

Diplomarbeit:

Entwicklung eines Eclipse Plugin zu Metrikbasierten Qualitätsanalyse von Softwaremodellen

Phillips Universität Marburg
Fachbereich: Mathematik und Informatik
Leitung: Prof. Dr. Taentzer
Diplomand: Pawel Stepień



Überblick

- Einführung
- EMF Henshin
- EMF Metrics - Aufbau
- EMF Metrics - Anwendung
- Fazit

Einführung

- **Softwarequalität**

- Es gibt zahlreiche Ansätze zur Messung der Qualität von Software.
- Softwaremetriken (kurz Metriken) sind mathematische Funktionen die gewisse Eigenschaften von Software auf einen Zahlenwert abbilden.
- Metriken bieten eine wohldefinierte und eindeutige Grundlage zur Qualitätsanalyse von Softwaresystemen.

- **Modellqualität**

- Entwurfsphase ist für die Kosten eines Projektes ausschlaggebend.
- In der Objektorientierten Softwareentwicklung ist eine Qualitätssicherung in frühen Entwicklungsphasen von zentraler Bedeutung.

- **Modellmetriken**

- Bilden die Qualitätseigenschaften von Softwaremodellen ab und können bereits in den Spezifikations- und Entwurfsphasen eines Softwareprojektes angewendet werden.

Graphtransformationen

Graphtransformationsregeln

- Es werden Eigenschaften auf die das Modell untersucht werden soll definiert.
- Die Regeln werden nicht angewendet, es wird nur die Anzahl der möglichen Anwendungen gezählt.
- Keine Veränderungen des Zielmodells.

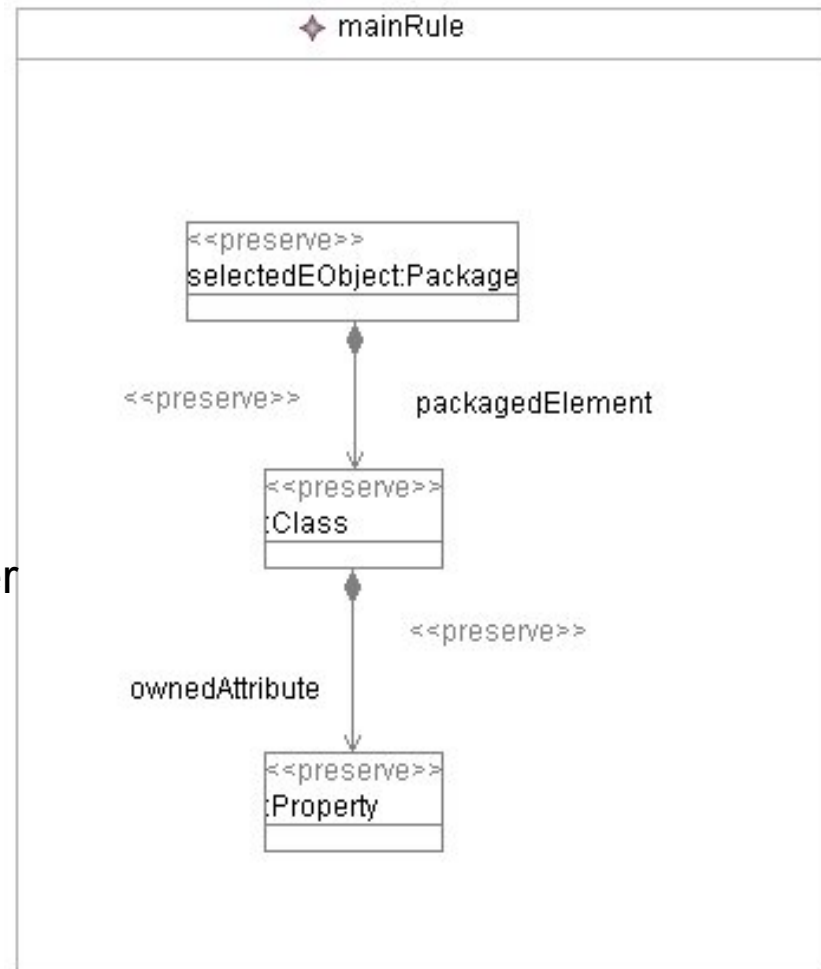
EMF Henshin

- Eclipse Plugin.
- Basiert auf Graphtransformationsregeln.
- Eignet sich sehr gut zur Definition von Metriken.
- Beinhaltet einen grafischen Editor in dem Transformationsregeln definiert werden können.

