

ArchView - Visualizing Multiple Evolution Metrics

Martin Pinzger
Department of Informatics
University of Zurich



University of Zurich
Department of Informatics



Context & Objectives

- Analysis of large complex software systems
 - Depict software modules and their dependency relationships
 - Size and complexity of modules
 - Strength of dependency relationships
 - Visually identify shortcomings in the design and the implementation
 - Bad Smells
 - e.g., God Module, Lazy Module, Dead Code, Cyclic Dependency
 - Hotspots of activity
 - Modules that changed often => unstable design

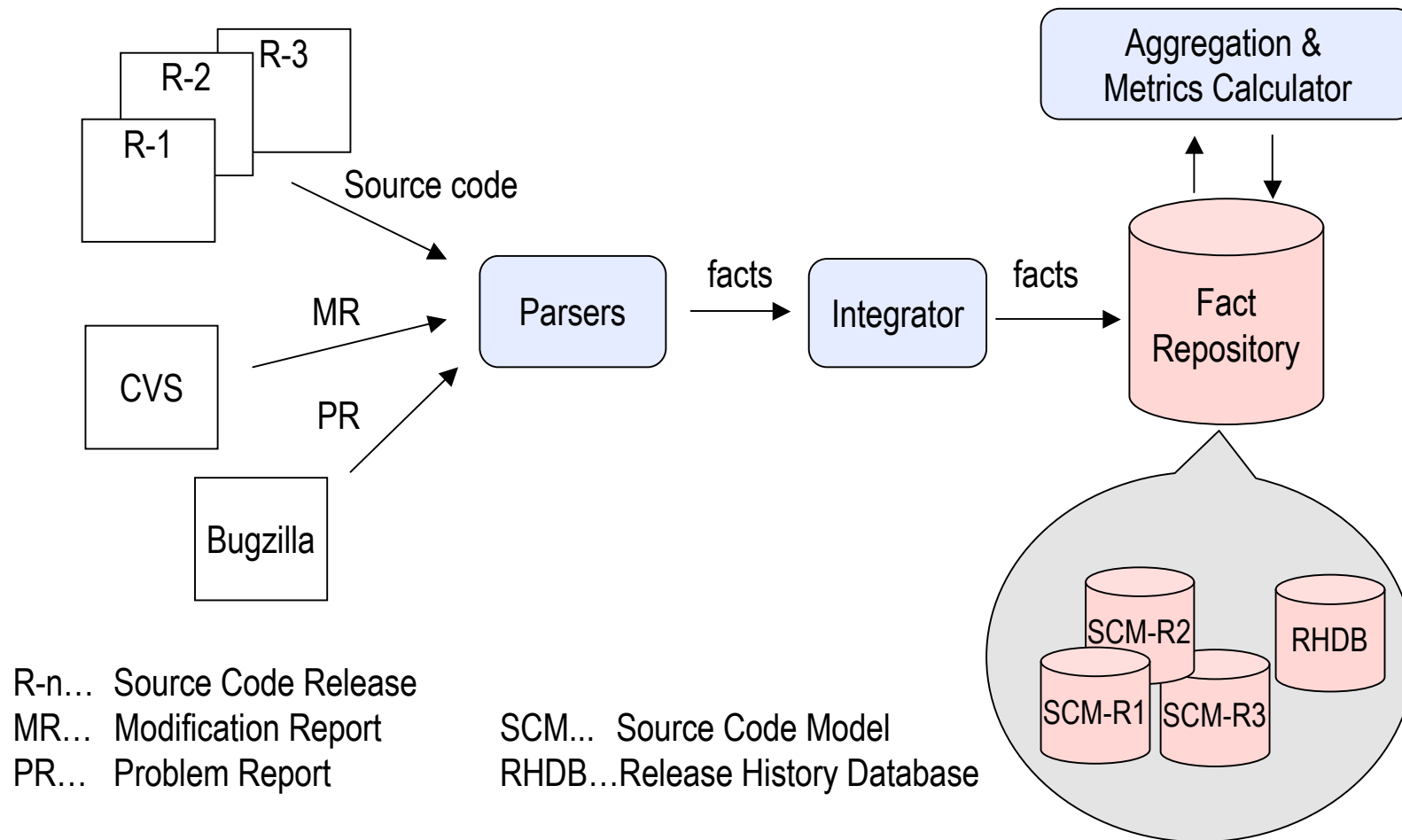
Outline

- Visualization
 - Kiviat Diagrams
 - Kiviat Graphs
- Demo

Basic Idea

