Dokumentation des Praktikums XML-Technologie

Technische Universität München

Ralf Baun
Georg Bonczek
Ivan Chimeno
Ferdinand Gruber

27.03.2017

Inhaltsverzeichnis

1. Problemstellung

Im Zuge des Praktikums “XML-Technologie” soll eine Version des Spieles “Mancala” mit den im Praktikum vorgestellten Werkzeugen implementiert werden. Der Schwerpunkt der verwendeten Technologien baut auf der erweiterbaren Auszeichnungssprache XML auf, welche die Grundlage für verwendete Datenbanksysteme, Vektorgrafiken und die Weboberfläche ist.

Das Projekt wird als Gruppe bearbeitet und anschließend im Zuge eines Vortrags vorgestellt. Diese Dokumentation gibt einen Überblick über benutzte Technologien, das Design und die Implementierung des finalen Spiels.

2. Technologie

Im Folgenden wird ein kurzer Überblick über die im Projekt angewandten Technologien gegeben, die gemäß den Vorgaben des Projekts auf XML basieren. Die unterschiedlichen Technologiegruppen lassen sich hierbei dem Frontend und dem Backend zuteilen.

Das Frontend der Anwendung ist mittels dem Standard HTML sowie der Stylesheet-Sprache CSS realisiert. Hierbei wird HTML zur Gruppierung und Ausgabe der einzelnen Seiten des Webauftritts verwendet. Die HTML-Elemente werden durch spezifizierte Klassen und IDs adressierbar gemacht. Dies ermöglicht das Individualisieren einzelner Teile einer Seite durch das Einfügen von personalisierten Informationen, womit das jeweilige HTML-Dokument somit als Schablone dient. Ebenfalls ermöglicht das die effektive Nutzung der Stylesheet-Sprache CSS, welche die Anordnung und das Aussehen der HTML-Elemente regelt. Schlussendlich stellt HTML außerdem über Links und Formularen Möglichkeiten bereit, Nutzereingaben an das Backend weiterzuleiten.

Zur Umsetzung der Logik unter dem Verzicht auf Skriptsprachen wie JavaScript wird die funktionale Sprache XQuery eingesetzt, welche direktes Arbeiten auf XML-Dokumenten ermöglicht. Sie wird von dem Datenbanksystem BaseX ausgeführt, welches ebenfalls als Webserver fungiert. Hierbei wird mittels RESTXQ eine REST-API zur Applikation bereitgestellt.

Um die Elemente des Spielbretts darzustellen und zu animieren, wird der SVG-Standard für Vektorgrafiken verwendet. Diese Grafiken werden dynamisch in die einzelnen HTML-Dokumente eingebettet. Durch die von CSS und SVG gegebenen Funktionen zur Animation der Vektorgrafiken kann auch hier der Einsatz von Skriptsprachen vermieden werden.

3. Architektur

Das Spiel ist in der Konzeptionsphase der Applikationsarchitektur in zwei Hauptbereiche gegliedert worden: In das Frontend, welches den aktuellen Zustand der Anwendung anzeigt und mit dem der Benutzer interagiert, sowie das Backend, das die Logik des Spiels und die Verwaltung der Spieldaten beinhaltet. Insgesamt folgt der Architekturentwurf dem Konzept der Trennung der Funktionalitäten. Diese Trennung ist durch die Unterteilung des Backends in mehrere Module realisiert. Diese kommunizieren über fest definierte Schnittstellen und garantieren einfache Wart- und Erweiterbarkeit der Anwendung. Das Frontend des Spiels besteht aus einer Kombination von HTML in Verbindung mit CSS, welches dessen grafische Darstellung regelt. Das Backend beinhaltet die Module GAME, GFX, WEB und das Datenbanksystem BaseX. Der zugrunde liegende Webserver, welcher die Erreichbarkeit der Anwendung über das Internet ermöglicht, wird zum Bereich des Backends gezählt, aber in dieser Dokumentation nicht explizit behandelt. Das folgende Kapitel gibt einen Überblick über die einzelnen Teilbereiche der Anwendung und erläutert deren Funktion.

3.1. Frontend

Die Weboberfläche, mit welcher der Nutzer interagiert, stellt das Frontend des Spiels dar und ist zuständig für die Ein- und Ausgabe von Spieldaten und Informationen. Dieses ist mittels der Standards HTML und CSS für die Entwicklung von Webseiten realisiert.

Die Website, die dem Benutzer angezeigt wird, wenn dieser die Internetadresse des Spiels aufruft, bietet eine Auswahl mehrerer Funktionen, welche das Spiel unterstützt. Das Hauptmenu umfasst dabei sowohl Verweise auf Anleitung und Bestenliste, als auch die Möglichkeit ein neues Spiel zu beginnen oder ein bereits begonnenes wie-

der aufzunehmen. Die durch HTML und CSS umgesetzten Seiten für die einzelnen Funktionalitäten stellen dem Nutzer dann die angefragte Information zur Vefügung. Hierbei greifen alle Seiten der Anwendung auf ein gemeinsames Stylesheet zu, das deren Darstellung innerhalb des Browser regelt. Die Anzeige des Spielfelds innerhalb des HTML-Dokuments erfolgt durch SVG Vektorgrafiken, welche in die dafür vorgesehenen Teile des HTML-Dokuments eingebettet sind. Zur Animation dieser Spielfeldelemente kommen Funktionalitäten, die CSS bereitstellt, zum Einsatz (wenn z.B. mit der Maus über diese gefahren wird).

