Entwurfsbeschreibung

Thema: SemanticChess
Gruppe: swp13-sc

Verantwortlich: Sascha Hildebrandt,
Stefan Wetzig, Gérard Treptow,
Erik Körner

Letzte Änderung: 08.04.13

Inhaltsverzeichnis

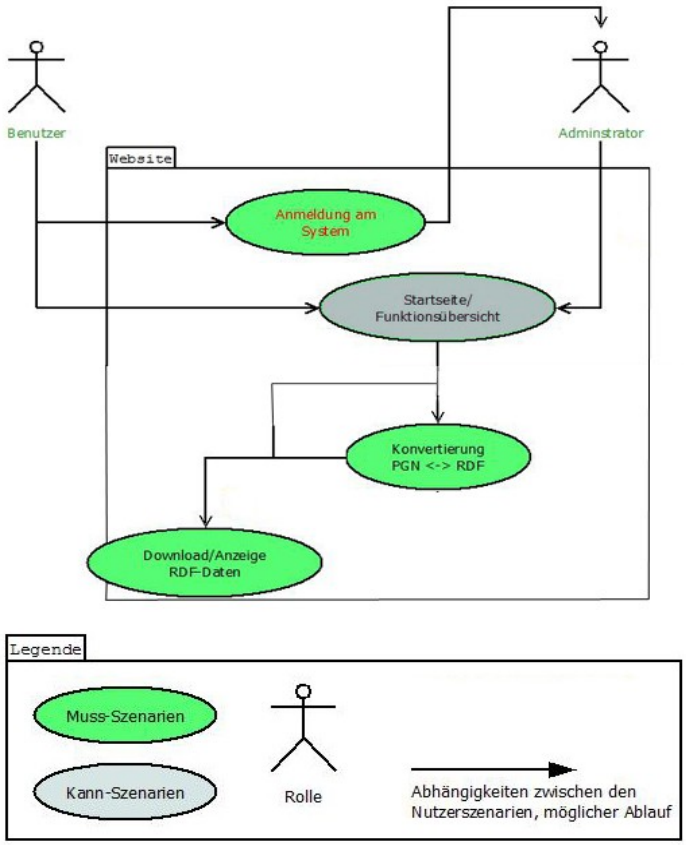
1 Allgemeines

Dieses Softwareprodukt soll dazu dienen, mithilfe einer Open-Source-Schachengine und einer eigens konzipierten Web-Oberfläche, Schachpartien zu simulieren bzw. durchspielen zu lassen.
Die Partien sollen hierbei nach bestimmten Kriterien auswertbar sein. Die Softwarestudie soll dabei über die Durchführbarkeit und das Zusammenspiel der wichtigsten Komponenten einer solchen Software Aufschluss geben. Hierfür soll als Prototyp eine geeignete Oberfläche, eine Schachontologie und eine Konvertierungsvorschrift (PGN zu RDF) erstellt werden. Dies soll später in Verbindung mit der Schachengine und einer geeigneten Suchmaske bzw. Suche und Auswertung die fertige Software darstellen.

2. Produktübersicht

Der Schwerpunkt der Funktionalität des Prototypen liegt im Großen und Ganzen auf der Konvertierung der PGN-Daten in ein RDF-Format mit Hilfe von Jena, sowie einer Vorabversion der Weboberfläche,welche in Vaadin geschrieben wird und zu großen Teilen durch Platzhalter dargestellt werden soll. Der Nutzer hat also die Möglichkeit PGN-Daten per copy-paste oder fileupload dem Server zu übergeben, diese Daten werden daraufhin vom Parser zur Weiterverarbeitung zurecht geschnitten und nach der Konvertierungsvorschrift in RDF-Daten übertragen.