- Take snapshots from **a number** of source code releases
- **Aggregate** information and **measure** size, complexity, and evolution metrics of modules and dependency relationships
- Compose **different views** that visualize modules and dependency relationships with metric trends

Data Processing



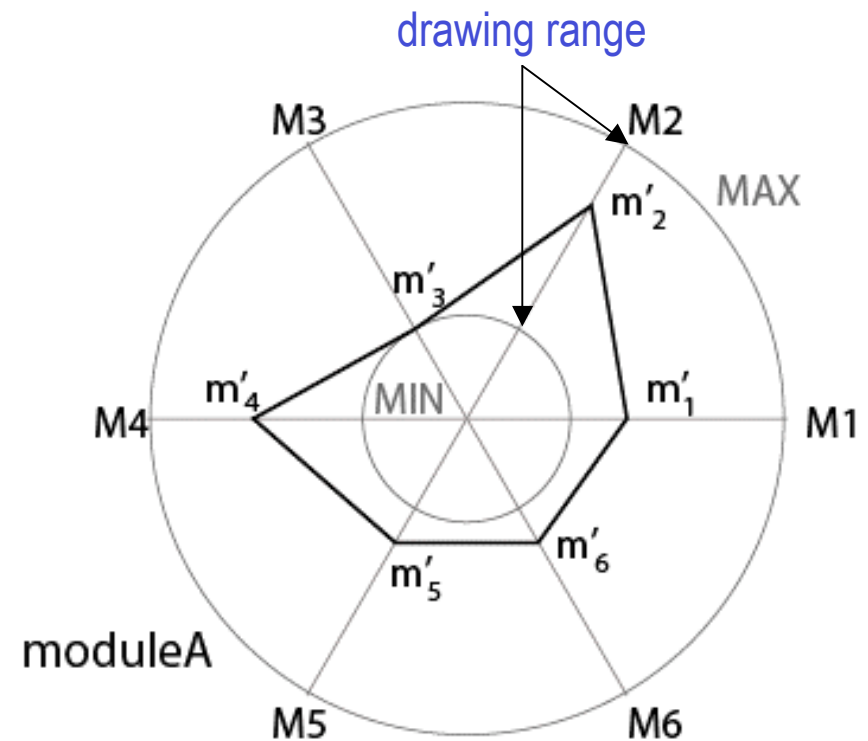
Evolution Matrix

- Basis for our calculations
 - i measures of one entity over n releases

$$E_{i \times n} = \begin{pmatrix} \begin{array}{c} \text{R1} \\ m'_1 \\ m'_2 \\ \cdot \\ \cdot \\ m'_i \end{array} & \begin{array}{c} \text{R2} \\ m''_1 \\ m''_2 \\ \cdot \\ \cdot \\ m''_i \end{array} & \dots & \begin{array}{c} \text{Rn} \\ m_1^n \\ m_2^n \\ \cdot \\ \cdot \\ m_i^n \end{array} \end{pmatrix}$$