3.2. Backend

Der Hauptteil der Anwendung wird dem Backend zugeordnet. Die einzelnen Module, welche zu diesem Bereich zählen, verwalten Spielinformation, Datenbank und die Eingaben des Benutzers. Die Logik der Applikation ist mittels der funktionalen Sprache XQuery realisiert, die in Verbindung mit dem Datenbanksystem BaseX auf dessen RESTXQ-API zugreift, welche die Möglichkeiten der Sprache zum Arbeiten mit Webdokumenten erweitert. Funktionale Teilbereiche der Applikation sind in einzelne Module gegliedert, die über fest definierte Schnittstellen miteinander kommunizieren. Diese Aufteilung der spezialisierten Logik ermöglicht es, Teile der Anwendung leicht zu ändern ohne dabei umfassend die Logik überarbeiten zu müssen.

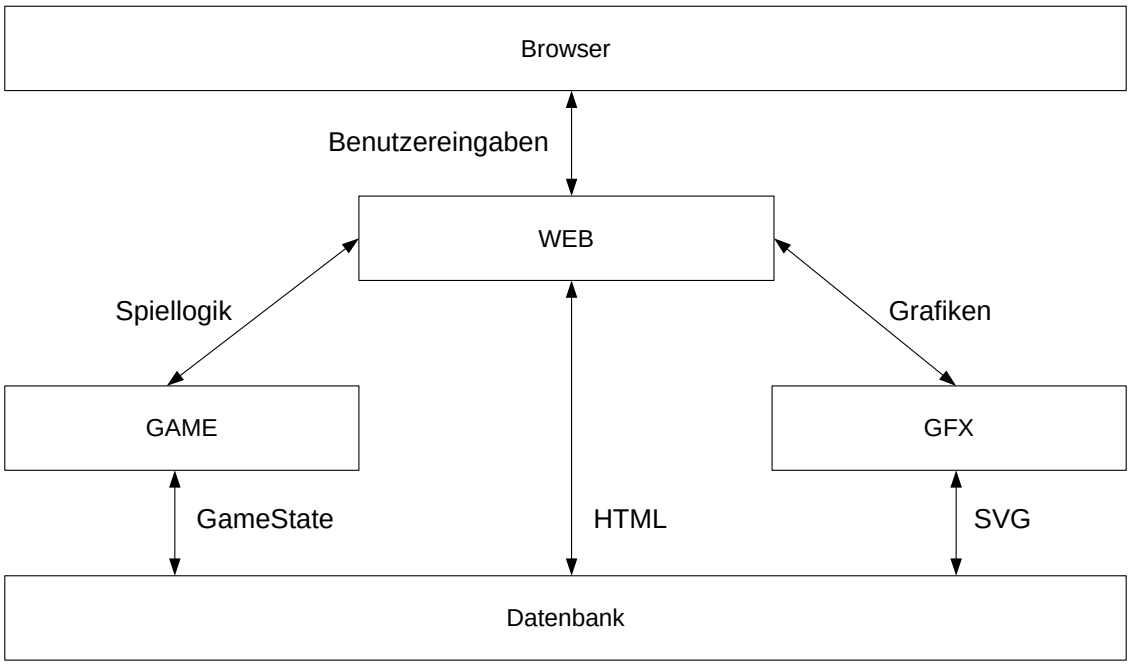

Abbildung 1: Schematische Übersicht der Architektur.

Die Module WEB, GAME und GFX beinhalten die Hauptfunktionalitäten des Spiels und übernehmen dabei fest definierte Aufgabenbereiche. Ein Modul ist hierbei unabhängig von den anderen und folgt so dem Konzept der Trennung von Funktionalitäten. Im Folgenden wird ein kurzer Überblick über die Aufgaben der einzelnen Funktionseinheiten gegeben.

Das Modul WEB ist zentrale Schnittstelle zwischen dem Frontend des Spiels und den anderen Modulen des Backends. Hierbei übernimmt es die Aufgabe, die vom Frontend eingehenden Anweisungen der Benutzer zu bearbeiten und diesen die Spielinformationen zu übermitteln. Um diese Aufgabe zu erfüllen, kommuniziert es – wie in Abbildung 1 zu sehen - mit den anderen Modulen und dem zentralen Datenbanksystem, welches alle für das Spiel relevanten Dateien vorhält. Das Modul verwaltet hierbei insbesondere die einzelnen HTML-Dokumente, die zur Darstellung benötigt werden. Diese werden von ihm mit den Informationen gefüllt und dann dem Spieler bzw. dem Frontend zurückgegeben.

GAME ist zuständig für die Verarbeitung und das Ausführen von Spielzügen sowie dem Anlegen neuer Spielstände. Das Modul erhält die Informationen, die es zum Erfüllen dieser Aufgabe benötigt aus der Datenbank bzw. von WEB. Es verwaltet hierbei das zentrale Speicherdokument „Gamestate“, das alle Informationen, welche die Partie betreffen beinhaltet. Das Modul ist die zentrale Einheit zur Verarbeitung von Spielzügen, d.h. deren Validierung und Ausführung. Auch das Erkennen von Fehlern im Ablauf der Bearbeitung von Anfragen und die Detektion von Betrugsversuchen durch die Benutzer der Anwendung ist Aufgabe des Moduls.

