Leichtgewichtigkeit als Prinzip – Gestaltung der Webanwendung myPIM durch UCD, FDD und Xtext

Torsten Krohn[1], Martin Christof Kindsmüller[2]

[1]itemis AG
Schauenburger Str. 116, 24118 Kiel
torsten.krohn@itemis.de,
[2]Universität zu Lübeck, Institut für Multimedia and Interactive Systems
Ratzeburger Allee 160, 23538 Lübeck
mck@imis.uni-luebeck.de

Abstract: Im Zuge des sich abzeichnenden digitalen Lebensstils, in dem Nutzer zunehmend zu partizipativen (Mit-)Gestaltern werden, wird die für den/die Einzelne(n) relevante Informationsmenge auch in Zukunft weiter ansteigen. Angesichts der Unübersichtlichkeit des Informationsangebots im WWW stellt sich zunehmend die Frage, wie nützliche Information gefunden und wiedergefunden werden kann. In diesem Aufsatz wird die Konzeption eines *Social Bookmarking Services* vorgestellt, der insbesondere die Anforderungen von Nutzern aus dem Forschungs- und Lehr-/Lernkontext unterstützt. In einem kombinierten Vorgehen auf Basis verschränkter UCD- und FDD-Prozesse, werden Kernsystem und Module flexibel gegenüber Änderungen realisiert. Der Einsatz des leichtgewichtigen Frameworks Xtext für domänenspezifische Sprachen ist dabei von großer Bedeutung.

1 Einleitung

Das Internet und eine seiner Hauptanwendungen, das World Wide Web (WWW), wächst in den letzten Jahren enorm. Laut einer von ARD und ZDF [AZ07] in Auftrag gegebenen Studie nutzten 2007 etwa 62 % aller Haushalte einen PC mit Internet-Anschluss, während es zehn Jahre früher nur 6,5 % waren. In den Unternehmen bei Bürokräften und in der Lehre wurde eine Versorgung von über 90 % erreicht [SB07]. Ende des Jahres 2006 wurde das "Web 2.0" vom Time-Magazin ausgezeichnet, indem "You" zur "Person of the Year" gekürt wurde [SG06]. Dadurch wurde der sich abzeichnende „digitale Lebensstil" in den Fokus gerückt, denn gemeint ist jeder Nutzer, der sich über Weblogs, Wikis oder anderweitig im WWW einbringt und zu dessen Entwicklung beiträgt. Somit wird der Nutzer vom passiven Verbraucher zum partizipativen (Mit-)Gestalter.

Infolgedessen steigt die Datenmenge massiv an. Etwa 16 Milliarden Webseiten [Wo09] sind beim beliebtesten Suchanbieter, Google, verzeichnet, dennoch erklärt CEO Eric Schmidt: von den geschätzten fünf Exabyte Daten auf der Welt seien bislang nur rund 170 Terabytes indexiert worden [Ku05]. Es vergingen noch mindestens 300 Jahre, bis alle Informationen indexiert und damit für Suchanfragen zugänglich sind.

2 User-Centered-Design, Feature- und Model-Driven-Development

In jüngster Vergangenheit gab es verschiedene Lösungsansätze zur Bewältigung dieser Datenmengen. Allerdings wurden diese oft in der ein oder anderen Weise am Nutzer vorbei entwickelt. Dies legt die Vermutung nahe, dass die potenziellen Nutzer der zukünftigen Anwendung das Design des Werkzeugs stärker mitgestalten sollten als dies bislang geschah. Diese nutzerzentrierte Gestaltung (vgl. User-Centered-Design [Ho05]) zielt darauf ab, interaktive Produkte so zu gestalten, dass sie über eine hohe Gebrauchstauglichkeit (Usability) verfügen. Dabei werden zukünftige Nutzer dieses Produkts mit ihren Aufgaben, Zielen und Eigenschaften in den Mittelpunkt des Entwicklungsprozesses gestellt. Dies wird im vorliegenden Fall primär durch wiederholte Benutzerbefragung und kurze Usability-Tests im Zuge einer formativen Evaluation aufgebaut.