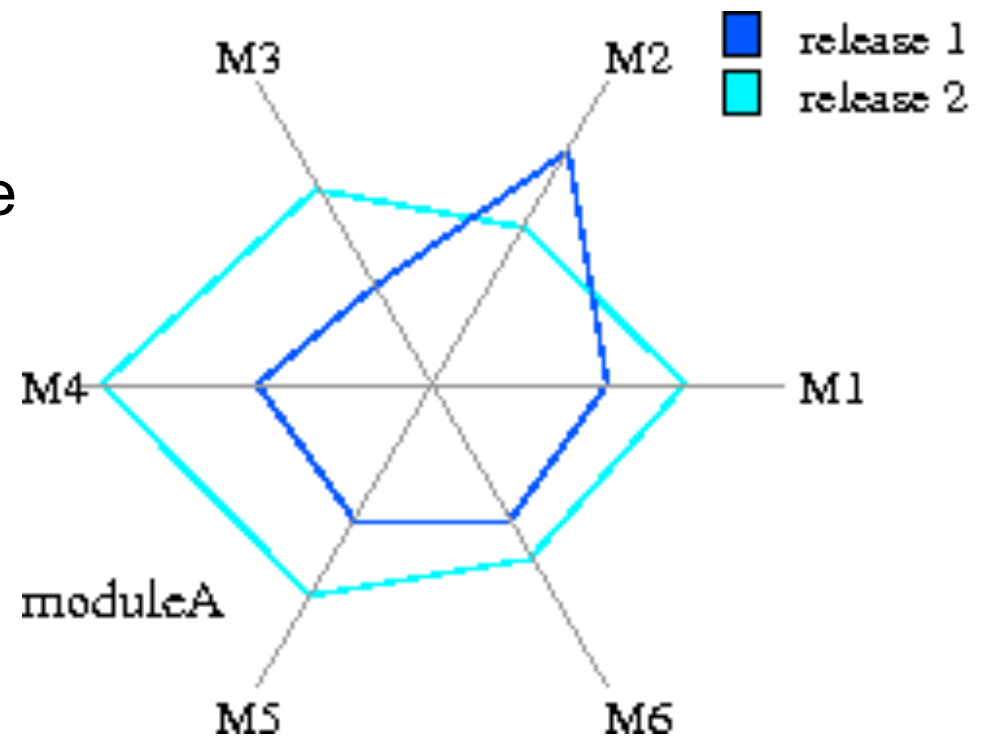
Kiviatic Diagrams (Radar Charts)

- Visualize *i* metric values per module
 - Normalize values to size of Kiviatic diagrams
 - Use offset to prevent cluttering in the center