Das Arrangieren der Spielmulden mit den Spielsteinen zum fertigen Element auf Basis der SVG-Assets obliegt dem Modul GFX. Dieses Modul liefert dem Aufrufer, basierend auf einer übergebenen Anzahl von Spielsteinen, Grafikelemente, welche entweder reguläre Spielmulden oder Zielfelder sind.

Das Datenbanksystem ist zentrale Speicher- und Ausführungseinheit der Module und basiert auf BaseX. Hier werden neben den Rohdatein für HTML und SVG auch die zentralen Speicherdokumente („Gamestate“) gespeichert. Die einzelnen Module greifen jeweils auf die Datenbank zu und fragen die vorgehaltenen Dateien an, die sie zum Ausführen ihrer Arbeiten benötigen. Das BaseX-System stellt zusätzlich mit den von ihm bereitgestellten APIs Dienste bereit, welche von den einzelnen Teilbereichen genutzt werden. Die Ausführung der in XQuery umgesetzten Module ist ebenfalls Aufgabe dieses Teils des Backends.

3.3. Schnittstellen

Die Kommunikation der einzelnen Module erfolgt über fest definierte Schnittstellen. Das zentrale Zustandsdokument „Gamestate“ wird als Schnittstellendokument angesehen, da es als Informationsträger zwischen den Modulen ausgetauscht wird und dabei zentrale Aspekte des Datenaustausches übernimmt.

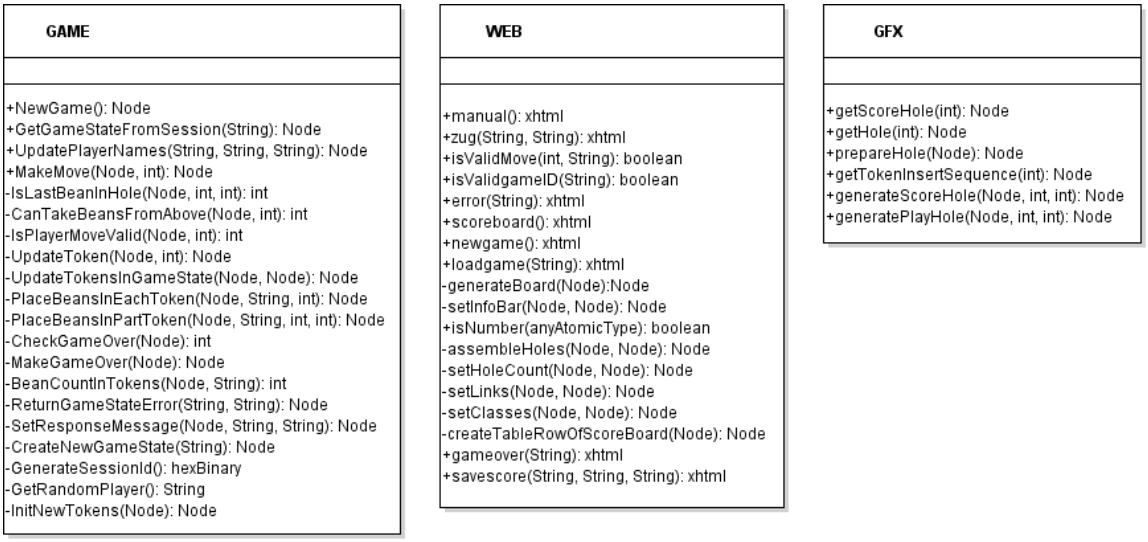

Abbildung 2: Klassendiagramm der einzelnen Module.

Das Modul Game besitzt, wie in Abbildung 2 zu sehen ist, vier öffentliche Methoden, welche vom WEB-Modul zur Bearbeitung der Anfragen des Spielers aufgerufen werden. Hierbei werden die Informationen, die GAME benötigt, der Methode beim Aufruf entweder direkt in Form eines Datentyps oder als Zustandsdatei übergeben, welche die Spielinformationen bündelt und anschließend vom Modul zur Weiterverarbeitung ausgewertet wird. GAME selbst ruft keine weiteren Module des Backends oder Teile des Frontends auf, da es alle benötigten Informationen entweder von Web erhält oder direkt aus der Datenbank bezieht. Das Modul liefert als Rückgabe stets eine Zustandsdatei mit den aktualisierten Spielinformationen zurück.

Das WEB-Modul stellt die zentrale Schnittstelle zwischen dem Frontend und den restlichen Teilen der Applikation dar. Das Modul wird nicht von anderen Teilen des Backends aufgerufen, sondern alle Aufrufe gehen von WEB aus. Um auf die Eingaben des Benutzers reagieren zu können, nutzt das Modul Funktionalitäten der REST-API, welche das BaseX Datenbanksystem bereitstellt. Der Austausch von Daten in Rückrichtung erfolgt über HTML-Dokumente und weitere Funktionalitäten HTMLs. Zur Bearbeitung der Anfragen kommuniziert WEB mit den Modulen GAME und GFX sowie dem zentralen Datenbanksystem.

Das für die Bereitstellung der Vektorgrafiken zuständige GFX-Modul bietet WEB, welches als einziges die Funktionalitäten von GFX nutzt, zwei Methoden an. Hierbei stehen Methoden für die zwei unterschiedlichen Muldentypen des Spielbretts, die anhand einer übergebenen Ganzzahl generiert werden. Dafür kommuniziert das Modul mit dem Datenbanksystem, welchem es die benötigten Ausgangsdateien zur Generierung der SVG-Grafiken entnimmt. Die erzeugten Grafiken werden dann später in das HTML-Dokument des Spielfelds eingebettet.