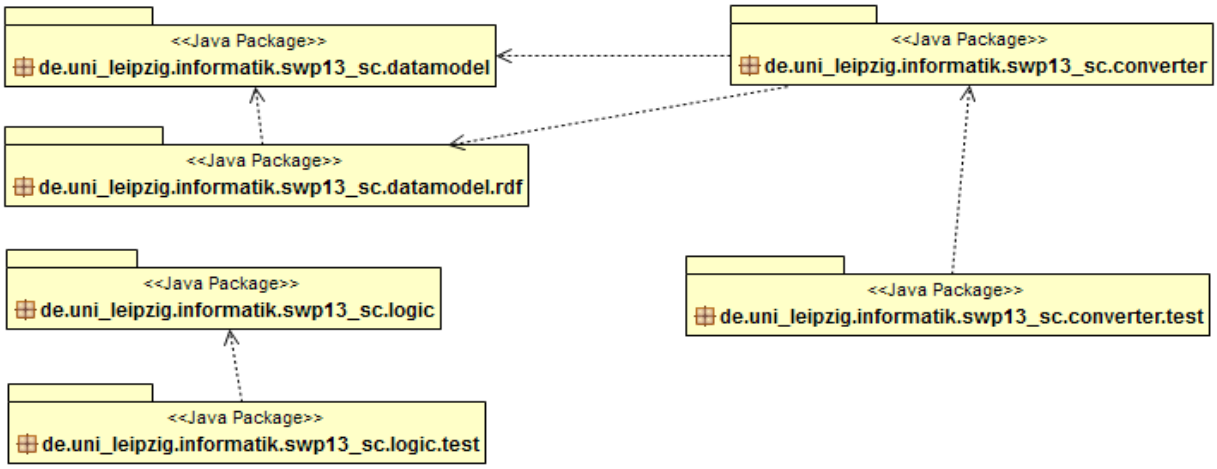
Use-Case-Diagramm Prototyp:



3 Struktur und Entwurfsprinzipien

Die Anwendung, samt GUI soll im Web als Java Servlet spezifiziert und implementiert werden.
Dies wird über den Open Source Webserver Apache Tomcat realisiert, welcher bereits
in den zur Verfügung stehenden Server integriert wurde. Im Allgemeinen besteht der Prototyp aus einer GUI (Benutzeroberfläche), mit welcher der Client in seinem Browser interagieren kann,
einer Schachontologie, welche Informationen zu verschiedenen Schachpartien im RDF bereit hält,
einer mit Virtuoso erstellten Datenbank und einem Konverter, welcher PGN-Daten in RDF-Form umwandelt. Für den Konverter wird ein Parser benötigt, der die mitgegebenen PGN-Daten in konvertierbare Stücke teilt.
Der Aufbau dieser Komponenten wird in den Punkten 4 und 5 näher erläutert.

Paketmodel des Projekts (logische Trennung des Codes ohne GUI):

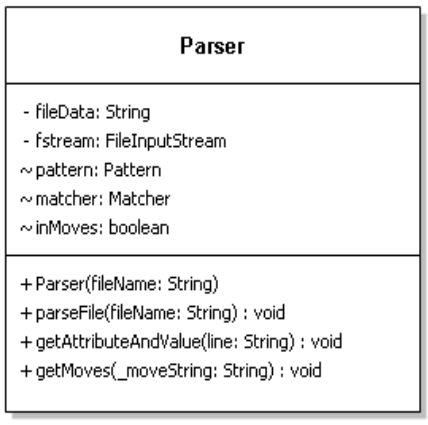


4 Struktur und Entwurfsprinzipien einzelner Pakete

(Pakete werden hier wie im Lastenheft beschrieben gekennzeichnet)

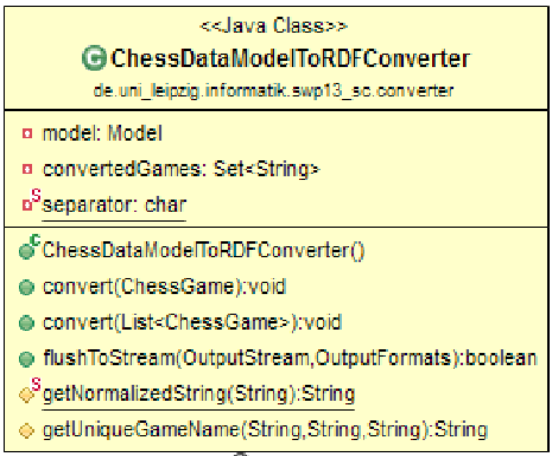
Konzipierung einer Schachontologie in RDF (1.M.1);
Es wurden mehrere Ontologien erstellt. Die erste stellte sich als zu umfangreich und zu spezifisch heraus, so dass eine zweite einfacher gehaltene Ontologie erstellt wurde. Grundsätzlich fasst sie Schachspiele als eine Sequenz von Zügen auf, wobei Züge Symbole, Kommentare und Namen haben können und Spiele Ergebnisse und weitere relevante Daten besitzen können. Die Ontologie soll nun im weiteren Projektverlauf iterativ wachsen – und kann auch noch nach Abschluss des Projektes iterativ erweitert werden. Für die Erstellung wurde das freie Softwaretool Protégé benutzt. Dieses ermöglicht den Export der Ontologie in verschiedene im Web gebräuchliche Formate, wie z. B. Turtle, XML/RDF.

