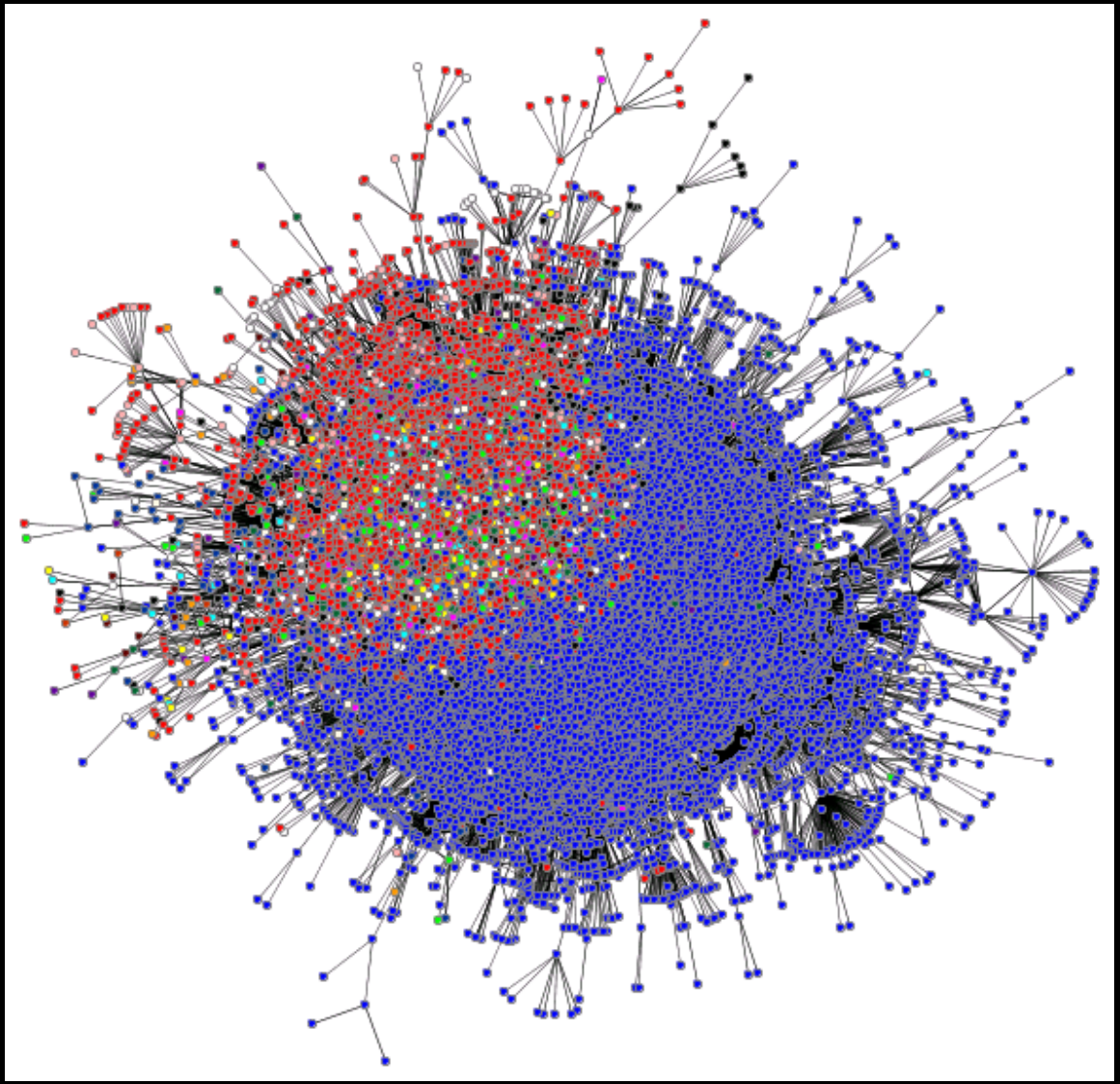
IBM Research

Netflix



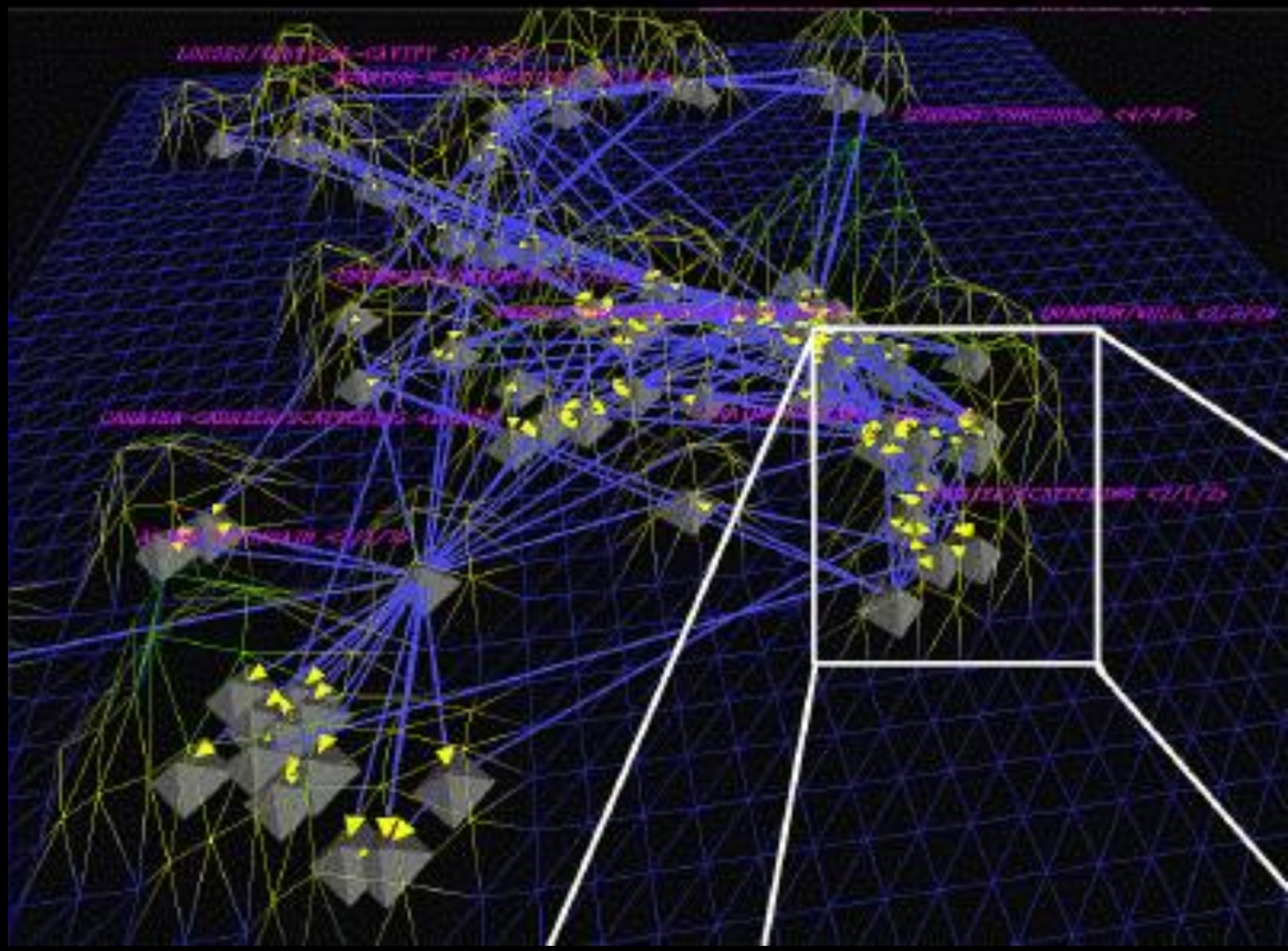
http://guess.wikispot.org/

Blogosphere



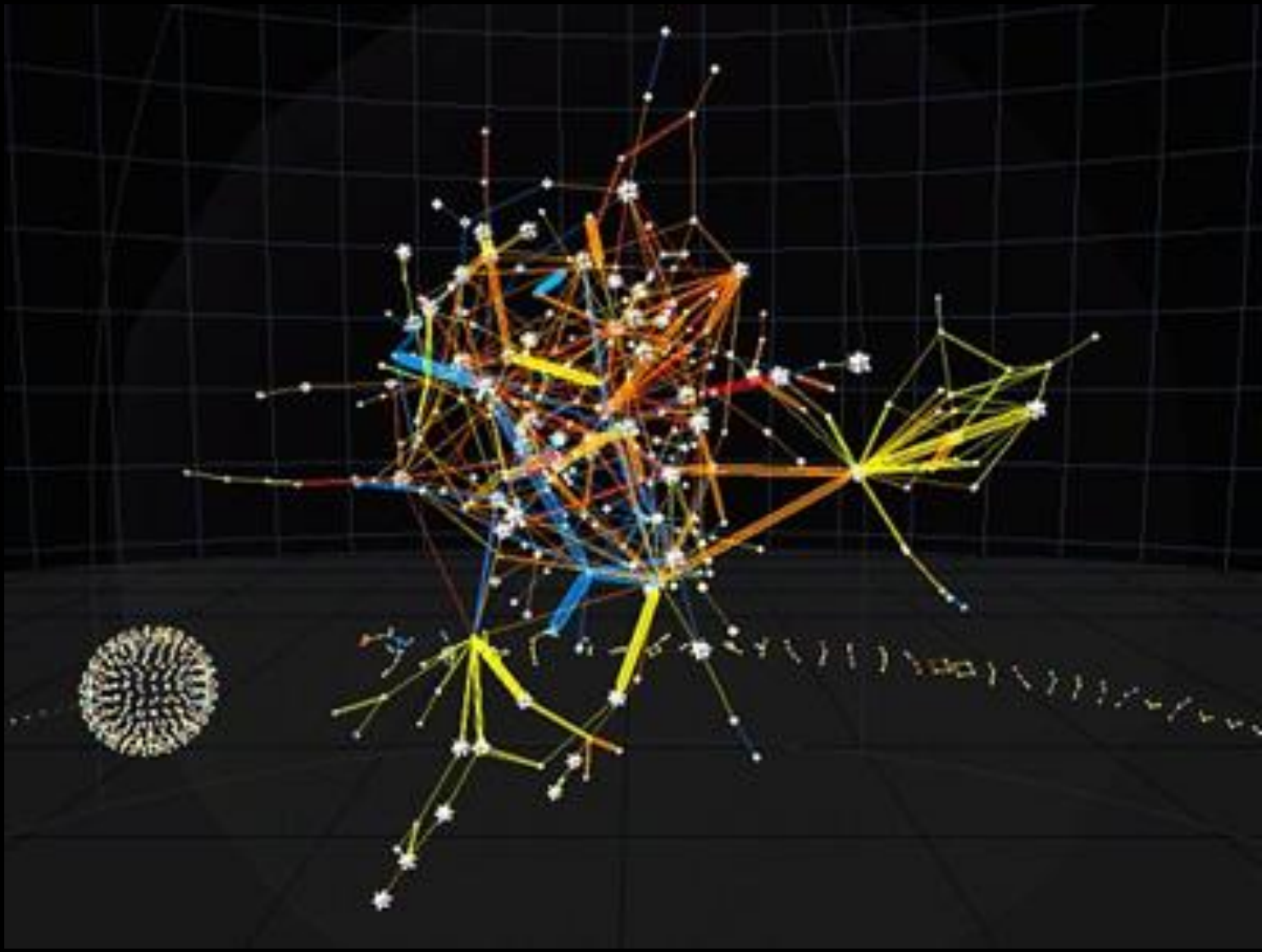
http://datamining.typepad.com/gallery/blog-map-gallery.html