Das gesamte Vorgehen im Rahmen dieser Tool-Entwicklung kombiniert den iterativen Entwicklungsansatz mit dem FDD-Ansatz (Feature-Driven-Development [CLL99]). Dabei werden zunächst Use-Cases aufgestellt und aus diesen potenzielle Features abgeleitet. Anschließend werden diese – zusammen mit der Zielgruppe – priorisiert. Nach der Implementierung eines Kernsystem-Prototypen mit den zentralen Features wird über die weiteren Module (Feature Sets) iteriert. Test, Feedback und Implementierung wechseln sich ab, so dass kontinuierlich ein besseres, mächtiger werdendes Werkzeug entsteht.

2.1 User Centered Design

Zur Bestimmung der Use-Cases und für die Ableitung von zu unterstützenden Funktionen wurden zunächst Nutzerbefragungen in Form von semistandardisierten Interviews durchgeführt. Charakteristisch für diese Form der Befragung ist ein Leitfaden, der – als Kompromiss aus standardisiertem Interview und explorativer Studie – Interviewinhalte vorgibt, dabei aber Abweichungen vom Plan nicht unmöglich macht. Die Interviews sollten Aufschluss über den alltäglichen Umgang mit Bookmarks geben. Dafür werden folgende Themenbereiche behandelt: Eröffnungsfragen zu Anzahl und Speichermedium der Bookmarks, Einsatz von Bookmarks im Lehr-/Lernkontext, Workflow, Social-Bookmarking, Gruppenarbeit, sonstige Arbeiten, Studium, Bewertung potenzieller Features, Erfahrungen mit anderen Tools. Nach Auswertung der Interviewdaten konnten elf prototypische Use-Cases[1] wie z. B. “Bookmark unter Zeitdruck merken”, “Schneller Zugriff auf häufig benutzte Bookmarks” identifiziert werden.

2.2 Feature-Driven-Development als Methode zur nutzerzentrierten Gestaltung

Feature-Driven-Development ist ein iterativer und inkrementeller agiler Entwicklungsprozess für Software bei dem eine Reihe *Best Practices*, die in der Industrie Verwendung

[1] Hier und im Folgenden verstehen wir Use-Case im Sinne von Lauesen [La04] als Interaktionen zwischen Akteuren und dem betrachtetem System, die stattfinden, um ein bestimmtes fachliches Ziel zu erreichen. Sie beschreiben einen Ablauf, wobei aufgezeigt wird, welche Aufgaben vom Computer und welche vom Benutzer übernommen werden.

233

finden, geeignet zusammenführt werden. Die von uns favorisierte Variante basiert auf der Bewertung der Funktionalität aus Sicht der Benutzer, die zuvor bereits als Features definiert und konkret für das vorliegende Problem erstellt wurden. Im Mittelpunkt von FDD stehen Feature und Feature-Set. Zentral ist das agile, hochtaktige Programmieren von Softwarekomponenten, deren Qualität frühzeitig und schrittweise durch Feedback gesichert werden kann. FDD gliedert sich in fünf Prozesse: “Develop Overall Model”, “Build Feature List”, “Plan By Feature”, “Design By Feature”, “Build By Feature”. Mit den ersten drei sequentiellen Prozessen wird ein Gesamtmodell entworfen, die beiden letzten Prozesse iterieren pro Feature.

Im vorliegenden Fall wurden aus den im UCD-Prozess generierten Use-Cases 30 Features abgeleitet. Auf Grundlage umfangreicher Recherchen konnten zahlreiche ergänzende Features identifiziert werden. Da auch zu den Letzteren die Meinung der zukünftigen Nutzer eingeholt werden sollte, wurden diese Nutzer in einem Workshop aufgefordert, die gesamte Feature-Liste nach Relevanz zu bewerten. Anschließend wurden die Features in so genannte Feature-Sets gruppiert. Ein Feature-Set ist dabei definiert als diejenigen Features, die sowohl vom Anwendungskontext her als auch in Bezug auf die Modulabhängigkeiten in direkten Zusammenhang stehen. Der Umfang eines Feature-Sets wird dabei so gewählt, dass dieses innerhalb von zwei Wochen implementiert werden kann.