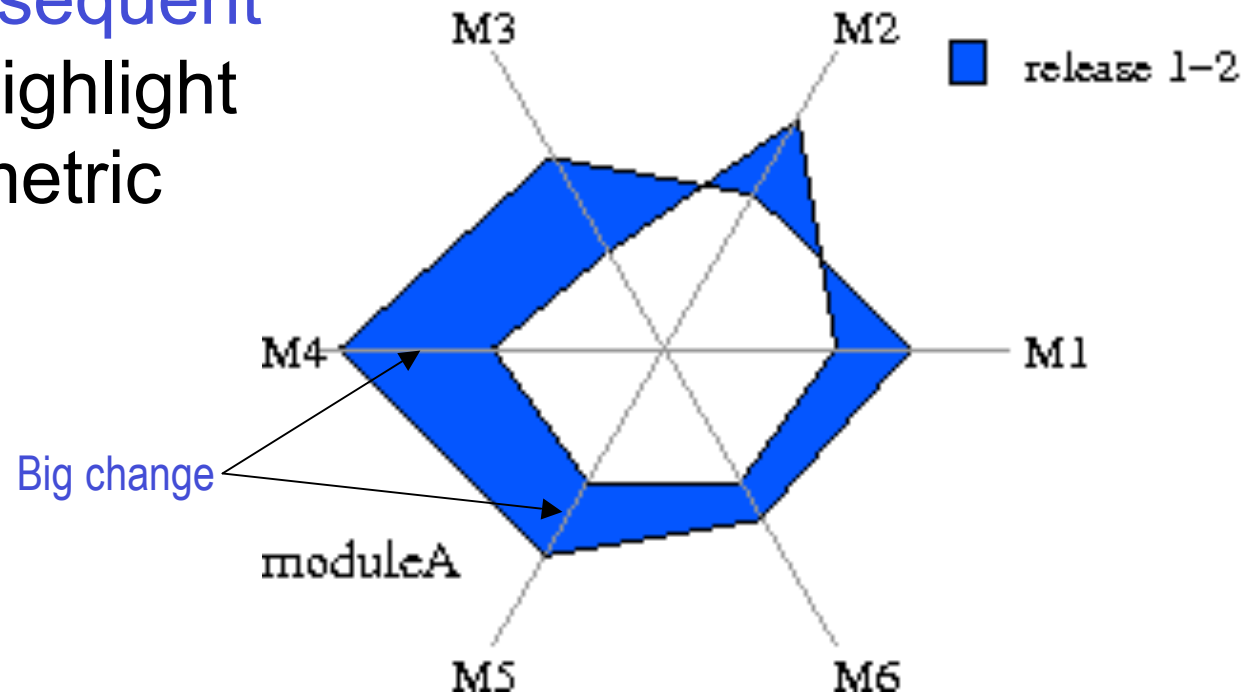
Visualizing Multiple Releases

- Visualize i metric values of n releases
 - Compute maximum per metric across the n releases



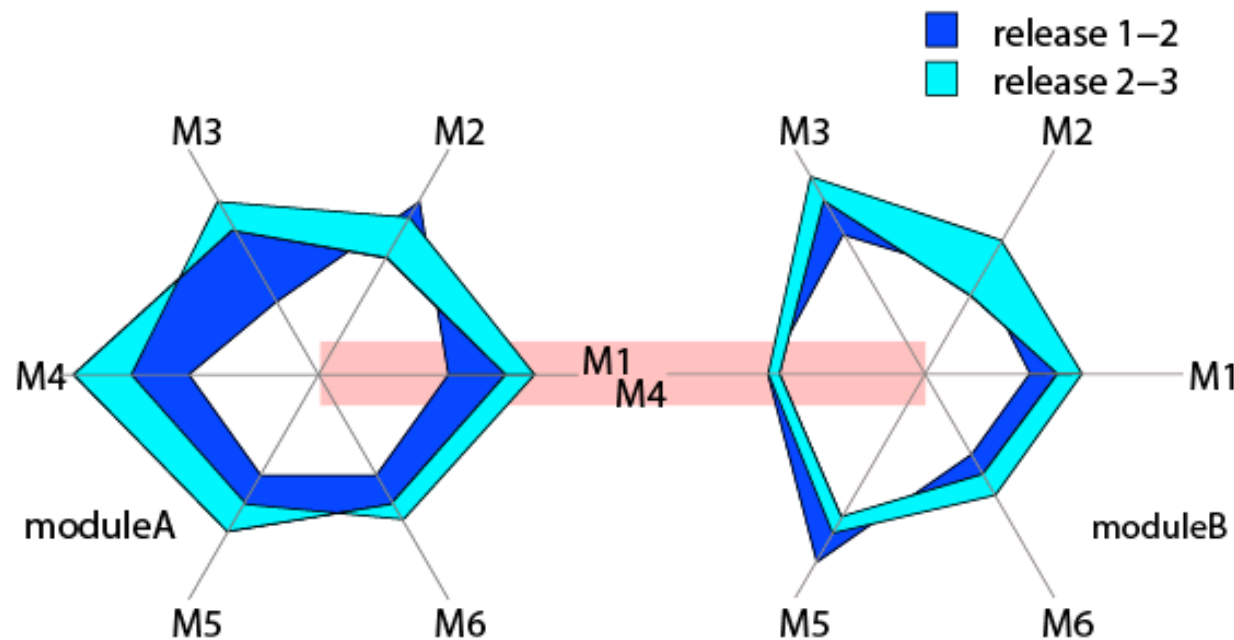
Visualizing Changes

- Fill polygons between subsequent releases to highlight changes of metric values



Kiviati Graphs

- Kiviati diagrams are connected by edges
 - Coupling dependencies between modules
 - Width denotes coupling strength



View Configurations

- System hotspots views
 - Highlight large and complex modules
- Modification hotspots views
 - Highlight change prone modules
- Coupling views
 - Source code coupling views
 - Change coupling views