VxInsight



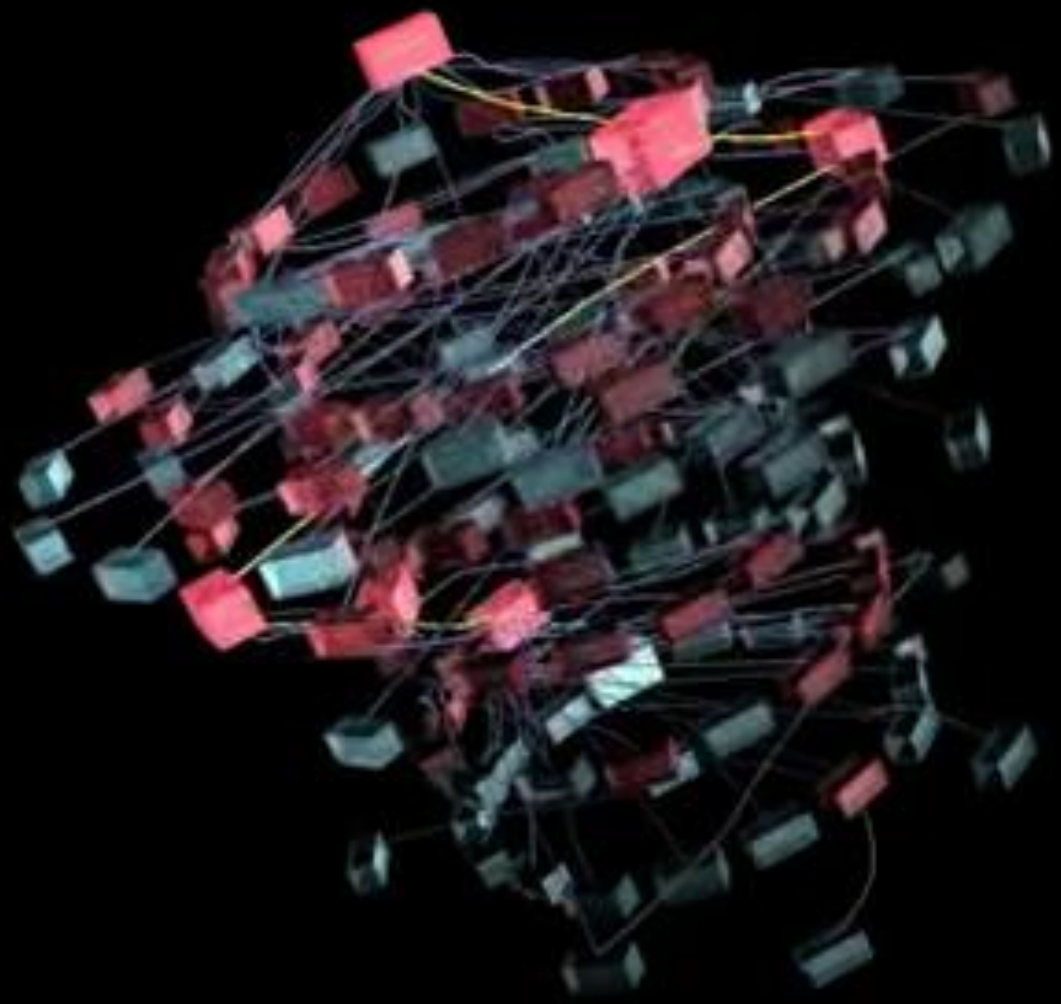
http://www.visualcomplexity.com/vc/project_details.cfm?id=78&index=4&domain=Computer%20Systems

Starlight



http://www.visualcomplexity.com/vc/project_details.cfm?id=150&index=6&domain=Computer%20Systems

