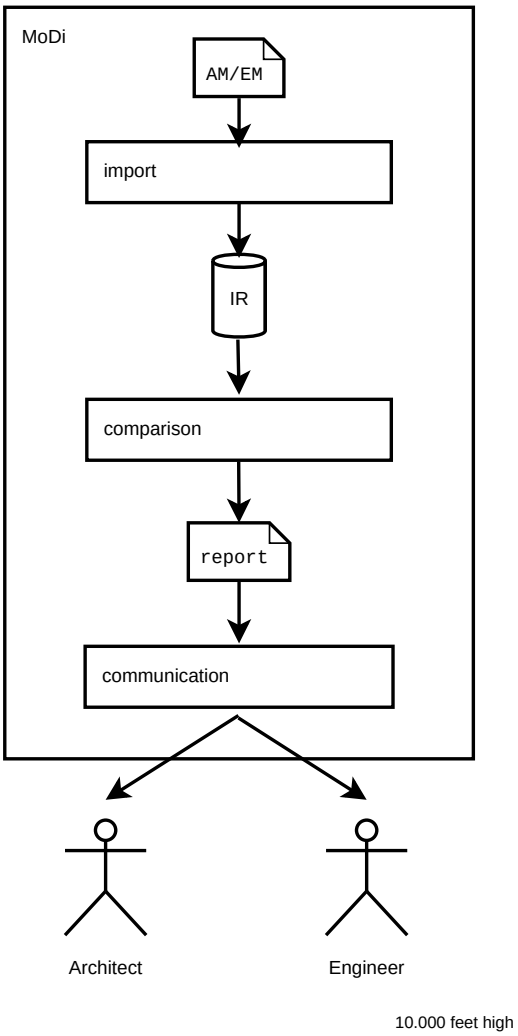
1 Architektur

Dieses Dokument beschreibt die Architektur für `MoDi`, ein Tool zur kontinuierlichen Qualitätssicherung von implementierten Architekturmodellen.

1.1 Grobarchitektur

Die folgende Grobarchitektur gibt einen Überblick über das System.



Das System muss aus einem Referenzmodell und einem Implementationsmodell eine Interne Repräsentation erstellen. Anschließend müssen in der Internen Repräsentation Modelldifferenzen erkannt und in einem Report zusammengefasst werden.

Die zu vergleichenden Modelle liegen als Codebasen (AM / EM) vor und werden durch die erste Komponente (import) in eine interne Repräsentation (IR) überführt.

Auf dieser Basis kann nun ein Modellvergleich stattfinden, bei dem die vorkonfigurierten Regeln ausgewertet werden (comparison). Hierzu werden die Eigenschaften der zu vergleichenden Modelle von der internen Repräsentation abgefragt (IR). Die Ergebnisse des Vergleichs werden bezüglich ihres Schweregrades analysiert und in einem Bericht (report) zusammengefasst.

Der Bericht wird an die beteiligten Rollen geschickt und soll die nötige Kommunikation einleiten (communication).

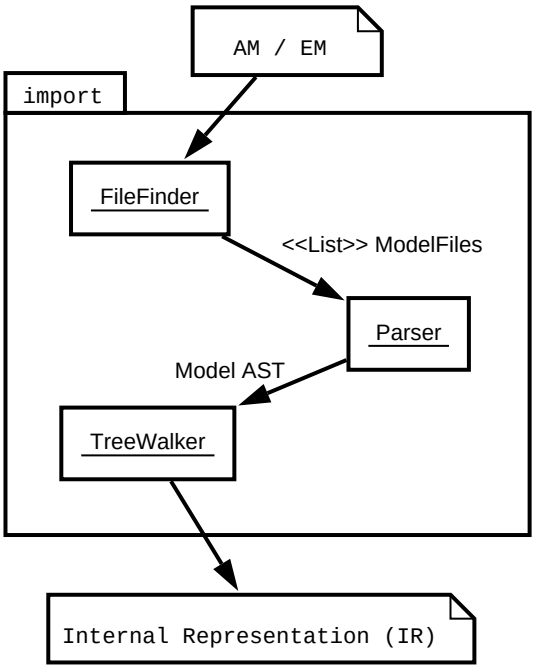
1.2 Feinarchitektur

1.2.1 Komponente: import

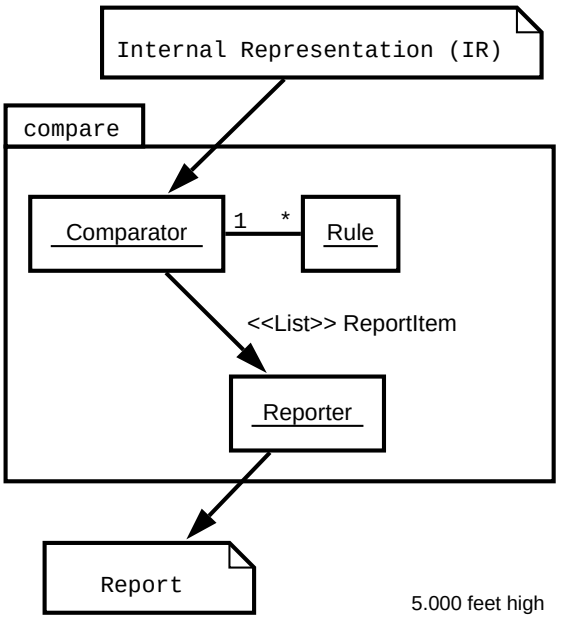
Für die `import`-Komponente liegen das Architekten- und das Entwicklermodell als Quelltext in entsprechenden Ordnern vor. Die Ordner sind im System zu konfigurieren.

Ein `FileFinder`-Objekt findet zunächst alle relevanten Dateien. Anschließend wird die entstehende Datei-Liste an einen `Parser` übergeben, der für jede Datei einen *Abstract Syntax Tree* (AST) erstellt.

Der so entstehende Wald wird zu einem Model-AST zusammengefasst (MAST). Ein `TreeWalker` traversiert den MAST, liest die relevanten Informationen aus und speichert sie in der Internen Repräsentation (IR).



1.2.2 Komponente: compare



In der `comparison`-Komponente werden nun die in der Internen Repräsentation gespeicherten Modelle verglichen und ein Bericht über den Vergleich erstellt.

Dazu wertet der `Comparator` jede vorkonfigurierte Regel (`Rule`) aus und meldet dem `Reporter`-Objekt die aufgetretenen Regelverstöße (die Regel wird zu `falsch` evaluiert) als sog. `ReportItem`s.

Das `Reporter`-Objekt sammelt schließlich alle Regelverstöße und fasst sie in einem Gesamtbericht zusammen.

1.2.3 Komponente: communication

An die Vergleichskomponente schließt sich eine `communication`-Komponente an.

Diese reicht den Bericht an einstellbare Rollen weiter. Die Kommunukationskomponente kann dann den Bericht entsprechend aufarbeiten, also `ReportItems` nach Schweregrad und Rollenassoziation sortieren bzw. gruppieren, um so spezialisierte Berichte anfzuertigen.

Die Berichte werden denn an den Architekten und den Entwickler geschickt, um so die bei auftretenden Modelldifferenzen nötig werdende Kommunikation zu initiieren.