Example

- Increase in earlier releases

0:nrPrio_undef

1:nrPrio_

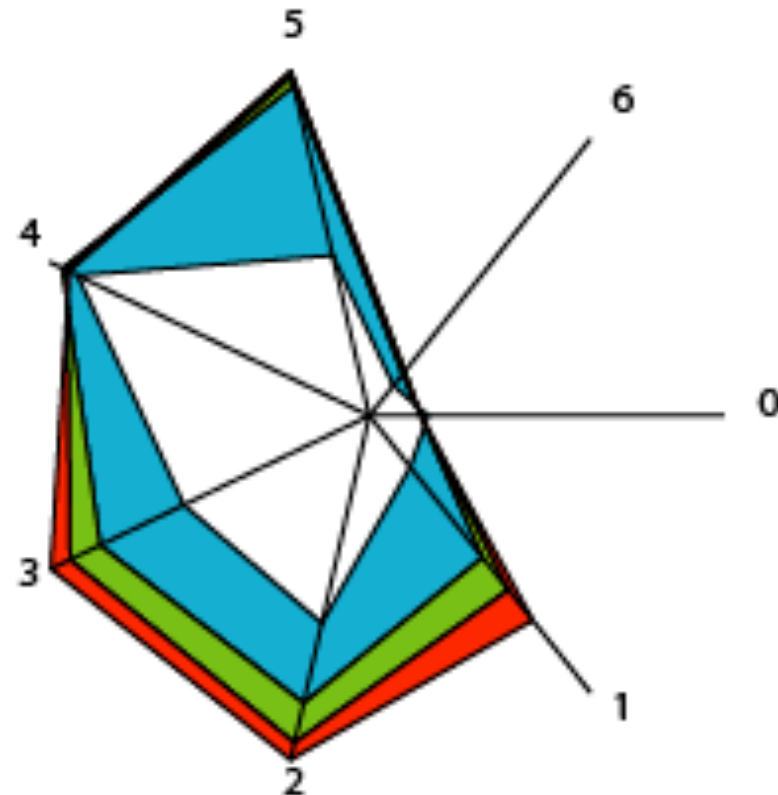
2:nrPrio_1

3:nrPrio_2

4:nrPrio_3

5:nrPrio_4

6:nrPrio_5

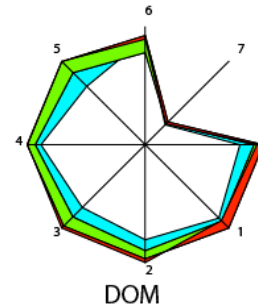
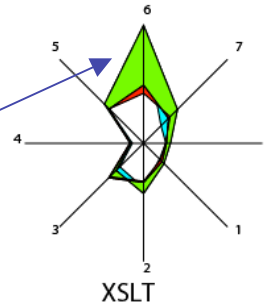
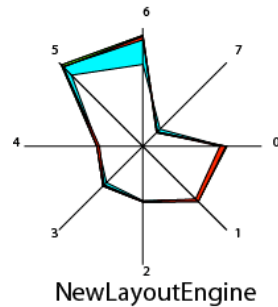
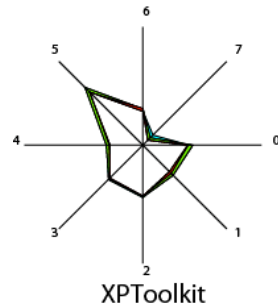
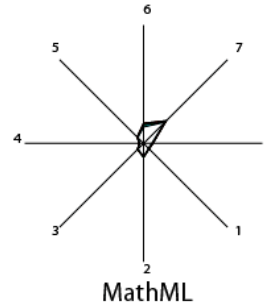


NewLayoutEngine

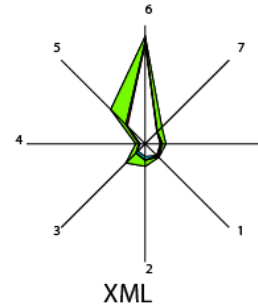
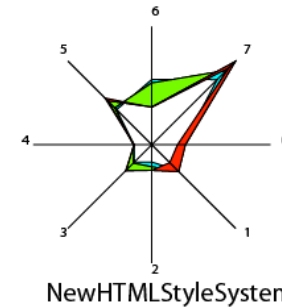
Size & Complexity Metrics