Bagle Worm

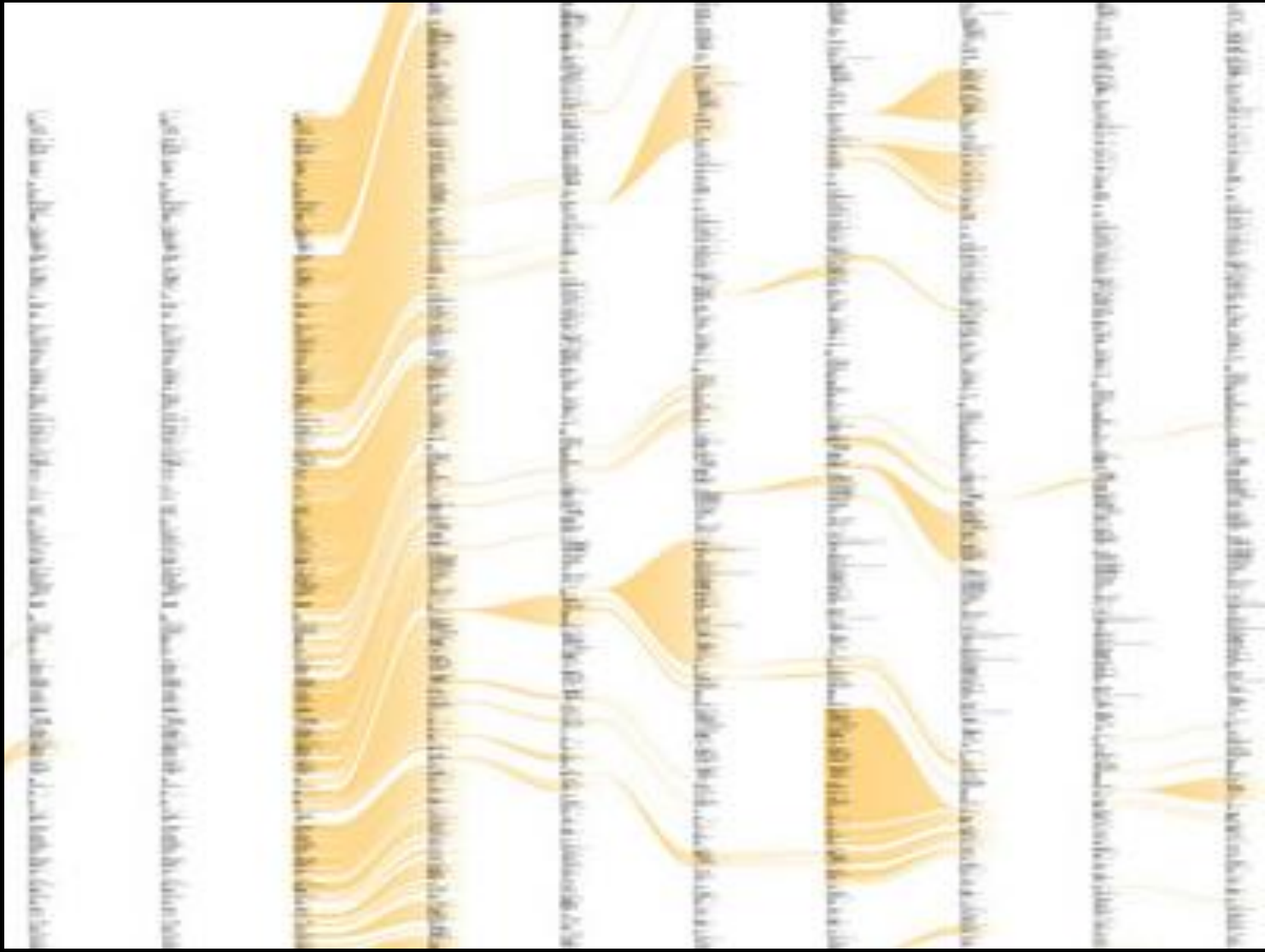


http://www.visualcomplexity.com/vc/project_details.cfm?id=197&index=10&domain=Computer%20Systems

Code Profiles

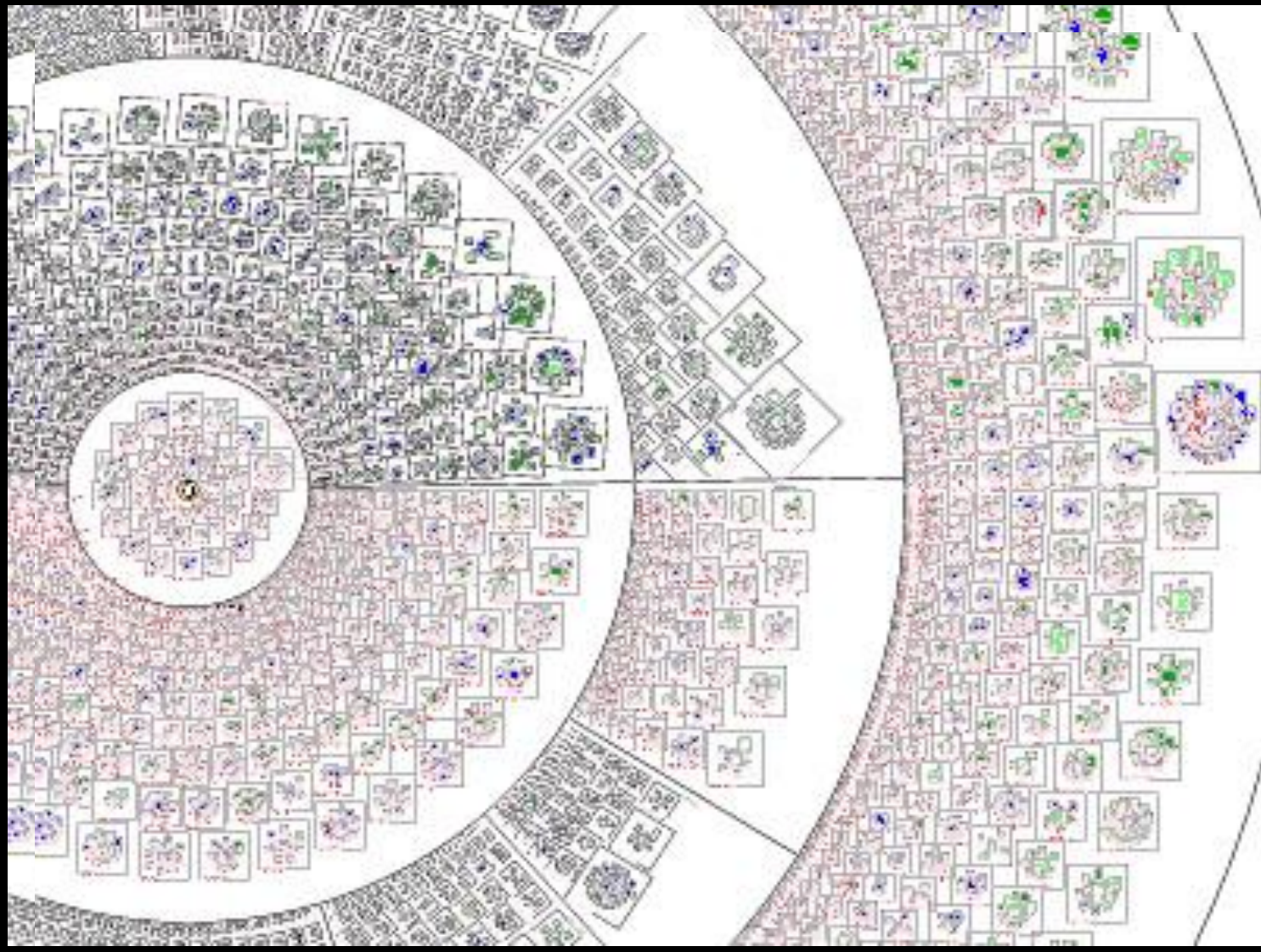
http://www.visualcomplexity.com/vc/project_details.cfm?id=223&index=12&domain=Computer%20Systems