Henshin Transformationsregeln

„NumberOfAttributes“

- *Kontext muss berücksichtigt werden (hier: Package)*
- *SelectedEObject* als Parameter für das zu untersuchende Modell.
- Regel beinhaltet ebenso ein *SelectedEObject* Element.
- Der Wert der Metrik entspricht der Anzahl aller möglichen Anwendungen der Regel.

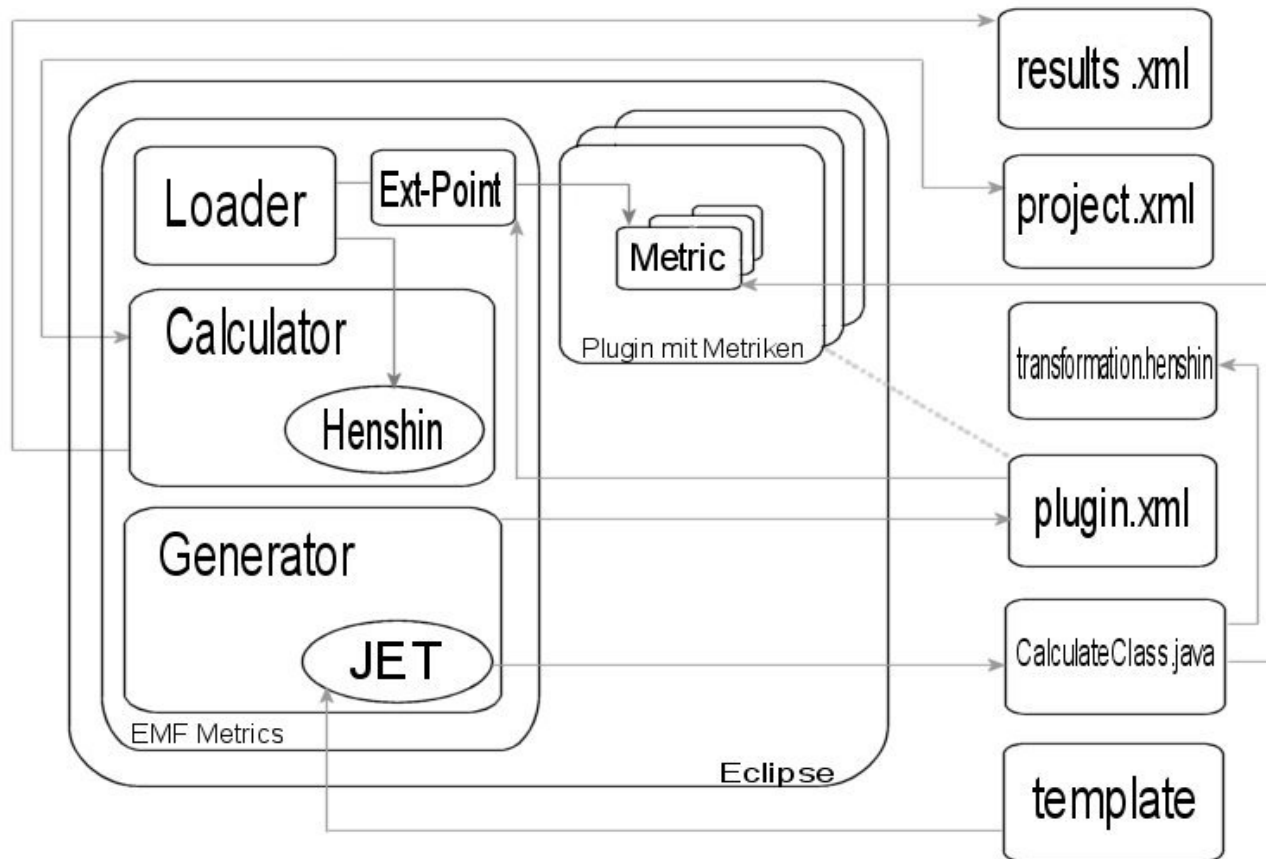


EMF Metrics - Aufbau

EMF Metrics Komponenten

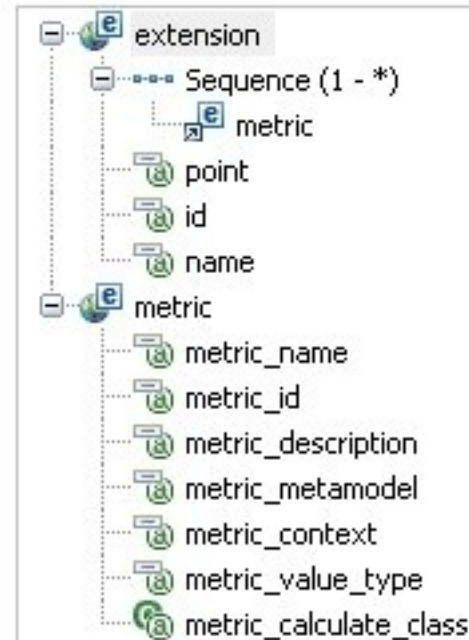
- Loader
- Calculator
 - EMF Henshin
- Generator
 - JET (Java Emitter Templates)
- Plugins mit Metriken
 - Extension-Point