2.3 Model-Driven-Development als Ergänzung zum FDD

Studien haben gezeigt, dass bis zu 90 % der Kosten, die bei der Entwicklung eines Softwaresystems entstehen, während der Weiterentwicklungs- und Wartungsphase verursacht werden [Ko03]. Gelingt die Anpassung bei jungen Systemen noch recht gut, so werden Tools im Laufe der Zeit immer schwieriger wartbar. Die Hauptursache dieses Problems ist die hohe Komplexität, die durch die fortwährende Implementierung zahlreicher Funktionen von nicht minder zahlreichen Entwicklern einhergeht. Dabei ist die Richtlinie, Softwaresysteme möglichst redundanzfrei und verständlich zu implementieren, schwer einzuhalten.

Es liegt auf der Hand, der Komplexität mit einer geeigneten Abstraktion zu begegnen. Tatsächlich stellt das Finden und Definieren von geeigneten Abstraktionen einen essentiellen Bestandteil der Softwareentwicklung dar. Der klassische Ansatz dies zu realisieren, ist die Erstellung von Bibliotheksfunktionen und Frameworks, doch dies ist aufwendig und verlagert die Komplexität nur hin zu den verwendeten Programmiersprachen statt sie außerhalb des Entwicklungsprozesses zu kapseln. Die Entfernung der Programmierung von der Plattform hin zu den zu lösenden Problemen führte zu den so genannten domänenspezifischen Sprachen (Domain-Specific Languages; DSLs), die auf eine bestimmte Anwendungsdomäne zugeschnitten sind. Somit können domänenspezifische Sachverhalte viel präziser und leichtgewichtig formuliert werden.

DSLs sind eine Ausgestaltungsform der so genannten modellgetriebenen oder modellbasierten Softwareentwicklung (Model-Driven-Software-Development; MDSD), bei der formale Modelle definiert und auf die jeweilige Zielplattform abgebildet werden. Dieses Vorgehen erhöht die Entwicklungsgeschwindigkeit, sorgt für klar verständliche Domä-

234

nenkonzepte, verbessert die Testbarkeit und steigert insgesamt die Effizienz der Softwareentwicklung. Dieser Prozess der Modellierung der Domäne ist zunächst mit einem höheren Programmieraufwand verbunden, zahlt sich aber im weiteren Verlauf der Entwicklung aus. Dies läuft konform mit der Einteilung in ein Kernsystem und der Features, die später an das System andocken. Im Gegensatz zu nicht-modellgetriebenen Ansätzen entsteht in der "Andockphase" so gut wie kein Overhead.

3 Kernsystem

Nach Darstellung der Vorgehensmodelle und der Ansätze zur agilen und effizienten Implementierung des Tools, folgt nun die Vorstellung des Kernsystems eingeteilt in die zugrundeliegenden Featuresets.

3.1 Benutzungsschnittstelle und Navigation

Die Interviews (vgl. Abschnitt 2.1) haben gezeigt, dass die meisten Benutzer Browserbookmarks benutzen. Notizen erfolgen oft auf losen Zetteln und URLs (zu Kollegen/Kommilitonen, aber oft auch an sich selbst vor einem Standortwechsel) über einen E-Mail-Client ausgetauscht.

Die Anwendung myPIM soll die Möglichkeit schaffen, diese Vorgehensweisen zusammengelegt auszuführen. Die Benutzungsschnittstelle stellt dabei den hauptsächlichen Handlungsrahmen dar und sollte deshalb als erstes nach der Realisierung des Anwendungsservers umgesetzt werden. Erbringt das Feedback der Benutzer bei den Iterationen des FDD weitere Featurewünsche bzw. -prioritäten, dann kann das Front-End modifiziert und erweitert werden.

Das Werkzeug verbindet die „modernen" computerbasierten Lösungen für Personal-Knowledge-Management [Ki06], nämlich einen graphischen PIM (Personal-Information-Manager), einen *Social Bookmarking Service* sowie die Kommentare in Blogs, mit denen sich die Benutzer untereinander austauschen können. Hierbei sollen wichtige Navigationspunkte, wenn möglich, mit Metaphern belegt werden, denn diese liefern den Benutzern ein Abbild einer schon bekannten Welt über den Computer [CMK88] und helfen beim Umstieg von z. B. Papiernotizen zu elektronischen Notizen. Wird die URL der Homepage (vgl. Abb. 1) aufgerufen, sucht die Anwendung zunächst nach bereits bestehenden Logindaten. Bestehen diese nicht, startet ein Registrierungsprozess in drei Schritten. Die besondere Leichtgewichtigkeit der Oberfläche zeigt sich dabei im starken Einsatz von asynchroner Kommunikation durch AJAX. Ein Fehler in der Anzeige wird nicht wie üblich nach Absenden des Formulars angezeigt, sondern schon während der Eingabe. Die Mensch-Computer-Schnittstelle wird dadurch so schnell wie ein Offline-Client. Nach dem Anmeldeprozess kann in einer Art Tour mit dem Sichten der Hauptmodule begonnen werden (Dashboard, Bookmarks, Community ansehen).