Revisionist



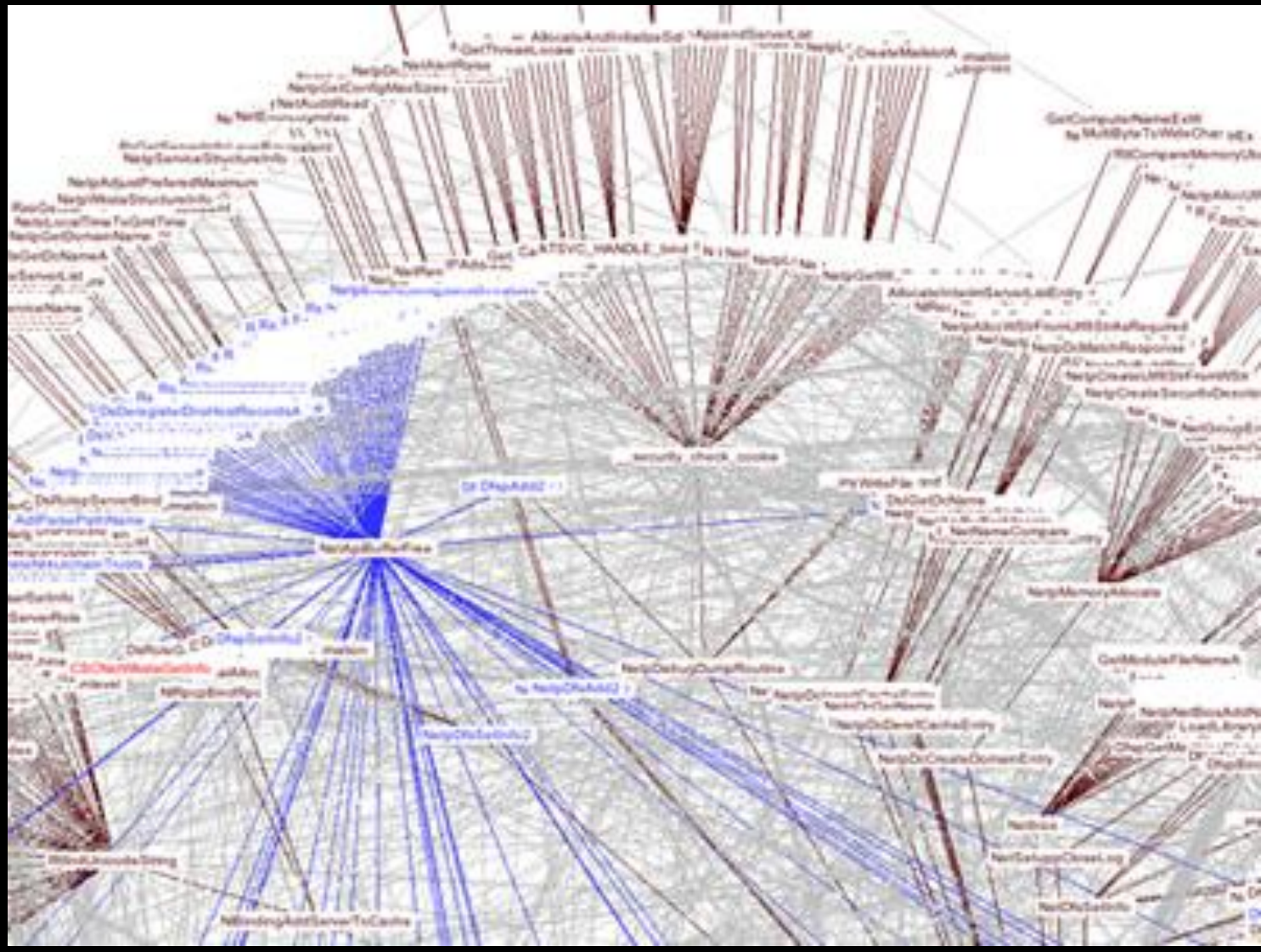
http://www.visualcomplexity.com/vc/project_details.cfm?id=244&index=13&domain=Computer%20Systems

Linux Kernel



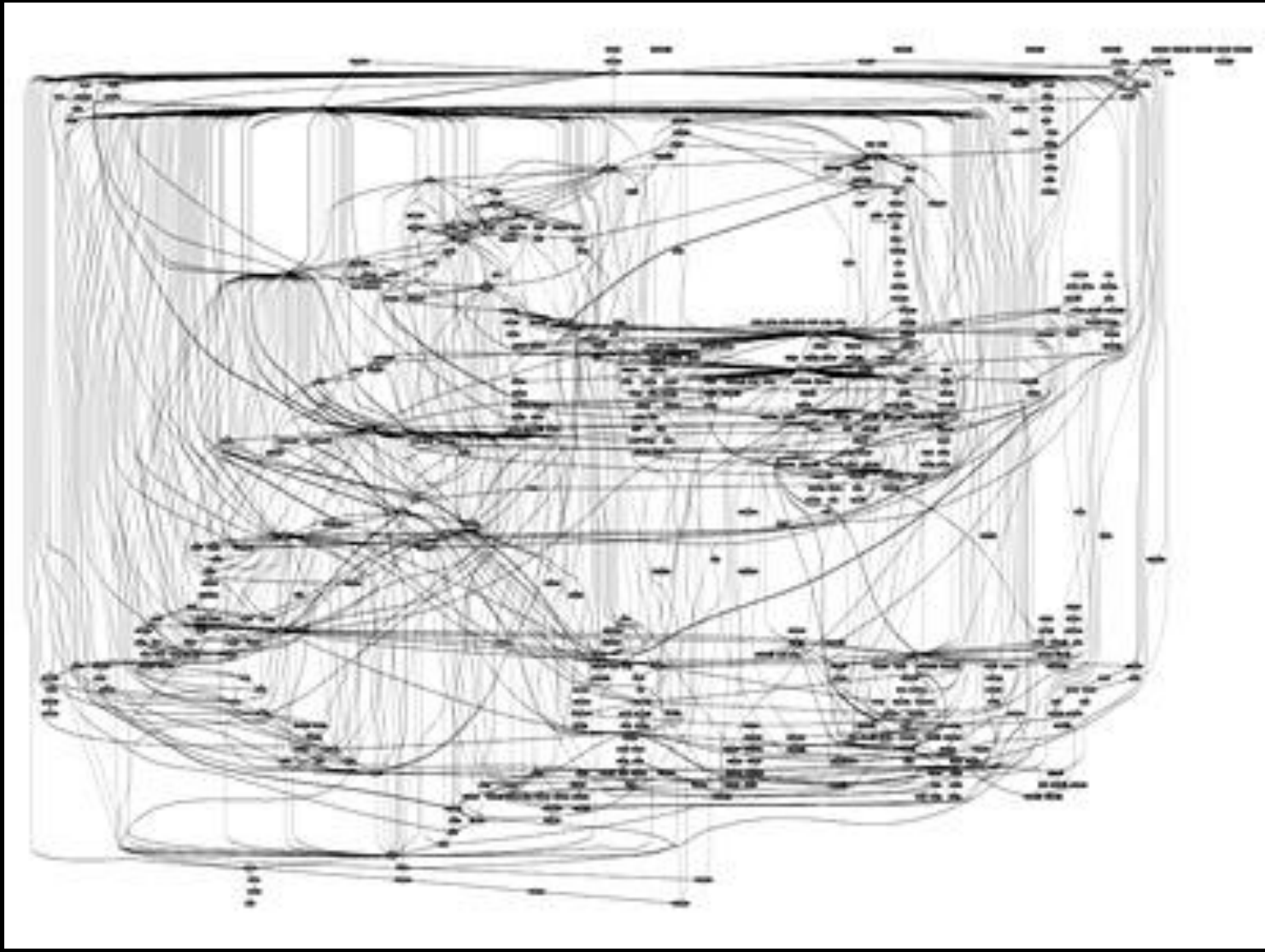
http://www.visualcomplexity.com/vc/project_details.cfm?id=261&index=14&domain=Computer%20Systems