EMF Metrics - Architektur

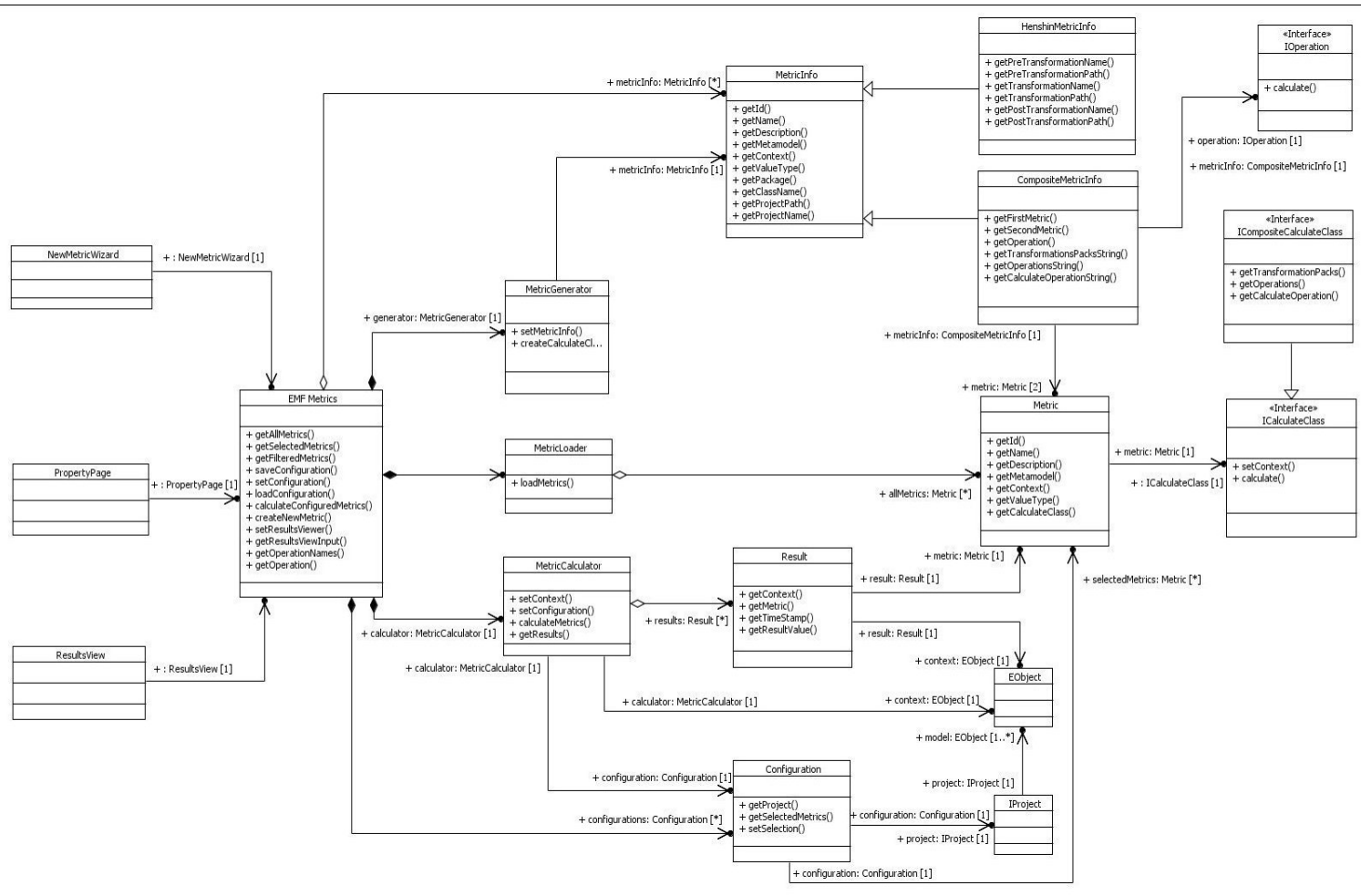


Extension Point

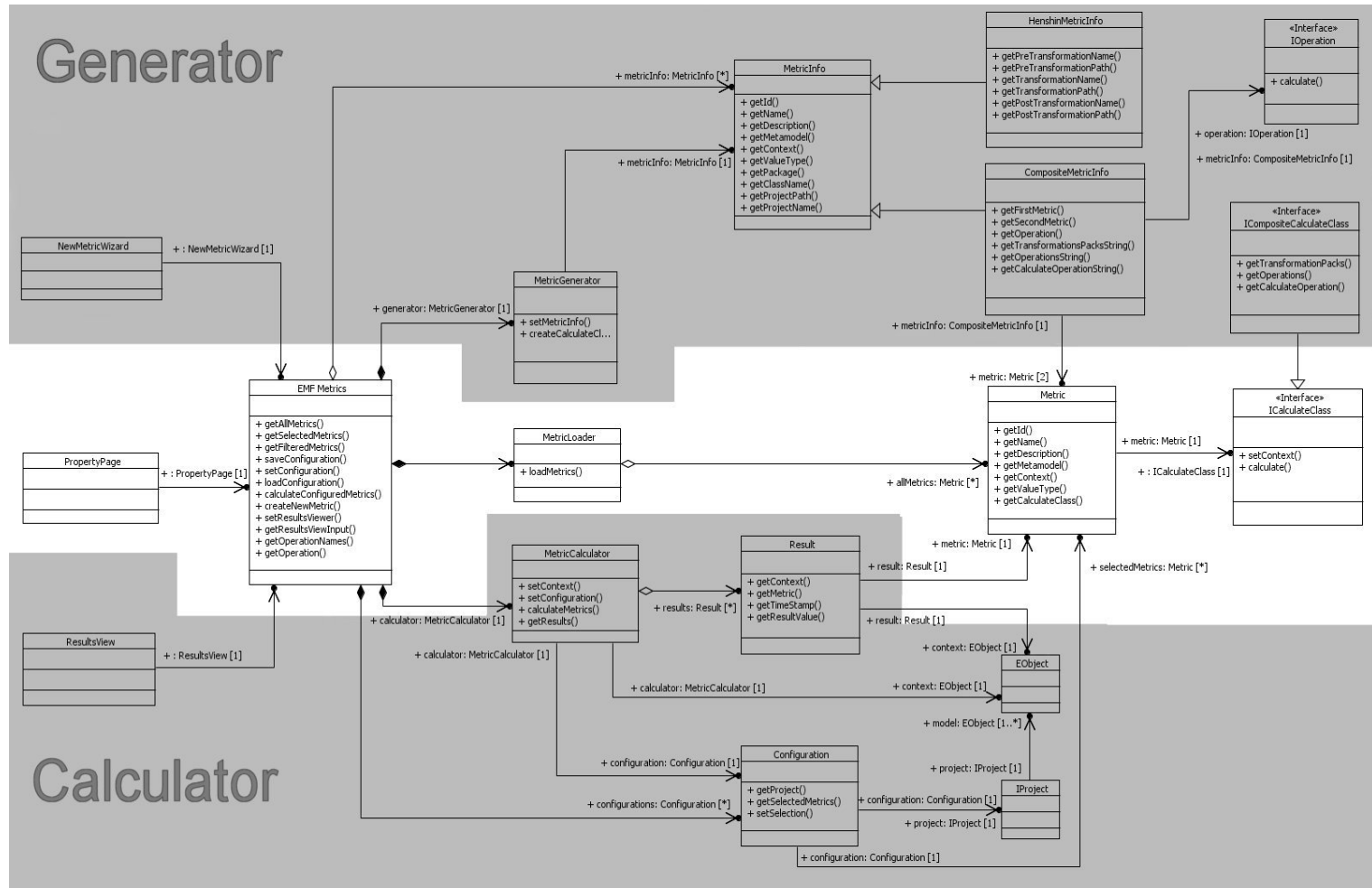
- **metric_id** - eine eindeutige Identifikation der Metrik
- **metric_name** - der Name der Metrik
- **metric_description** - die Beschreibung der Metrik
- **metric_metamodel** - das Metamodell für das die Metrik berechnet werden soll
- **metric_context** - der Kontext in dem die Metrik berechnet werden soll
- **metric_value_type** - der Typ des Wertes den die Metrik als Ergebnis liefern wird
- **metric_calculate_class** - ein Verweis auf die *ICalculateClass* implementierende Klasse die für die Berechnung der Metrik zuständig ist