Das Dashboard (Instrumententafel) gibt einen Überblick über die letzten Aktivitäten (sowohl die der Freunde in der Community als auch die eigenen), die Nachrichten und

235

Aufgaben, die zu bearbeiten sind, als auch die eigenen. „Bookmarks“ ist der Bereich zur Suche (sowohl in den eigenen Bookmarks als auch in denen der anderen Benutzer) und Pflege der Bookmarks. Auf der Community-Seite sind die typischen Bearbeitungsfunktionen für Freunde und Aufgaben bereitgestellt, d. h. Friend-Request, Auflösen des Freundschaftsattributs sowie die Auflistung der geführten Gespräche über Bookmarks.

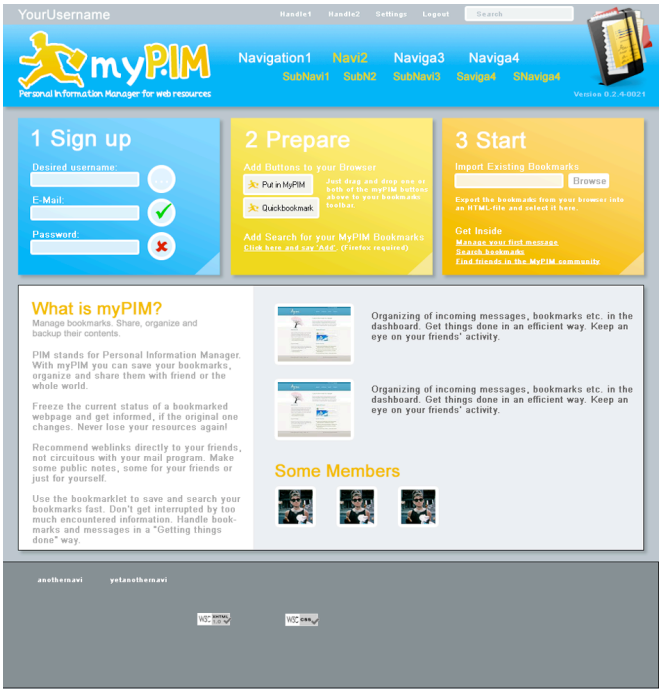

Abbildung 1: Registrierungsseite

3.2 Bookmarking und Workflow

Angesichts der Erkenntnisse bezüglich Management, Workflow und multiplen Kontexten stellt insbesondere das Dashboard als zentraler Information-Hub hohe ergonomische Gestaltungsanforderungen. Es existiert ein gemeinsamer Kontext für die Verwaltung der eingehenden Nachrichten über Bookmarks, d. h. der vom System generierten Aufgaben („bitte Tags vergeben für...“) und der Teilprojekte des Benutzers (d. h. nutzergenerierter Aufgaben). Hierbei wird die Handhabung an das populäre Selbstmanagementverfahren „Getting Thinks Done“ [Al05] angelehnt.

3.3 Suchfunktion und Community

In dieser Entwicklungsphase soll das elementare Auflisten der Suchergebnisse mit dem Laden einer neuen Seite realisiert werden. Auch die Anfragen von einem Searchlet des Browsers werden bearbeitet und direkt angezeigt.

236

Ab diesem Zeitpunkt ist die Benutzung des Bookmarkmanagers möglich. Ein kleiner Kreis von Testnutzern erhält Zugriff und kann nach der Benutzung ein erstes Feedback über den aktuellen Stand geben. Im weiteren Verlauf des FDD gehen die Wünsche und Meinungen der Gruppe in die Priorisierung der Features bzw. Featuresets ein. Durch Integration eines Blogs kann der Entwickler zeitnah über Neuerungen informieren, die dann von den Nutzern ebendort kommentiert werden können. Gleichzeitig wird so der Entwicklungsverlauf langfristig nachvollziehbar.