Win32



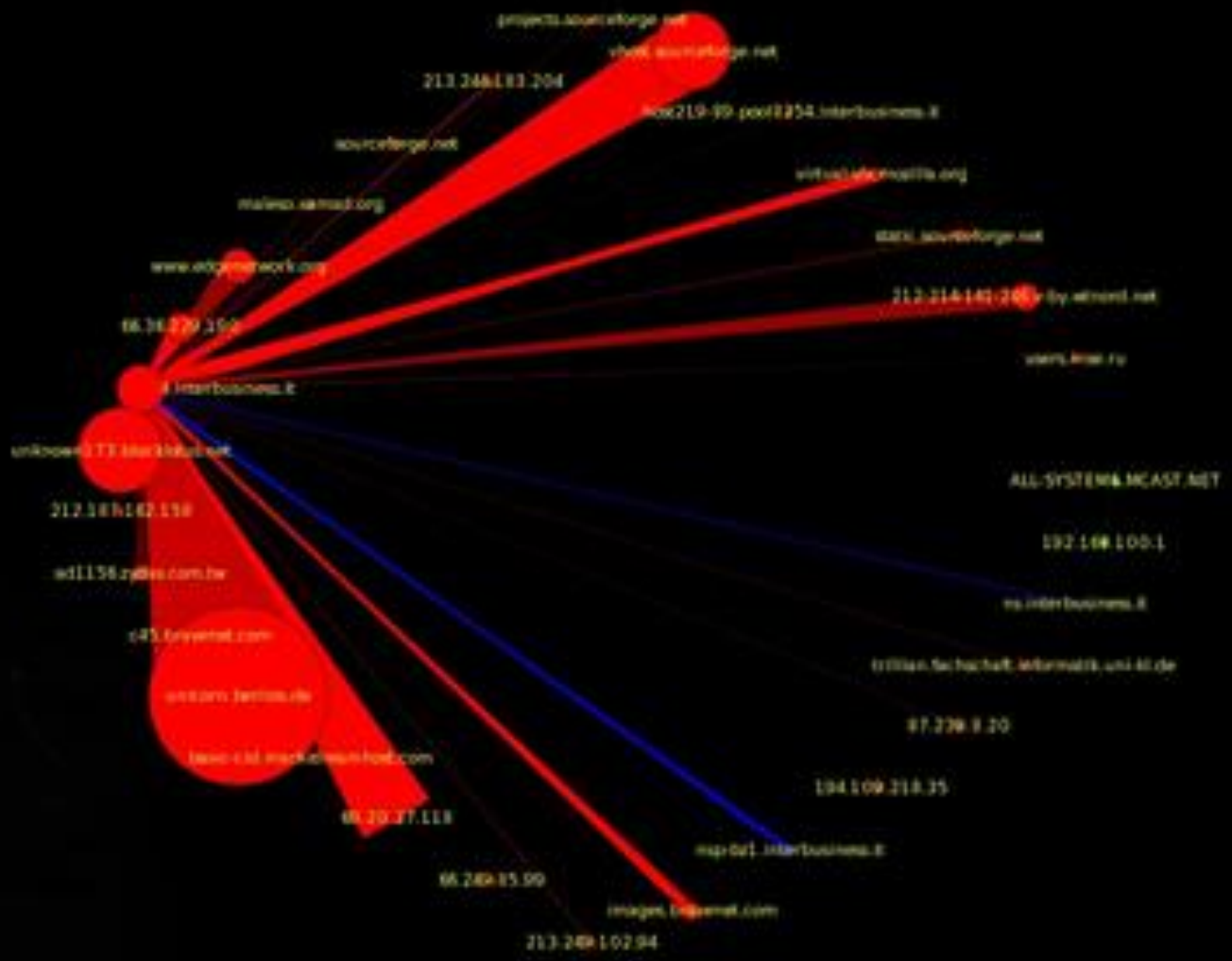
http://www.visualcomplexity.com/vc/project_details.cfm?id=280&index=17&domain=Computer%20Systems

Linux/Windows Security



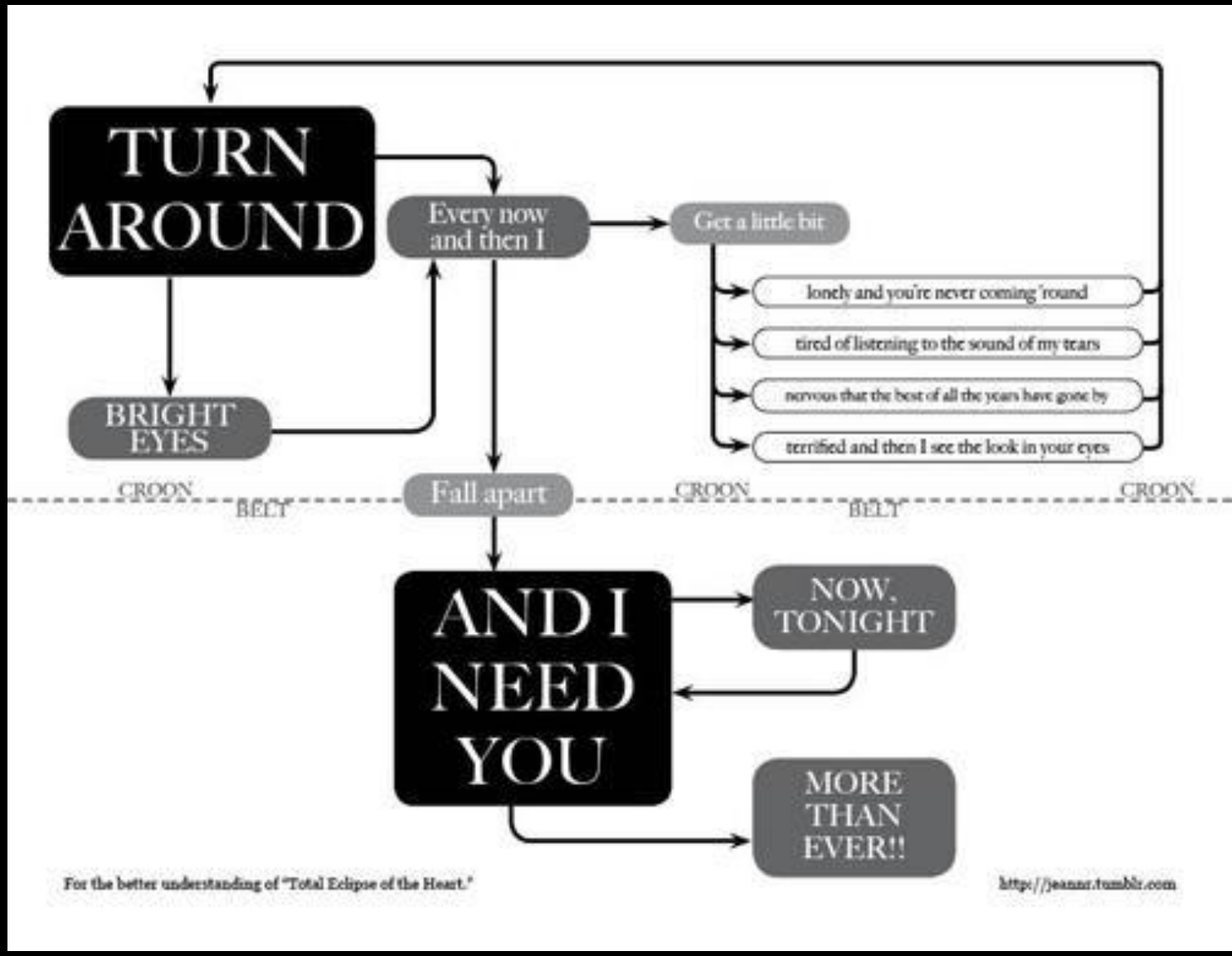
http://www.visualcomplexity.com/vc/project_details.cfm?id=392&index=19&domain=Computer%20System