- 0:nrStmts
- 1:CCMPLX
- 2:nrFiles
- 3:nrClasses
- 4:nrMeths
- 5:nrAttr
- 6:nrGlobFuncs
- 7:nrGlobVars

Idle modules



Large, complex, growing module



Decrease because of change source tree

- release 0.92-1.0
- release 1.0-1.4
- release 1.4-1.7

Problem Report Metrics

0:nrPrio_undef

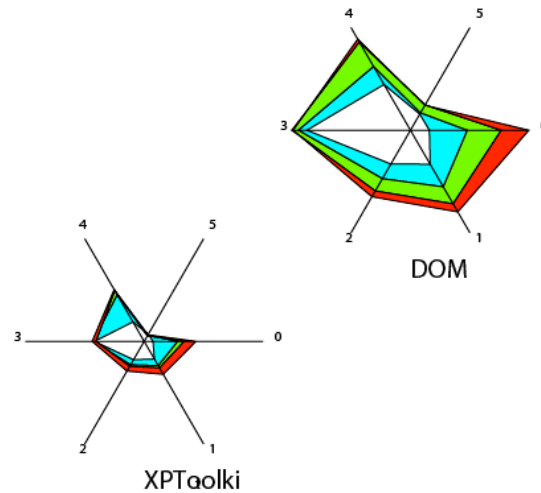
1:nrPrio_1

2:nrPrio_2

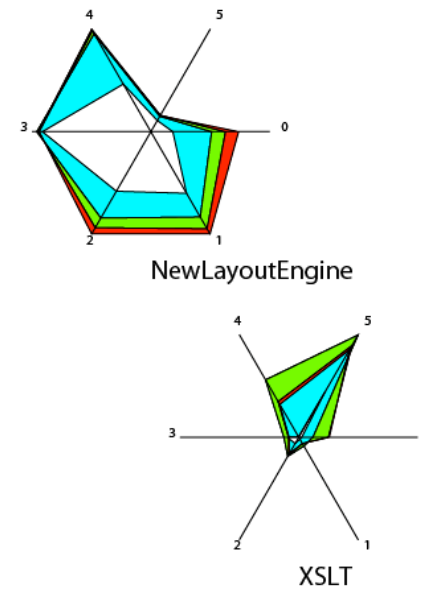
3:nrPrio_3

4:nrPrio_4

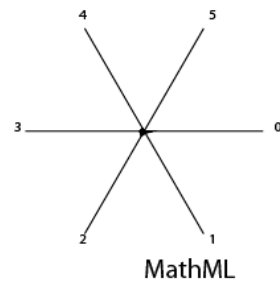
5:nrPrio_5



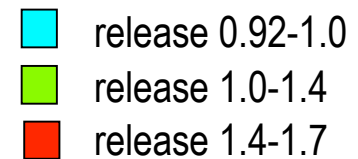
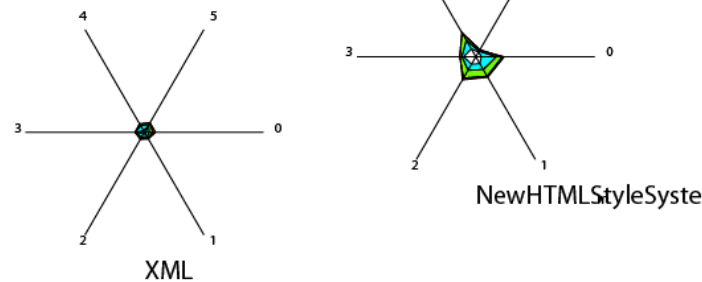
Change prone modules