Neben der Modulkommunikation über Methoden und deren Parameter dient auch der Austausch des GameState-Dokuments der Kommunikation. Dieses kapselt eine ganze Reihe an spielbezogenen Informationen zentral in

einem Dokument, welches dann vom jeweiligen Modul nach Bedarf ausgewertet wird. Spielrelevante Informationen sind beispielsweise der Zustand der einzelnen Mulden des Spielbretts sowie Spielernamen und Sieginformationen der aktuellen Partie. Neben diesen dient die Zustandsdatei auch dem Speichern der Identifikationsnummer und der Kommunikation bei Fehlern im Ablauf der Anwendung. Sollten intern Probleme bei der Verarbeitung auftreten oder Spielzüge nicht durchführbar sein, so werden Rückmeldungen und Fehlercodes im Dokument hinterlegt und hierüber durch die Module zum Nutzer zurückgereicht. Durch die Realisierung als XML-Dokument ist die GameState-Datei leicht erweiterbar, da sich jederzeit neue Elemente zur Informationsspeicherung einfügen lassen. Dies ermöglicht es die Anwendung leicht zu erweitern und anzupassen.

3.4. Ablauf: Neues Spiel

Im Folgenden soll anhand einer Aktion des Benutzers die Funktionsweise der Anwendung veranschaulicht und die Kommunikation zwischen den einzelnen Einheiten des Spiels genauer erläutert werden. Hierzu wird das Erstellen eines neuen Spiels aus dem Hauptmenu der Weboberfläche heraus betrachtet. Das Sequenzdiagramm, welches diesen Ablauf veranschaulicht, ist hierbei zur besseren Übersicht an manchen Stellen vereinfacht, der logische Ablauf allerdings unverändert.

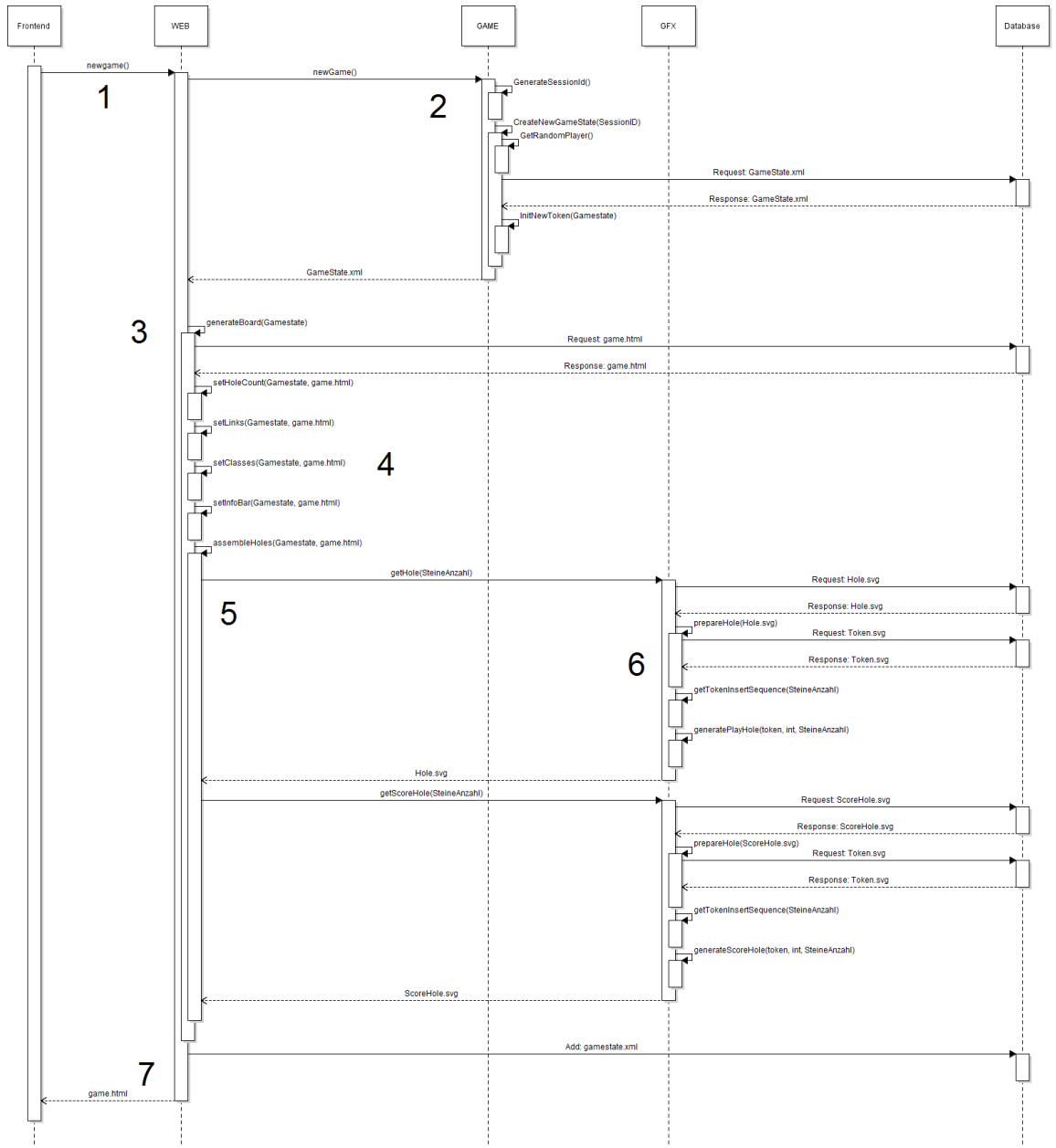

Abbildung 3: Vereinfachtes Sequenzdiagramm des Programmablaufs beim Erstellen eines neuen Spiels.