EtherApe



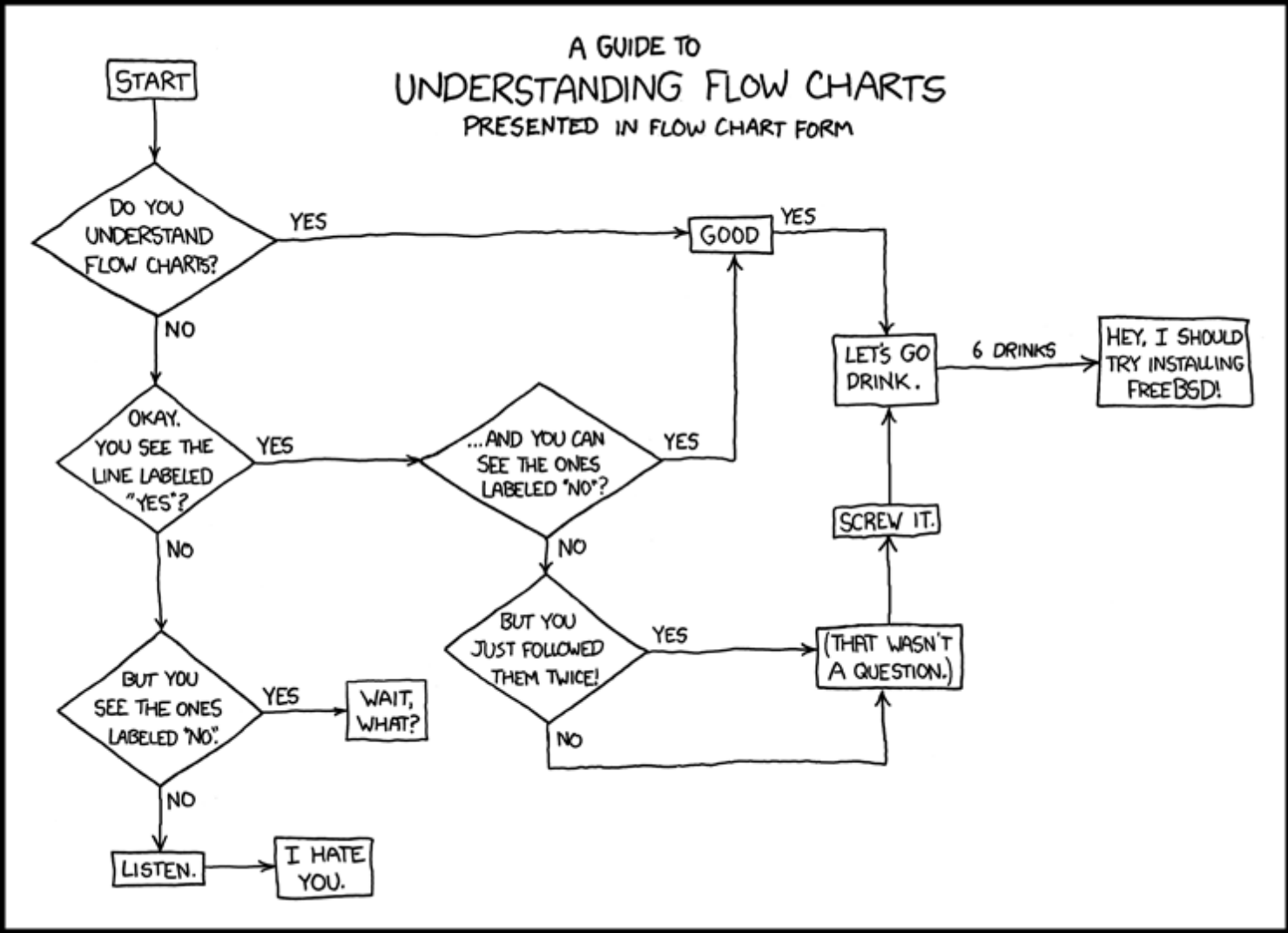
http://www.visualcomplexity.com/vc/project_details.cfm?id=515&index=24&domain=Computer%20Systems

Total Eclipse Of The Heart



http://jeannr.tumblr.com/post/165291081/i-made-a-flow-chart-that-we-might-better#disqus_thread