Nach [KMM09] ist eine Online-Community eine reale (nicht virtuelle), freiwillig agierende Gruppe von Leuten, die aktiv mit Hilfe eines bestimmten Computersystems partizipieren. Um die Etablierung einer Online-Community zu unterstützen muss myPIM als eine Art Überträger zur Vermittlung der sozialen Interaktion dienen. Die Möglichkeit der Gruppenbildung über das Feature des Friend-Requests (Freund-Anfrage) ist deshalb essentiell. Nach dem Senden eines solchen Requests mit einer Kurznachricht erhält der angefragte Nutzer eine Nachricht und kann entsprechend bestätigen oder ablehnen.

3.4 Domänenmodell

Um die gesamten Systemdaten zu persistieren und zu verwalten, wird eine Datenbank eingesetzt. Die Realisierung dieser Domäne, also der elektronischen Variante des abgrenzbaren Problemfelds im Alltag, geht sowohl einher mit der genannten Komplexität als auch mit gewissen Redundanzen: Das Datenbankschema beinhaltet n Tabellen, die Entitäten und Relationen wie “Users” oder “Bookmarks” halten. Für die Persistenzschicht des Systems muss ein Mapping definiert werden (hier Hibernate), das diesen Tabellen entspricht. Zum komfortablen Umgang mit den Domänenobjekten gibt es Klassen für jede Entität und Relation, so dass wiederum die gleiche Information anders dargestellt werden muss. Eine weitere Schicht (die Data-Access-Objects) ist auch größtenteils davon ableitbar.

Um diese Redundanzen nicht pflegen zu müssen, wird das Open-Source Werkzeug Xtext [Xt09] eingesetzt, das die Definition beliebiger Programmiersprachen ermöglicht. Auf Basis dieser Definition kann Xtext eine komplette Programmierumgebung generieren. Mit der generierten Sprache bekommt man ohne weiteres Zutun Werkzeuge wie z. B. einen Editor an die Hand, die optimal auf die vorliegende Domäne zugeschnitten sind. Ist dies geschehen, „modellieren“ Entwickler im eigentlichen Wortsinn möglichst große Teile des vorliegenden Systems anstatt sie auf niedriger Ebene “von Hand” zu kodieren.

In Abbildung 2 lässt sich die abstrakte Domäne myPIM anhand des Modells leicht erkennen: das Schlüsselwort *entity* kennzeichnet die Deklaration einer Entität. Danach folgt der Name und eine Kapselung durch geschwungene Klammern. Innerhalb der Klammern werden die Attribute (*id*, *name*, ...) und ihre Datentypen (*Int*, *String*, ...) bzw. Referenzen (1-zu-n, n-zu-n) gesetzt. Parameter wie *length* verändern die Standardeinstellungen des Datentyps.

237

```
entity Group {
  id : Int
  name : String
  users : User nton ( cascade = all, inverse = yes )
  authorities : Authority 1ton ( inverse = yes )
}

entity Authority {
  id : Int
  name : String ( length = 32, notnull = yes, unique = yes )
}

entity User {
  id : Int
  name : String ( length = 32, notnull = yes )
  created : TimeStamp
  changed : TimeStamp
  email : String
  timezone : String
  country : String ( length = 64 )
```

Abbildung 2: Auszug des Domänenmodells

4 Erweitertes System

Im Folgenden werden die Featuresets des erweiterten Systems, welche an das Kernsystem angedockt werden, kurz vorgestellt.

4.1 Typisierung und Sichten der Ressourcen

Insbesondere bei der Eingabe in Formularfelder wird häufig eine Ergebnisseite nach vorherigem Laden angezeigt. Eine ergonomischere Bedienung wird möglich durch eine asynchrone Behandlung der Suchanfrage, d. h. der Begriff wird eingegeben, nach einer Sekunde Inaktivität (kein Tippen) werden die Daten transferiert, die entsprechenden Ergebnisse ohne Laden einer weiteren Seite rechts gelistet. Die Suchergebnisse sind unterteilbar anhand ihrer Datentypen (PDF, Webseite, Bild etc.) und können entsprechend sortiert als auch gruppiert werden. Die dafür nötige Typisierung der Ressourcen spezialisiert den bislang einzigen Typus „Bookmark“.