Wenn der Benutzer die Internetadresse des Spiels aufruft, wird das Hauptmenü der Applikation angezeigt, welches ihm die Option bereitstellt ein neues Spiel zu starten. Beim Anwählen des entsprechenden Buttons wird der Benutzer auf einen anderen Pfad des Webservers weitergeleitet, der im Anschluss das *Ereignis 1* (siehe *Abbildung 3*) auslöst. Diese Auslösung erfolgt über die Annotation der Methode „web:newgame“ an einen relativen Pfad mit Hilfe der REST-API. Die Methode des Moduls WEB initiiert im Anschluss das Erstellen einer neuen Partie für den Benutzer. Dieser Arbeitsschritt teilt sich in zwei Teilbereiche auf, welche zum einen das Erstellen einer neuen Zustandsdatei und zum anderen die Bereitstellung des Spielfelds für die Benutzer sind. Der erste Schritt hierzu

ist der Aufruf der Methode „game:newGame“ des Moduls GAME, welche den dort neu generierten „GameState“ zurückliefert (siehe *Ereignis 2*).

Das Erstellen eines neuen „GameState“ innerhalb des GAME-Moduls ist in fünf Arbeitsschritte unterteilt, an deren Anfang das Generieren einer neuen „SessionID“ steht. Diese Identifikationsnummer besteht aus hexadezimal kodierten Bytes, welche eine Partie einmalig referenziert und der korrekten Zuordnung von Spieleingaben dient. Nach Generierung dieses Kennzeichens wird mittels Aufruf der Methode „game:CreateNewGameState“ innerhalb des Moduls das Erzeugen einer neuen Zustandsdatei eingeleitet. Zuerst wird hierfür mit Hilfe der Methode „game:GetRandomPlayer“ ein zufälliger Spieler bestimmt, welcher später den ersten Zug der Partie ausführen darf. Nach Ermittlung des Spielers wird beim BaseX-Datenbanksystem die Datei „gamestate.xml“ angefragt, welche als leerer Rohling der Zustandsdatei angesehen werden kann. Neben dem ermittelten Beginnenden und der Kennnummer der Partie wird das erhaltene „GameState“-Dokument mit Initialwerten ausgefüllt, welche zum Spielbeginn stets gleich sind. Der letzte Schritt zum Finalisieren der Zustandsdatei wird von der Methode „game:InitNewTokens“ übernommen, die im übergebenen „GameState“ die Ausgangsbelegung der Felder – also die Anzahl an Spielsteinen pro Mulde – definiert. Nach diesem Schritt wird das fertige Dokument an „game:newGame“ zurückgegeben, welches *Ereignis 2* mit der Übermittlung von jenem an die aufrufende Methode „web:newgame“ abschließt.

Nach Erhalt des Zustandsdokuments von Modul GAME beginnt mit *Ereignis 3* die Generierung der Spielumgebung in Form eines HTML-Dokuments. Hierzu ruft die „web:newgame“-Routine die Methode „web:generateBoard“ auf und übergibt dieser als Parameter das von GAME erhaltene Zustandsdokument. Im ersten Schritt fragt die Methode beim Datenbanksystem das Dokument „game.html“ an, welches zusammen mit dem „GameState“ die Basis für die darauffolgenden Arbeitsschritte bildet. *Ereignisgruppe 4* umfasst hierbei die ersten Schritte der Finalisierung der Datei. Zuerst werden mit dem Aufruf von „web:setHoleCount“ die „div“-Elemente im HTML-Dokument aufbereitet, welche für das Gruppieren der Spielmulden zuständig sind. Die Methoden „web:SetLinks“ und „web:setClasses“, welche im Anschluss aufgerufen werden, fügen die Links der einzelnen Mulden und HTML-Klassen in „game.html“ ein. Dies macht die einzelnen Mulden für den Benutzer im späteren Spielverlauf anwählbar. Das Setzen des Infobars („web:setInfoBar“) ist der letzte Schritt vor dem Einfügen der Vektorgrafiken in das Spielfeld. In diesem werden den Benutzern neben Fehlermeldungen auch ereignisbezogene Informationen – beispielsweise der Name des Spielers, welcher am Zug ist – angezeigt.

Ereignis 5 leitet das Finalisieren des HTML-Dokuments ein, da in diesem Schritt das Einfügen der SVG-Elemente in „game.html“ stattfindet. Hierzu wird von „web:generateBoard“ die Methode „web:assembleHoles“ aufgerufen, welche über die einzelnen Spielmulden des Bretts iteriert und in diese schrittweise die Vektorgrafiken einbettet. Hierzu kommuniziert WEB mit dem Modul GFX, das diesem die zwei verschiedenen Grafiktypen der Arten von Mulden bereitstellt. Der Ablauf des Generierens und Einfügens ist hierbei jeweils gleich und wird exemplarisch anhand einer regulären Feldmulde aufgezeigt. Zuerst liest „web:assembleHoles“ aus dem ihr übergebenen „GameState“ die Anzahl der Spielsteine aus, welche die Grafik innerhalb der Mulde beinhalten soll. Diese Information wird anschließend der Methode „gfx:getHole“ des WEB-Moduls beim Aufruf übermittelt, die dann mit dem Erzeugen der Grafik beginnt.