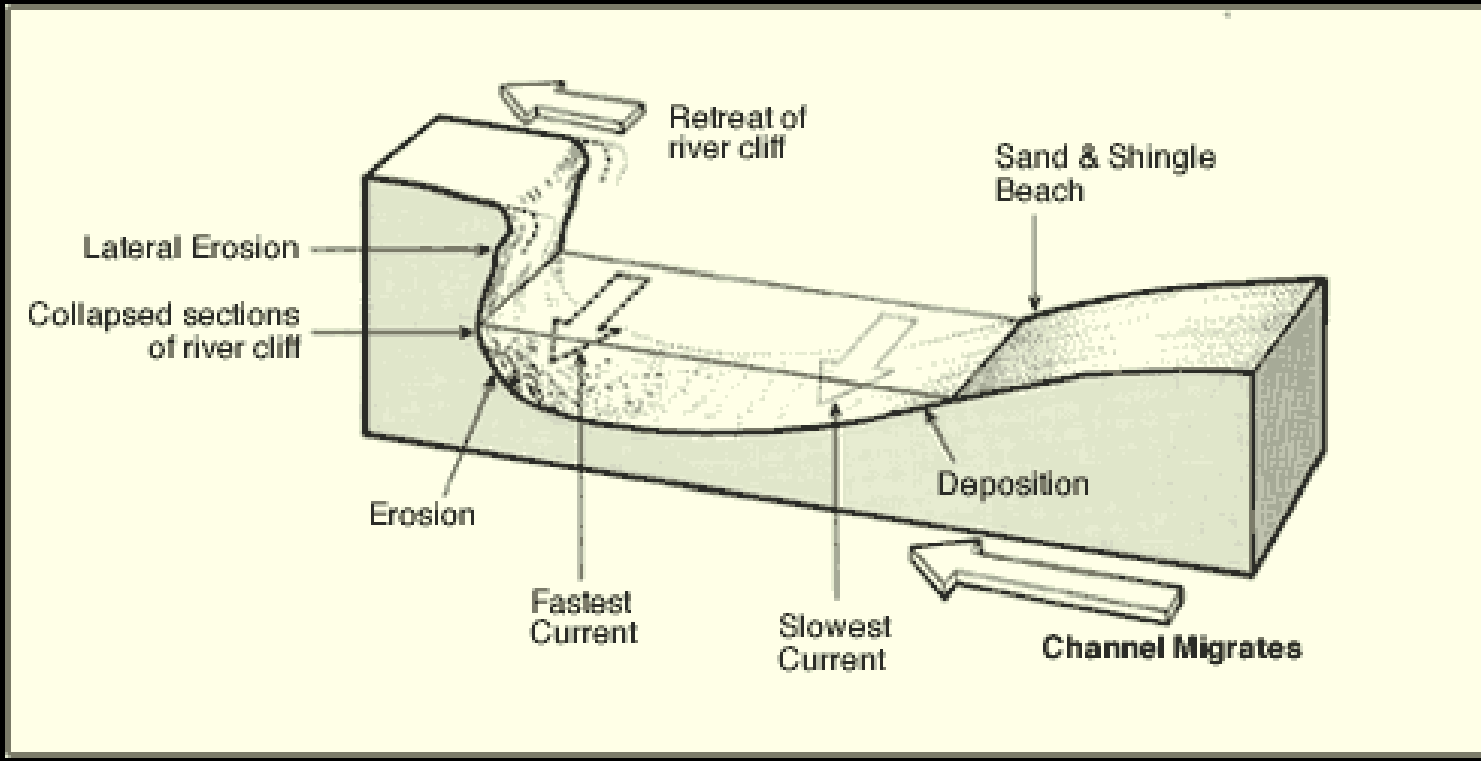
Metaflowchart



http://xkcd.com/518/

An Classic Analogy

A Fresh Analogy



A Fresh Analogy

Therefore The architecture of an enterprise's software intensive systems is akin to the instantaneous structure and behavior of a river

The lifecycle of that architecture is akin to the intentional and accidental morphing of those instantaneous architctures over a region of time.

Points of Pain

What Pain Do You Feel?

How do we attend to new requirements without being saddled by our legacy (but at the same time not compromising that legacy?)

How do we integrate new technology into our existing code base?

How do we integrate our existing systems to extract greater value from the whole?

How do we increase our agility in response to the market while simultaneously improving efficiency and quality yet also reducing costs?

How do we attend to assets introduced through acquisition?

How do use software to improve market efficiency through the creation of dominant product lines?

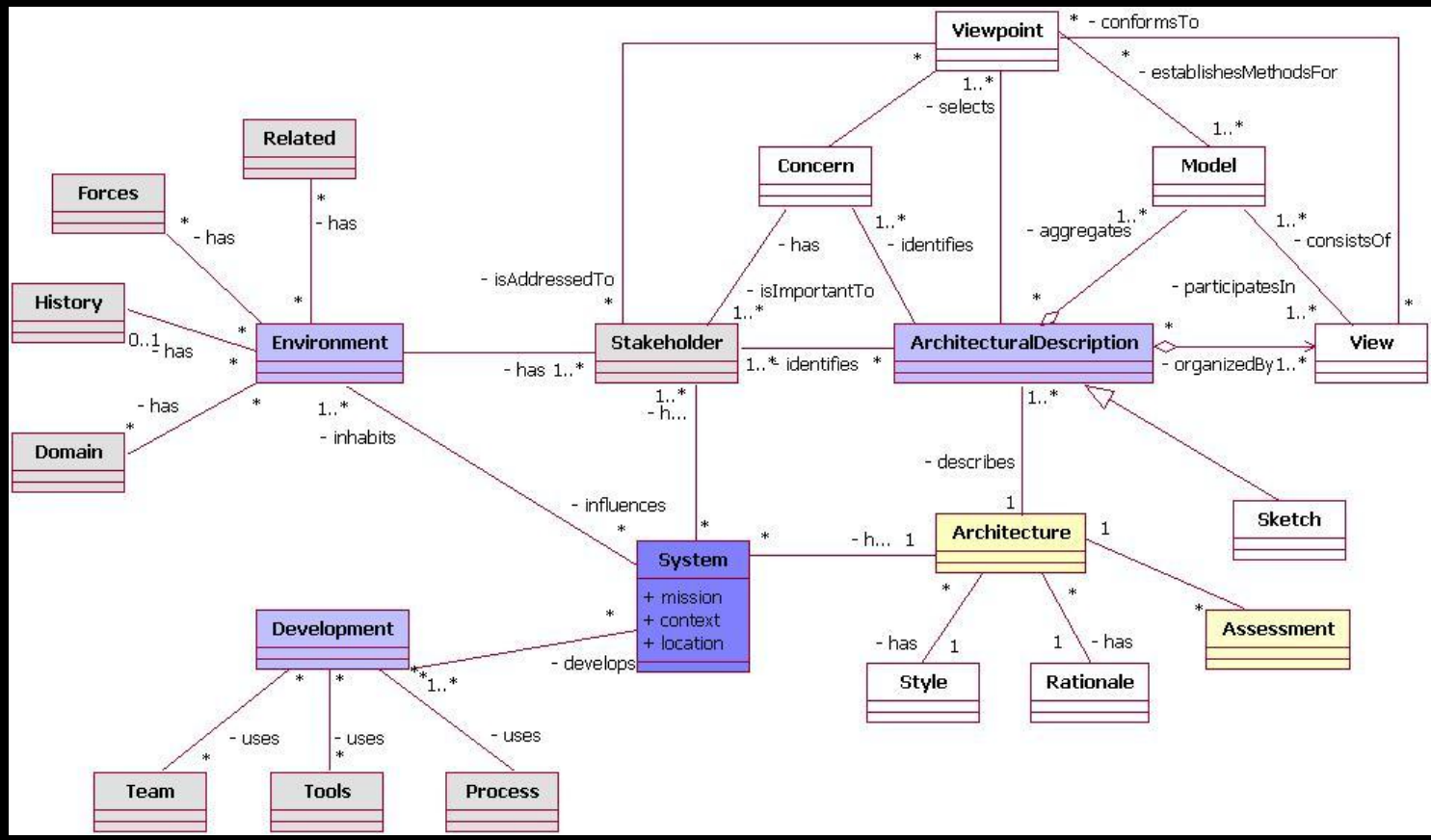
How do we attend to a continuously refreshed stakeholder community, a globally and temporally distributed development team, and inevitable leakage/loss of institutional memory?

While doing all this, how do we continue to innovate?

Forces

Business
Schedule
Cost
Mission
Resources
Legal
Values
Environment
Compatibility
Ethical/Moral
Complexity
Dependability
Production
Quality
Operations
Development
Resilience
Performance
Functionality

Architecture Metamodel



Open EA Frameworks

- Zachman
- The Open Group Architecture Framework (TOGAF)
- The DoD Architecture Framework (DoDAF)
- Federal Enterprise Architecture (FEA)

Proprietary EA Frameworks

Capgemini Integrated Architecture Framework (IAF)

IBM Architecture Description Standard (ADS)

SAP Enterprise Architecture Framework (EAF)

Oracle Enterprise Architecture Framework (OEAF)

Other Governmental EA Frameworks

European Interoperability Framework (EIF)

Standards and Architecture for eGovernment (SAGA)

Commission Enterprise Architecture Framework (CEAF)

Other Defense EA Frameworks

Ministry of Defense Architecture Framework (MoDAF)

NATO Architecture Framework (NAF)

Atelier de Gestion de l’archtectue des Systemes d’Infromation et de Communication (AGAGTE)

