



Philipps-Universität Marburg
Fachbereich: Mathematik und Informatik
Diplomand: Pawel Stepien
Leitung: Prof. Dr. Gabriele Taentzer
WS 2009/2010

Entwicklung eines Eclipse Plug-In zu Metrikbasierten Qualitätsanalyse von Softwaremodellen

Lastenheft

1. Produktname: *EMF Metrics*

2. Zielbestimmungen

Das Plug-In soll Eclipse um eine benutzerfreundliche Umgebung für Qualitätsanalyse von Softwaremodellen basierend auf dem Eclipse Modelling Framework (EMF) anhand von Metriken erweitern.

Dem Benutzer soll eine Palette an fertigen Metriken für UML2-EMF-Modelle zur Verfügung stehen.

Des Weiteren soll der Benutzer die Möglichkeit haben selber Metriken für EMF basierte Modelle zu spezifizieren und den Code zur Metrikenberechnung automatisch zu generieren.

3. Produkteinsatz

Benutzergruppe: Softwaremodellierer, Reviewer, Metrikendesigner.

Das Plug-In soll weltweit genutzt werden und erfordert deshalb Englisch als Verkehrssprache.

Das Plug-In soll schließlich eine Komponente eines großen Modell-Qualitätsanalyse Paketes bilden.

Es soll aber auch ohne die weiteren Plug-Ins funktionieren.

4. Produktfunktionen

4.1. Definition von Metriken für Metamodelle auf EMF.

Für die Spezifizierung der Softwaremetriken soll es die folgenden zwei Möglichkeiten geben:

4.1.1. Beschreibung durch Modelltransformationen.

Hierzu soll die EMF Modell-Transformations-Werkzeug: „EMF Henshin“ benutzt werden. EMF Henshin ist als ein weiteres Eclipse Plug-In realisiert und kann somit direkt aufgerufen werden. EMF Henshin unterstützt direkte Transformationen von EMF Modellinstanzen wie auch Generierung von Instanzen einer Zielsprache aus einer gegebenen Instanz einer Quellsprache. Dem Benutzer steht sowohl ein baumbasierter (EMF) als auch ein grafischer (GMF) Editor zur Verfügung.

4.1.2. Zusammensetzung von vorhandenen Spezifikationen.

Hierzu soll erstmal ein Metrikenkatalog mit allen verfügbaren Metriken angezeigt werden. Der Benutzer soll aus diesem Katalog die für die Zusammensetzung gewünschten Metriken auswählen können. Des weiteren soll hier ein Operator zur Verbindung von den markierten Metriken ausgewählt werden. Der Operator soll auch aus einer Liste ausgewählt werden können.

4.1.3. Speicherung der Metrik

Die erstellte Metrik soll über ein Extension Piont abgespeichert werden können.

4.1.4. Generierung des Codes zur Metrikenberechnung.

Der Code zur Metrikenberechnung soll automatisch generiert werden.

4.1.5. (kann Anforderung) Generierung eines JUnit Testgerüsts für den Metrik-Code.

Für die Fälle 4.1.1. und 4.1.2. soll ein Dialogfenster mit den folgenden Eingabefeldern auftauchen:

- Name der Metrik.
- Beschreibung.
- Angabe des Metamodells für das die Metrik angewendet werden kann.
- Angabe des Metriken-Projektes in das der Code generiert werden soll. Hier soll über ein Dateidialogfenster, der Ort an dem gespeichert werden soll, angezeigt werden.
- Verweis auf die Datei mit der Modelltransformation (4.1.1.). Diese soll über ein Dateidialogfenster ausgewählt werden.
- Information über Kontext in der Form eines Dropdown-Menüs.
- Wertebereich (z.B. als Radiobutton)

4.2. Berechnung der Metriken auf bestimmten Modellen

Dazu sollen dem Benutzer folgende Funktionen zur Verfügung stehen:

4.2.1. Laden von Metriken

Die Metriken sollen über ein Extension Point angebunden werden. Dieser soll bei der Programminitialisierung aufgerufen werden.

4.2.2. Anzeigen von Metriken

Es soll möglich sein über das Property-Menu des Projektes eine Liste von Verfügbaren Metriken anzuzeigen. Die Metriken sollen nach Metamodellen gruppiert werden. Des weiteren soll eine Untergruppierung nach dem Kontext der Metriken erfolgen.

4.2.3. Metriken für Berechnung auswählen

Der Benutzer soll im Property-Menu aus der Liste aller verfügbaren Metriken die von ihm gewünschten auswählen können.

4.2.4. Metriken berechnen

Die Metrikenberechnung soll über das Kontextmenü eines Modellelementes im baumbasierten EMF Instanzmonitor ausgeführt werden können.

Die entsprechenden Berechnungen sollen Kontextentsprechend erreichbar sein (nur die Metriken die für den Kontext sinnvoll sind).

4.2.5. Berechnungsergebnisse anzeigen

Die Ergebnisse der Metrikenberechnungen sollen dem Benutzer in einer Tabellarischen (und eventuell auch Grafischen – *kann* Anforderung) Form in einer eigenen Metriken-View angezeigt werden.

4.2.6. Berechnungsergebnis abspeichern

Es soll möglich sein die gesamten Berechnungsergebnisse zu speichern. Dies soll in der Metriken-View über einen Button geschehen.

5. Produktdaten

Die Metriken sollen separat (unabhängig vom Speicherort des Metrik-Plugins)gespeichert werden. Sie sollen über ein Extension Point eingeladen werden können. Der Extension Point soll automatisch bei der Plug-In Initialisierung aufgerufen werden.

Die projektspezifische Metrikenauswahl (siehe 4.2.3) soll in einer separaten Datei abgespeichert werden

Das Berechnungsergebnis für eine Metrikberechnung soll extern gespeichert werden können. Es muss in einer Datei im Textformat gespeichert werden können. Als *kann* Anforderung wird noch ein PDF und ein JPEG (für die grafische Darstellungsform) Format gewünscht.

6. Qualitätsanforderungen

Da es sich um eine Diplomarbeit handelt wird ein besonders großer Wert auf ausführliche Dokumentation gelegt.

Gute Erweiterbarkeit sollte gewährleistet sein.

Testfälle für das Plug-In sowie für die fertigen Metriken sollen vorhanden sein.

Es wird eine Modellbasierte Vorgehensweise erfordert.

7. Realisierung

Als Programmiersprache wird Java verwendet.

Das Plug-In soll in die Eclipse Galileo modelling version eingebunden werden.

Als Modell-Transformations-Werkzeug soll EMF Henshin benutzt werden.