Dieser Prozess (siehe *Ereignisgruppe 6*) beginnt mit dem Anfordern der Datei „Hole.svg“, welche die Basis für die Grafik der Mulde bildet, von der Datenbank. Mittels des Aufrufs von „gfx:prepareHole“ wird in jene Datei im Anschluss ein weiteres SVG-Element eingebettet, das zuvor von der Datenbank angefordert worden ist. Innerhalb des Sequenzdiagramms ist der Arbeitsschritt des Lochfüllens vereinfacht dargestellt. Für jeden Spielstein, der sich später in der Mulde befinden soll, wird auf die anfänglich eingebettete Grafik „Token.svg“ referenziert. Diese Referenzierung wird mit Hilfe des Aufrufs der Methode „gfx:generatePlayHole“ - innerhalb der für den jeweiligen Typ der Mulde festgelegten Grenzen - neu ausgerichtet bzw. positioniert. Nach dem Fertigstellen einer Mulde wird jene dem WEB-Modul zurückgegeben, welches diese in die „game.html“ einbettet. Die Anzeige des Spielbretts basiert somit nur auf SVG-Elementen und verzichtet auf Funktionalitäten, welche HTML und CSS – mit Ausnahme von Animationen- zur Darstellung bereitstellen.

Nach dem Abschluss der Brettgenerierung bei *Ereignis 7* wird dieses über die Kette an Aufrufern zurückgereicht und schließlich wieder der Methode „web:newgame“ übergeben, die das HTML-Dokument schließlich an das Frontend zurückreicht. Vor diesem Schritt wird noch das „GameState“-Dokument in das BaseX-Datenbanksystem eingetragen. Die von GAME generierte Kennzahl ist hierbei ein Teil des Dateinamens und dient somit der Identifizierung der verschiedenen Zustandsdateien noch bevor deren Auswertung beginnt.

Im Browser wird nun das in den vorherigen Schritten erzeugte Spielfeld angezeigt und die Spieler können mit der Partie beginnen.

4. Implementierung

Die Trennung von Funktionalitäten mittels der Konzeption spezialisierter Module hat es ermöglicht, verschiedene Teile der Anwendung parallel zu entwickeln. Durch die Festlegung der obig beschriebenen Schnittstellen ist es möglich gewesen, dass die einzelnen Funktionseinheiten durch verschiedene Gruppenmitglieder umgesetzt werden konnten. Auch ist die Anwendung Schrittweise entwickelt worden und somit - nach der Implementierung der Basis - an Funktionalität über den Zeitraum des Praktikums gewachsen. Dies ist möglich gewesen, da einzelnen Module erweitert und modifiziert werden konnten, ohne dabei Einfluss auf die Umsetzung der restlichen Teile zu nehmen.

4.1. Designentscheidungen

Vor der konkreten Umsetzung des Architekturentwurfs mussten Entscheidungen in Bezug auf die zu verwendenden Technologien und unterschiedliche Lösungsstrategien getroffen werden. Im Folgenden soll ein Überblick darüber gegeben werden, nach welchen Maßstäben und Überlegungen verschiedene Technologien einbezogen oder herausgelassen worden sind.

Da die Anwendung auch Eingaben von Seiten des Benutzers in Form von Spielernamen unterstützen soll, hat es die Wahl zwischen zwei verschiedenen Technologien zur Umsetzung dieser Funktion gegeben. Sowohl „XForms“ als auch HTML-Forms verfügen über die Funktionalität für die Eingabe von Zeichenketten. Die "XForms"-Technologie ist ein mächtiges Werkzeug zur Generierung interaktiver Web-basierter Eingabeformulare, welche in Form von XML-Dokumenten beschrieben werden. Diese erlaubt darüber hinaus den Verzicht auf Javascript zur Validierung der Eingaben und stellt bei Nutzung somit eine Einsparung an Ressourcen dar. Die zweite ist ein Teil von HTML und findet Verwendung in vielen modernen Webanwendungen. Die Komplexität bei Nutzung von XForms ist größer als die, welche mit Nutzung des HTML-Pendants einhergeht. Diese größere Komplexität steht allerdings nicht im Verhältnis zur Anforderung des Einlesens einer einzelnen Zeichenkette, weshalb die Nutzung von XForms als Technologie für die Anwendung verworfen worden ist.

Die zur Visualisierung eingesetzte SVG-Technologie ermöglicht vielfältige Lösungsansätze zur Generierung einer einzelnen Grafik. Das Augenmerk der Projektgruppe ist auf einer möglichst effizienten Nutzung des Standards gelegen und sollte zugleich die Komplexität der einzelnen Grafiken reduzieren. Durch Nutzung der „def“ und „use“ Funktionalität des SVG-Standards ist es möglich, mehrere gleiche Grafiken in ein SVG-Dokument einzubinden ohne dabei Informationen redundant zu speichern. Erste Versuche zur Generierung der Mulden inklusive der Spielsteine hatten zu großen und komplexen Dateien geführt, welche das Arbeiten mit den Vektorgrafiken erschwerte. Durch die Speicherung eines Referenztokens im „def“-Bereich der übergeordneten SVG ist es möglich, diese mittels „use“ mehrmals zu verwenden und die neuen Instanzen dabei neu zu positionieren. Die Verwendung dieser Technologie hat die Größe einzelner Vektorgrafiken reduziert und somit die Performanz und Handhabbarkeit der Anwendung erhöht.