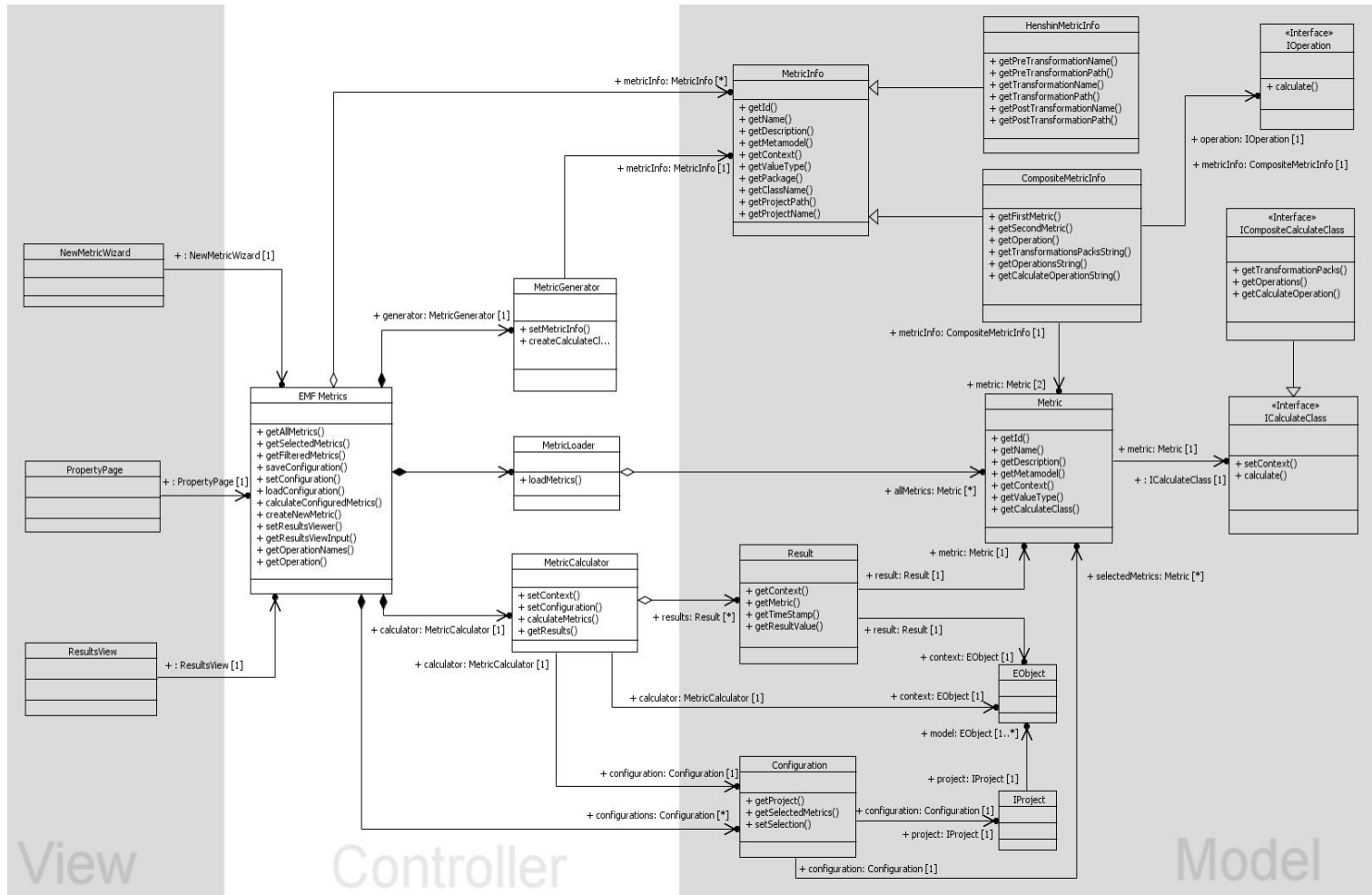
EMF Metrics - Klassenstruktur



EMF Metrics - Klassenstruktur



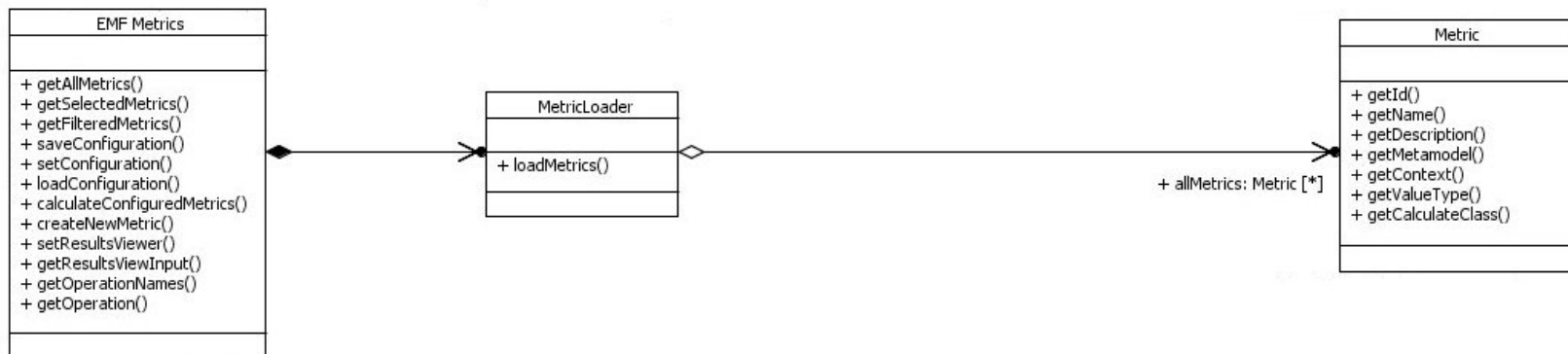
EMF Metrics - Klassenstruktur



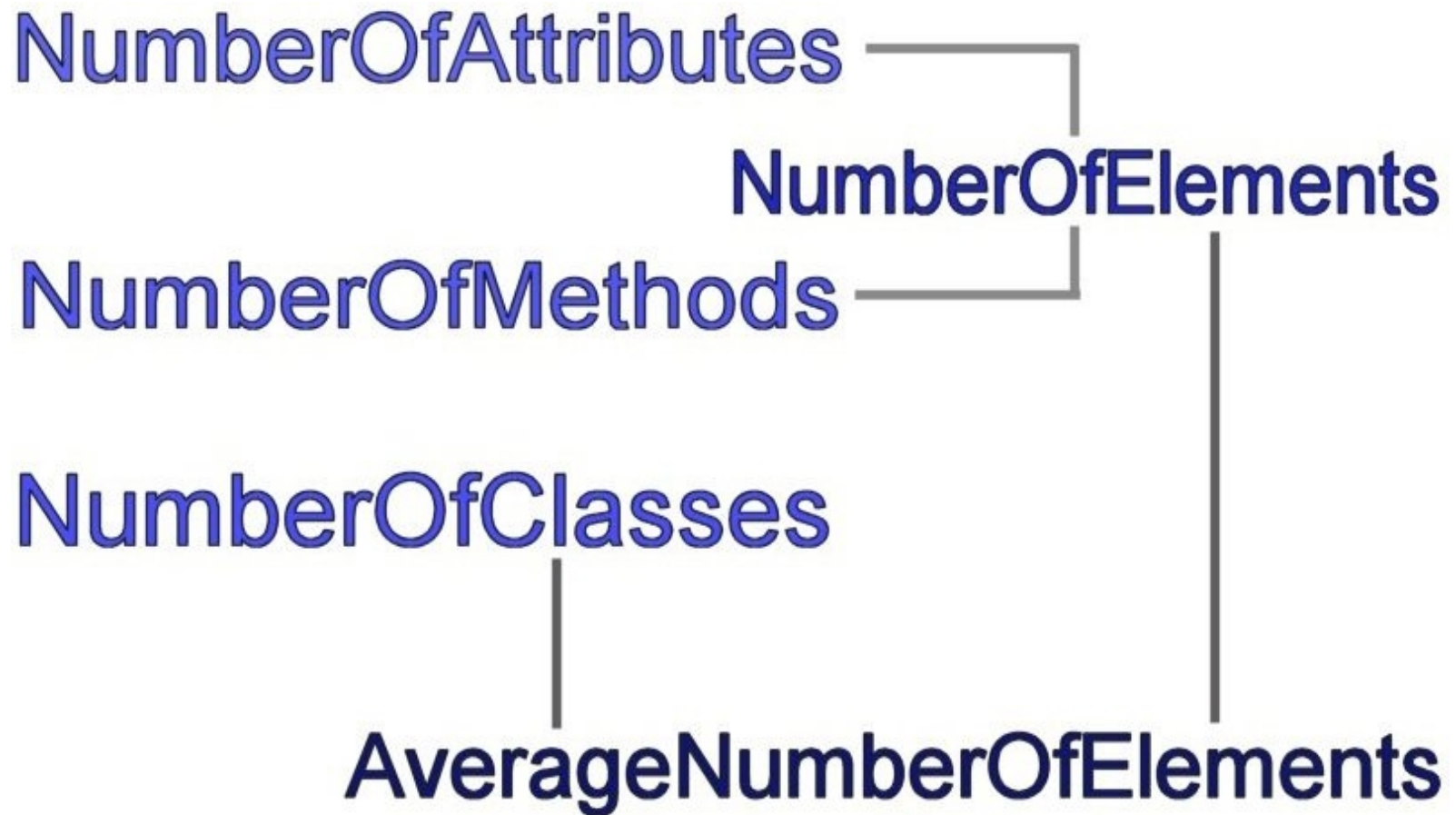
EMF Metrics - Klassenstruktur