Other Proprietary EA Frameworks

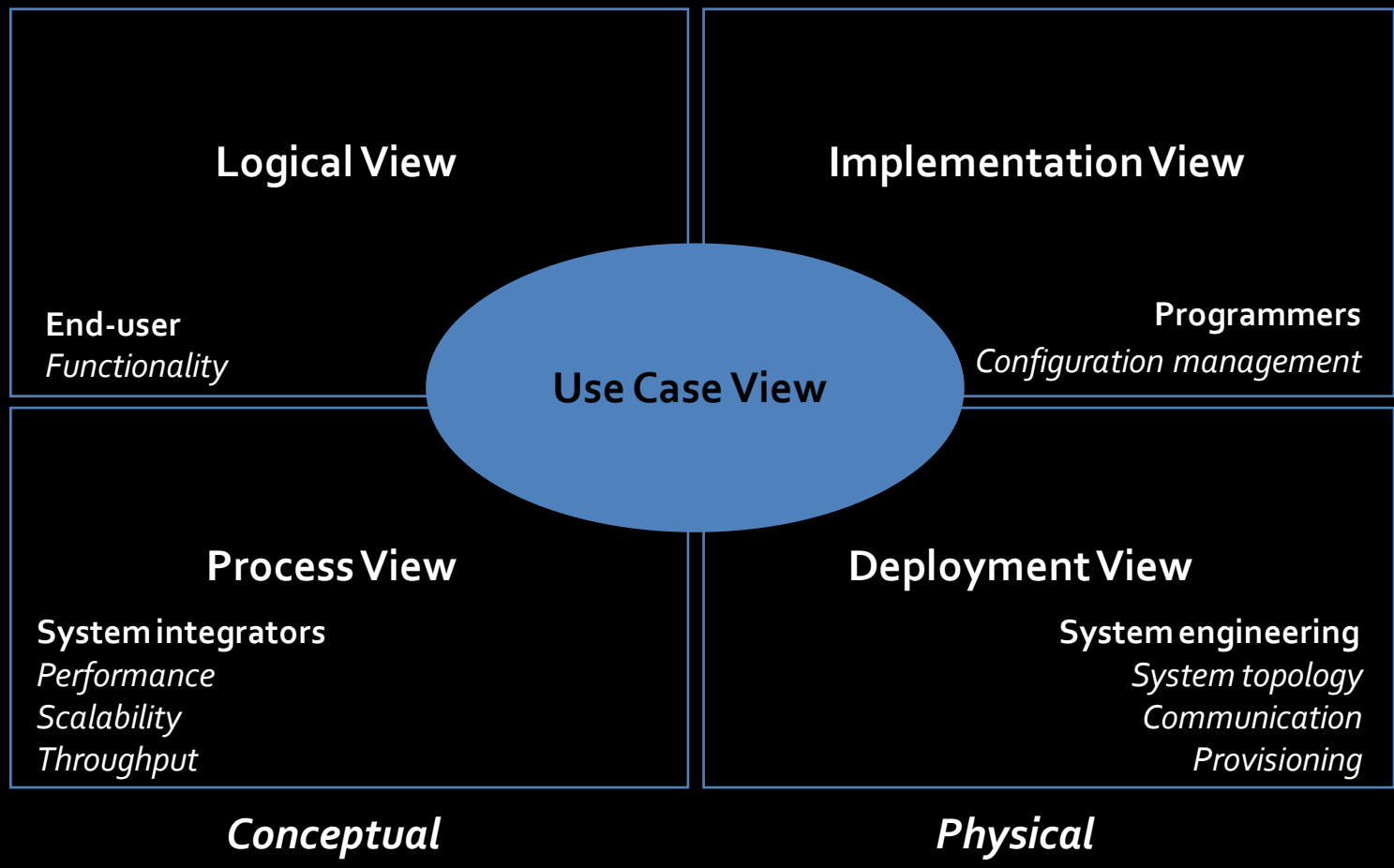
- Business Enterprise Architecture Modeling Framework (BEAM)
- Extended Enterprise Architecture Framework (E2AF)
- Pragmatic Enterprise Architecture (PeaF)
- Purdue Enterprise Reference Architecture (PERA)
- Gartner Enterprise Architecture Framework
- OBASHI

Technical Frameworks

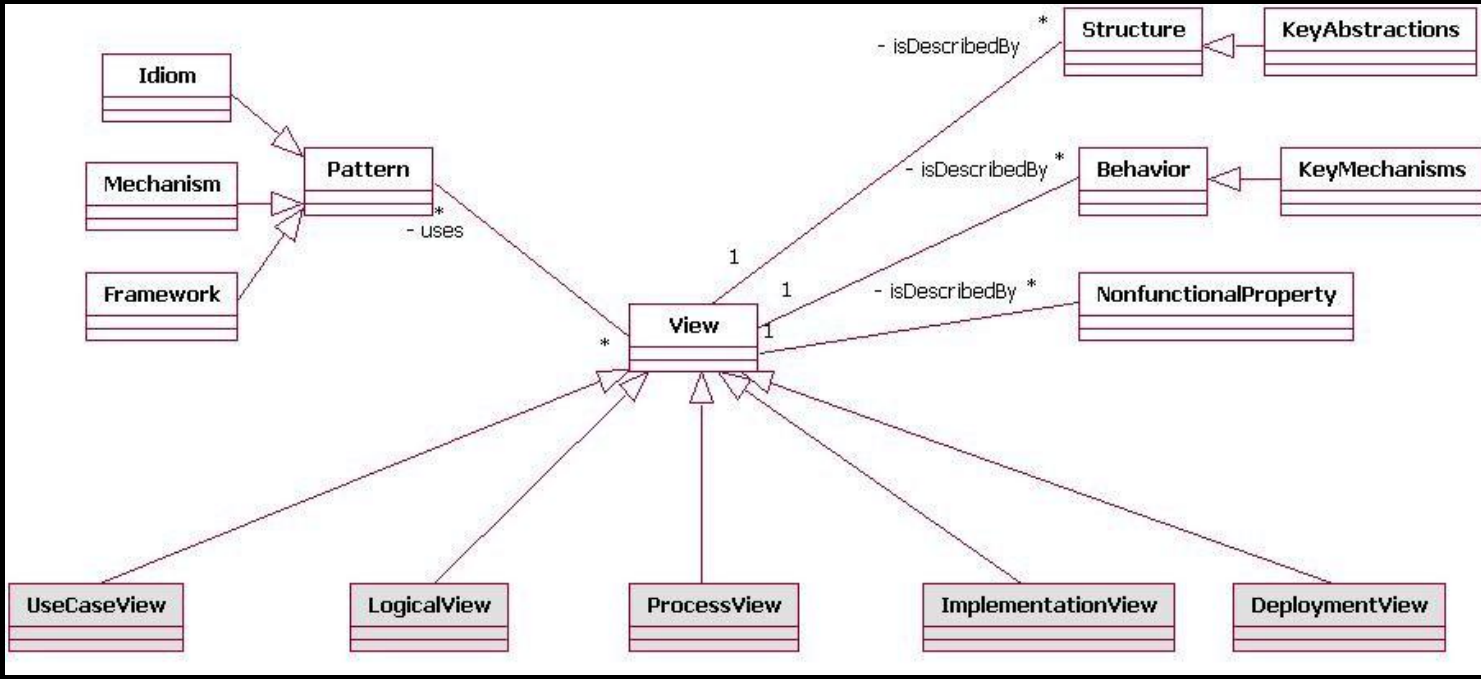
Kruchten 4+1

Rozanski and Wood's Viewpoints and Perspectives

Kruchten 4+1


Logical View
End-user
Functionality
Implementation View
Programmers
Configuration management
Use Case View
Process View
System integrators
Performance
Scalability
Throughput
Deployment View
System engineering
System topology
Communication
Provisioning
Conceptual
Physical

Kruchten 4+1


Idiom
Mechanism
Framework
Pattern
- uses
*
*
View
1
1
1
- isDescribedBy
- isDescribedBy
- isDescribedBy
*
*
*
Structure
KeyAbstractions
Behavior
KeyMechanisms
NonfunctionalProperty
UseCaseView
LogicalView
ProcessView
ImplementationView
DeploymentView

What Modeling Should Be

Abstraction with freedom but without ambiguity
Abstraction with focus
Artifacts at a moment in time
Artifacts across time and space
Artifacts for many stakeholders
Artifacts made manifest