Zur Animierung des Spielfelds bei Eingaben des Benutzers sind der Gruppe SVG eigene Technologien und Funktionalitäten aus JavaScript und CSS zur Auswahl gestanden. Da der Verzicht auf JavaScript zentrales Element der Überlegungen gewesen ist, fiel die Wahl auf die Animationsmöglichkeiten der SVG-Technologie in Zusammenspiel mit CSS. Einzig der Verbund aus Funktionalitäten von CSS und dem SVG-Standard zur Einbindung sog. „Hover“-Animationen findet bei der Darstellung des Spielbretts Verwendung. Auf Skriptsprachen konnte somit verzichtet werden.

Zur Verwaltung der Spielinstanzen hat es verschiedene Lösungsansätze gegeben, welche jeweils Vor- und Nachteile sowohl im Nutzen als auch der Implementierung bieten. Die anfänglich im Betracht gezogene Umsetzung mittels der alleinigen Identifikation einer Spielinstanz durch eine URL und somit deren Laden ist aufgrund der Notwendigkeit zur Speicherung der Internetadresse verworfen worden. Diese Herangehensweise ermöglicht es zwar, keinerlei Informationen zur Spielinstanz beim Nutzer speichern zu müssen, hat allerdings durch Bedenken im Bereich der Sicherheit nicht überzeugen können. Der Ansatz "SessionIDs" des Spiels unter Verwendung eines Passworts zu schützen und das Laden einer bereits begonnenen Partie durch Eingabe dieser Identifikationsnummer und dem korrekten Passwort zu realisieren, konnte zum einen aufgrund der aufwendigeren Technik zur Speicherung und Überprüfung nicht überzeugen und erschien der Gruppe zum anderen für die Verwendung als zu komplex. Als guter Kompromiss zwischen Sicherheit und Handhabbarkeit hat am Ende das Nutzen von Cookies zur Speicherung der „SessionID“ auf Benutzerseite gegolten. Dieses beim Erstellen eines neuen Spiels gesetzte Cookie identifiziert den Benutzer als Besitzer einer Partie und wird beim Laden eines Spiels automatisch aus dem

Browser des Nutzers ausgelesen. Dies erhöht auf der einen Seite den Komfort, andererseits ermöglicht es dem Nutzer die Referenz auf ein Spiel (das Cookie) sowohl selbständig zu löschen, als auch das Verwenden eines anderen Browsers zur Fortzsetzung des Spiels.

4.2. Sicherheit

Im Laufe der Implementierung ist das Einbeziehen des Aspekts der Sicherheit ein zentraler Bestandteil der Arbeiten am Projekt gewesen. Im Kontext des Projekts steht Sicherheit hierbei für den Schutz des Spielablaufs vor Betrugsversuchen durch den Benutzer.

Das Verwenden nicht vorhandener Spielinstanzen durch die Manipulation der URL des Spiels führt zu einer Rückmledung an den Benutzer, welche ihm mitteilt, dass unter dieser Identifikationsnumer kein Spiel gefunden werden kann. Diese Überprüfung der Validität von Spielinstanzen stellt einen einfachen Schutz vor fehlerhafter Verwendung des Spiels und daraus resultierenden Problemen dar.

Da die Ausführung eines Zuges mit Hilfe von URLs umgesetzt ist, wäre es prinzipiell möglich durch Abänderung der Adresse Spielzüge auf Mulden des Spielbretts auszuführen, für die der Spieler keine Berechtigung hat. Das GAME-Modul überprüft vor der Ausführung eines Spielzuges, ob dieser im aktuellen Zustand der Partie valide sein kann und reagiert dann entsprechend des Ergebnisses. Solte ein Zug keine Gültigkeit nach den Regeln des Spiels haben, dann wird dies dem Benutzer mit Hilfe des Inforbars mitgeteilt und das Spielfeld unverändert zurückgegeben. Dieser Mechanismus der Überprüfung ist eine elegante und wirksame Schutzmaßnahme gegen Eingriffe in den regulären Spielablauf.

5. Evaluation des Projekts

Im folgenden Kapitel wird der Verlauf des Praktikums und die Implementierung des Projekts evaluiert. Im Anschluss wird ein Fazit zum Verlauf des Praktikums gezogen.

5.1. Evaluierung der Implementierung

Die Realisierung von Multiplayer-Spielen mittels Technologien aus dem XML-Universum ist möglich, aber an vielen Stellen nicht optimal umzusetzen. Im Laufe der Implementierung ist die Gruppe auf einige Hindernisse gestoßen, deren Umsetzung durch den Verzicht auf Technologien - wie den Einsatz von Skriptsprachen - komplexer in der Gestaltung sind. Wenn zwei Nutzer auf verschiedenen Browsern eine Partie gegeneinander spielen, dann aktualisiert sich der Inhalt der Spielseite nicht automatisch bei einem Zug des Gegenüber. Dieses automatische Nachladen des aktuellen Zustands des Spielfelds kann durch den Einsatz von JavaScript umgesetzt werden und würde somit den Komfort der Anwendung für die Nutzer erhöhen. Auch das Generieren und Arrangieren der Grafiken durch das GFX-Modul innerhalb des Backends der Anwendung ist eine dem Einsatz von XML-Technologien geschuldete Lösung. Mittels einer Skriptsprache könnte diese Arbeit auf Seiten des Nutzers verrichtet werden und würde die Anwendung somit auch entlasten. Der Einsatz eines Cookies zum Laden des Spiels bringt den Nachteil, dass durch Manipulation dieses Datensatzes die Möglichkeit besteht, bereits begonnene Partien anderer Spieler zu laden, wenn denn deren Kennnummer bekannt ist. Dieser Umstand kann durch die Speicherung von mehr nutzerbezogenen Daten auf Seiten der Anwendung gelöst werden, die den Besitzer zusätzlich identifizieren würden.