Loader

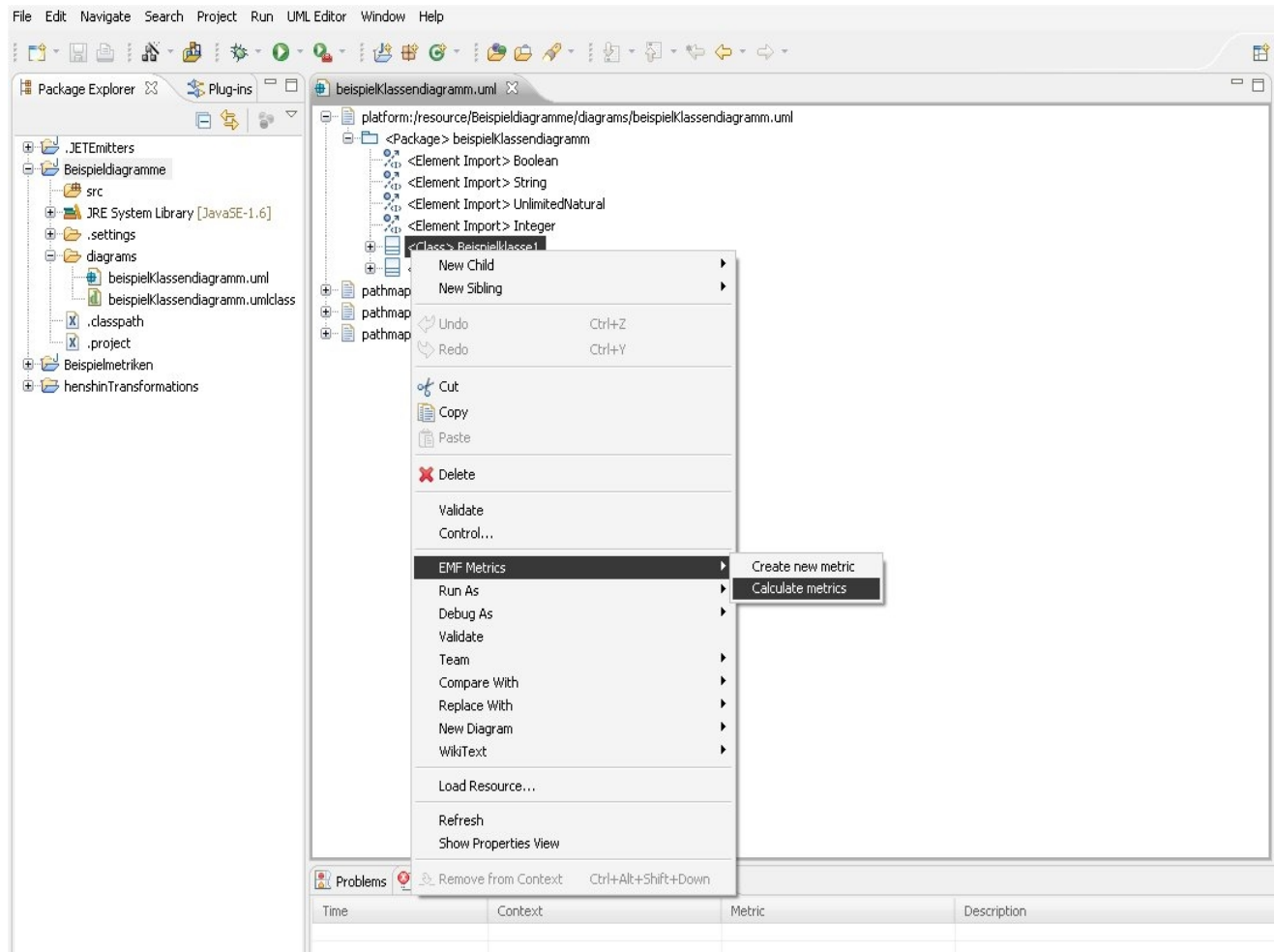
- Hauptkontroller: *EMFMetrics*.
- Hilfskontroller: *MetricLoader*.
- „Composite“ Klasse: *Metric*.
- Wird von beiden anderen Komponenten benutzt.



EMF Metrics - Anwendung



EMF Metrics - Anwendung



Erreichte Ziele

- Ein benutzerfreundliches Werkzeugpaket zur Qualitätsanalyse von Softwaremodellen.
- Transformationsregeln ermöglichen durch ihre Ausdrucksstärke die Definition von beliebigen Modellmetriken.
- Modellmetriken durch den grafischen Editor von EMF Henshin unkompliziert zu erstellen.
- Zahlreiche offene Schnittstellen zur Erweiterung.

Fazit

Erweiterungsvorschläge

- Mehrere Plugins
- Transformationsdateien
- Iteration als neue Operation
- Weitere Metriken

Ausblick

- EMF Refactor
- Model smells