Module with most "unimportant" problem reports

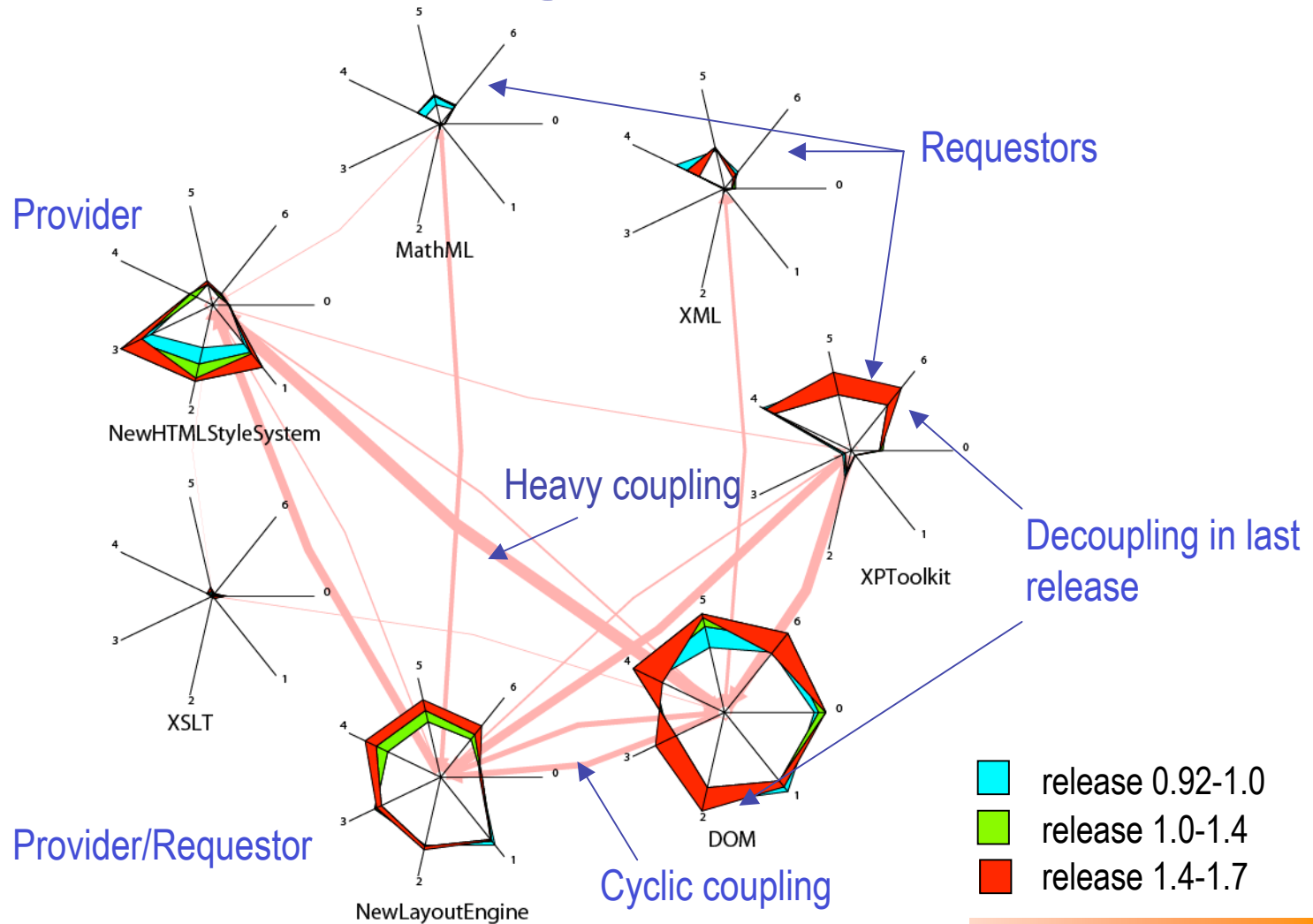


Idle modules



Module Coupling

- 0:nrFuncs
- 1:Fan-in
- 2:nrCF-in
- 3:nrCR-in
- 4:nrCF-out
- 5:Fan-out
- 6:nrCR-out



Conclusions

- ArchView facilitated abstraction of higher-level views on the implementation of software systems
 - Kiviat diagrams
 - Suited to visualize **multiple metric values** of n releases
 - **Trends** and strong changes of metric values were highlighted
 - Kiviat graphs
 - Showed the **coupling dependencies** between modules and indicated coupling strength
- **Visual** identification of design shortcomings?