Erstellen einer Konvertierungsvorschrift um PGN in RDF zu wandeln(1.M.2):
Eine der Hauptfunktionen neben dem Durchspielen der Partien an sich ist das Konvertieren von PGN nach RDF und auch von RDF in PGN-Format zurück. Dieser Vorgang geschieht in 2 Schritten. Zuerst wird die zu konvertierende Datei dem System per Fileupload oder Copy-Paste mitgegeben. Die Datei wird dann vom Parser in konvertierbare Stücke zerlegt und an den Konverter weitergegeben. Zu aller erst muss die mitgegebene PGN-Datei dem Parser übergeben werden
Dieser Schritt ist für die weitere Verarbeitung der Daten zwingend notwendig.

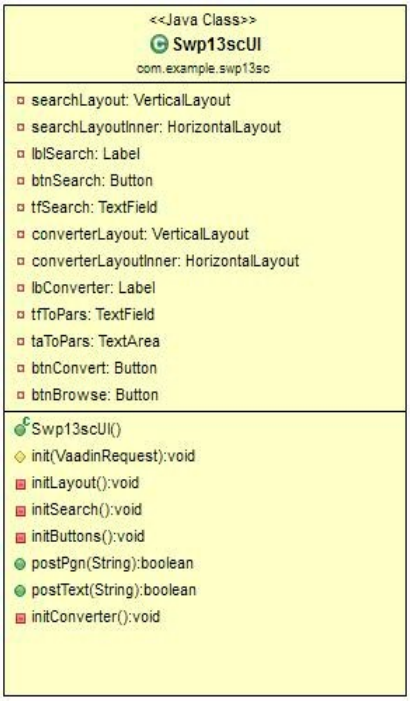


Konvertierungsvorschrift PGN – RDF(1.M.2):
Zunächst wird die mitgegebene PGN-Datei dem Parser übergeben. Dieser stellt die PGN-Daten als Speicherdaten für den Computer bereit, so dass Letzterer damit arbeiten kann. Die geparsten Daten werden nun mit Hilfe von Jena konvertiert, so dass die Daten letztlich in einem RDF-Format vorliegen. Bei der Speicherung der Dateien in Triple Stores wird der Name der Ressource nomalisiert damit daraus eine gültige URI gebildet werden kann.

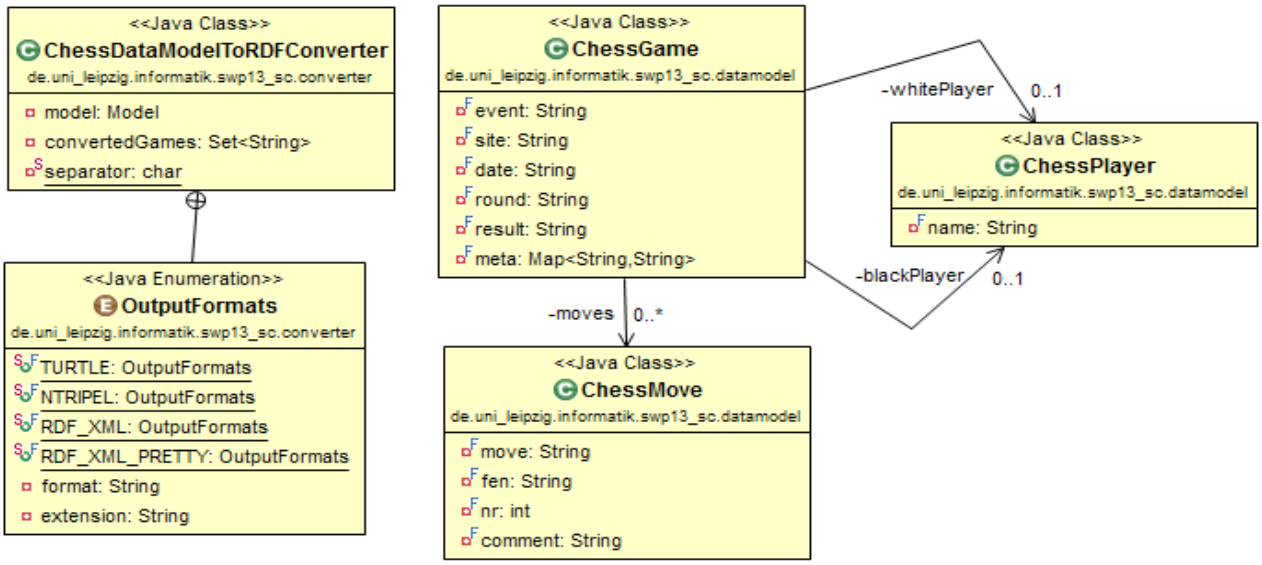
z.B. : göämaxö. , muster >http://.../resources/g__max____muster_5