Die Umsetzung des Architekturentwurfs in der Phase der Implementierung mittels XQuery ist von der Gruppe gut durchgeführt worden. Die Qualität des Programmcodes ist durch das permanente Refaktorisieren stetig gestiegen und hat zum Abschluss der Arbeiten am Projekt zu einem hohen Maß an Güte geführt. Hierbei sind aktiv Konzepte und Möglichkeiten von XQuery in der Implementierung eingesetzt worden, um die Möglichkeiten der zur Verwendung kommenden Techniken bestmöglich für das Projekt auszunutzen.

5.2. Evaluierung des Projektverlaufs

Im Folgenden soll ein kurzer Überblick über die Arbeit der Gruppe am Projekt gegeben und zugleich herausgearbeitet werden, auf welche Probleme bzw. Schwierigkeiten im Verlauf des Projekts gestoßen worden ist.

Beim gemeinsamen Entwurf der Architektur des Spiels ist die Anwendung in mehrere Teile aufgegliedert worden, welche dann einzelnen Mitgliedern der Gruppe als Hauptverantwortliche für diesen Bereich übergeben worden sind. Diese Aufteilung der Arbeiten ist durch die Trennung der Funktionalitäten mittels der Modularisierung des

Spiels ermöglicht worden und hat es der Gruppe gestattet, das Projekt an verschiedenen Stellen parallel zu entwickeln. Trotz einer von Anfang an weitreichenden Konzeption der Anwendung ist es immer wieder vorgekommen, dass nachträglich kleine Änderungen am ursprünglichen Entwurf gemacht worden sind. Dieser Umstand ist zum einen nicht beachteten Fällen während der Konzeption geschuldet und zum anderen den neuen Technologien, welche die Gruppe für die Realisierung des Projekts verwendet hat.

Neben den generellen Problemen bei der Einarbeitung in neue Technologien ist der funktionale Aspekt von XQuery am Anfang ein Problem für die Gruppe gewesen. Das funktionale Programmieren fordert ein Umdenken in Bezug auf die Herangehensweise beim Lösen von Problemen im Zuge der Anwendungsentwicklung. Die Schwierigkeiten beim Verwenden dieses Programmierparadigmas im Zusammenhang mit XML-Technologien und der Anforderung, dass das Spiel eine Webapplikation sein muss, hat die größte Hürde beim Beginn der Arbeiten am Projekt dargestellt. Die während des Praktikums gereichten Unterlagen und Arbeitsblätter sind allerdings eine gute Hilfestellung gewesen, da sie die neuen Technologien grundlegend eingeleitet haben und somit das Nehmen erster Hürden erleichtert haben.

Durch die Vernetzung der Gruppe und die regelmäßigen Treffen ist es möglich gewesen, offene Fragen einzelner Mitglieder, die beim Bearbeiten der Aufgaben aufgekommen sind, schnell zu beantworten und somit die Arbeit am Projekt aufrechtzuerhalten. Dieser Einsatz hat sich bereits am Anfang des Praktikums gezeigt, als die Gruppe beschloss, den Kommilitonen das erste Arbeitsblatt vorzustellen.

Die Arbeit der Gruppe am Projekt kann somit trotz kleinerer Probleme und Schwierigkeiten während der Arbeiten als zielführend und gut angesehen werden.

5.3. Fazit

Das Praktikum über XML-Technologie ist eine Lehrveranstaltung, welche Studenten Technologien vorstellt, die im regulären Studienbetrieb weniger Beachtung finden. Die XML-Welt bietet vielfältige Technologien, welche sich in diversen Bereichen der Informatik einsetzen lassen und eine echte Alternative zu anderen Lösungen darstellen. Der Aufbau und die Organisation des Praktikums hat uns als Gruppe sehr gut gefallen und wir haben dadurch mit Elan den Einstieg in die XML-Welt gemeistert. Die anfängliche Ratlosigkeit, welche die verschiedenen neuen Technologien anfänglich verursacht haben, konnten wir dank der Arbeitsmaterialien gut überwinden.

Das Projekt, welches wir im Rahmen des Praktikums realisiert haben, ist ein Spiel für mehrere Personen. Während der Bearbeitung des Spiels ist schnell ersichtlich geworden, dass die Techniken aus dem XML-Universum ursprünglich nicht für solche Dinge konzipiert worden sind. Die Arbeit kann also eher als „Proof of Concept“ gesehen werden als ein in der Realität eingesetzter Lösungsansatz. Aufgrund dieses Umstands werden der Quellcode und die zum Spiel benötigten Dateien am Ende der Lehrveranstaltung veröffentlicht, um Interessenten aus aller Welt die Möglichkeiten von XML-Technologie zu zeigen. Trotzdem denken wir, dass es in zukünftigen Praktika sinnvoll sein könnte, Projekte umzusetzen, die einen größeren Nutzwert haben als ein Mancala-Spiel und so von den Teilnehmern auch persönlich nach dem Praktikum genutzt werden können.