Weboberfläche(5.M.*):
Die Weboberfläche soll so übersichtlich und einfach bzw. komfortabel wie möglich gestaltet werden.
Um dies zu erreichen soll mit Hilfe von Vaadin eine in Java geschriebene WebGUI aufgesetzt und mittels des Tomcatservers visualisiert werden.



5 Datenmodell



6 Testkonzept

Die automatischen Tests werden während der Implementierung in der Entwicklungsumgebung angezeigt und stellen weniger Probleme dar, da sie vom Programmierer nahezu akut gelöst werden können. Den manuellen Tests muss hingegen größere Aufmerksamkeit geschenkt werden. Während der Programmierung sollten einzelne Komponenten auf Funktionstüchtigkeit getestet werden. Z.B. sollte jede Klasse eine eigene „Testmain“ mit Ein- und Ausgabe enthalten, außerdem sollte dabei auf Grenzfälle wie verschiedene Datentypen, unvorhergesehene Eingaben, geachtet werden. Beim Testen muss auf die syntaktische und semantische Funktionalität geachtet werden. Wenn alles getestet und zusammengefügt wurde, dann gilt es, die Gesamtheit auf Funktionstüchtigkeit zu testen. Hierbei ist es von großer Bedeutung, dass viele Leute testen können, die den Quellcode nicht einmal gesehen haben. Somit hat man die Möglichkeit, mit vielen Testern schnell eine Liste von Bugs zu erstellen, die von den Programmieren dann beseitigt werden sollen. In unserem Beispiel können in den Zweierteams von Programmierern die automatischen Tests voraussichtlich gut in der Entwicklungsumgebung gehandhabt werden. Die gesamtheitlichen Test werden andererseits sehr viel Zeit in Anspruch nehmen. Folgende Fragen müssen im Zusammenspiel der Systemkomponenten analysiert werden:

Wurde die Ontologie auch genau so konvertiert wie geplant?
Ist das Nutzerinterface fehlerfrei?
Ist das Nutzerinterface benutzerfreundlich?
Funktionieren alle Schnittstellen zwischen Server, Datenbank und Client wie geplant?

Mit mehreren Testdurchläufen und Updates wollen wir versuchen alle Fragen mit einem signifikantem „Ja“ zu beantworten.

7 Glossar

Ontologie
Explizite formale, maschinenlesbare Spezifikationen einer Konzeptualisierung (Beziehungen, Ableitungsregeln) zu einem bestimmten Wissensbereich bezeichnet man als Ontologien. Diese ermöglichen es ein Netzwerk von Informationen mit logischen Relationen darzustellen, innerhalb dem logische Schlussfolgerungen sowie Gewährleistung der Gültigkeit möglich sind. Ontologien beschreiben RDF-Daten formal für andere Systeme und ermöglichen eine Validierung.

Parser
Ist ein bestimmtes Programm zur Zerlegung oder Umwandlung einer beliebigen Eingabe in ein für die Weiterverarbeitung brauchbares Format

Vaadin
Freies Apache Lizenz 2.0 Webanwendungs-Framework für Internetapplikationen
Vaadin steht im Gegensatz zu Browserplugins oder Java-Script basierten Lösungen, da ein Großteil der Programmlogik Serverseitig ausgeführt wird.

Virtuoso
Ist eine DatenbankEngine, die funktionalitäten wie z.B. RDBMS, RDF, XML, web application server, virtual database in einem einzigen System kombiniert.

OWL
Bedeutet Web Ontology Language, ist eine Spezifikation um Ontologien anhand einer formalen Sprache zu erstellen.