4.2 Importfunktion

Die meisten Benutzer ziehen bislang die browserinterne Verwaltung zur Speicherung der Bookmarks heran. Um einen schnellen Umstieg auf die ortsunabhängigen Bookmarks in myPIM zu ermöglichen wird eine Importfunktion als Migrationshilfe zur Verfügung gestellt.

4.3 Mirroring

Um Ressourcen dauerhaft verfügbar zu halten und gleichzeitig Veränderungen in den Inhalten nachvollziehen zu können ist eine Spiegelung von Ressourcen in einem Reposi-

238

tory notwendig. Beim Setzen des Bookmarks wird die aktuelle Webseite gespeichert und ist somit in einem eigenen Archiv sicher aufbewahrt. Ein Verlust der Inhalte ist damit – auch ohne manuelle Sicherheitskopien – verhindert.

5 Fazit

Die ersten Evaluierungen haben gezeigt, dass nicht nur die besagte Zielgruppe (Forschungs- und Lern-/Lehrkontext) von myPIM profitieren können, sondern eigentlich jeder, der eine gewisse Anzahl von (Online-)Ressourcen zu verwalten hat. Die Kombination der Prozesse UCD und FDD sowie der leichtgewichtigen, textuellen DSL in Xtext hat den Weg bereitet zu einer einfachen Entwicklung und Weiterentwicklung des leichtgewichtigen Kollaborationswerkzeugs myPIM.

Literaturverzeichnis

[Al05] Allen, D.: Wie ich Dinge geregelt kriege – Selbstmanagement für den Alltag. Getting Things Done. 3. Auflage, Piper, München, 2005

[AZ07] ARD/ZDF-Onlinestudie: http://www.ARD-ZDF-Onlinestudie.de, Stand: 10.06.2007.

[CLL99] Coad, P., Lefebvre, E. & De Luca, J.: Java Modeling in Color With UML: Enterprise Components and Process. Prentice Hall, Upper Saddle River, 1999.

[CMK88] Carrol, J.M. Mack, R.L. & Kellog W.A.: Interface Metaphors and User Interface Design. In Helander, M. (Ed.). Handbook of Human Computer Interaction. Elsevier, Amsterdam, 1988; S. 67-85.

[Ho05] Holtzblatt, K.: Rapid Contextual Design: A How-to Guide to Key Techniques for User-Centered Design. Morgan Kaufmann, London, 2005.

[Ki06] Kindsmüller, M. C.: Personal Knowledge Management & Social Software. Research Colloquium Cognitive Systems. Universität Bamberg, 2006.

[KKH08] Krohn, T., Kindsmüller, M.C., Herczeg, M.: myPIM: A Graphical Information Management System for Web Resources. In: Ågerfalk, P. J., Delugach, H., Lind, M. (eds.) 3rd Int. Conference on Pragmatic Web. ICPW '08, vol. 363. ACM, New York; S.3-12. 2008.

[KMM09] Kindsmüller, M. C., Melzer, A., & Mentler, T..: Online Communities and Community Building. In M. Khosrow-Pour (Ed.) Encyclopedia of Information Science and Technology, 2nd Edition. S. 2899-2905 Hershey, PA: Information Science Publishing. 2009.

[Ko03] Koskinen, J.: Software Maintenance Costs, Information Technology Research Institute, ELTIS-project, University of Jyväskylä, Stand: 21.07.2009. Verfügbar auf: http://users.jyu.fi/~koskinen/smcosts.htm

[Ku05] Kuri, J.: Google will in 300 Jahren am Ziel sein. Heise online News. C't – Das Magazin für Computer und Technik, 10.10.2005. Verfügbar unter: http://www.heise.de/newsticker/meldung/64749.

[La04] Lauesen, S.: User Interface Design. A Software Engineering Perspective. Addison Wesley, London, 2004.

[SB07] Statistisches Bundesamt: Nutzung von Informationstechnologie in Unternehmen – Ergebnisse für das Jahr 2006, Statis, Wiesbaden,; S. 32-34, 2007

[SG06] Stengel, R. & Grossman, L.: You – Yes, You – Are TIME's Person of the Year. TIME Magazine, 26/2006; S. 28-29., 2006

[Wo09] WorldWideWebSize: http://www.worldwidewebsize.com, Stand: 04.07.2009.

[Xt09] Xtext, Eclipse TMF: http://xtext.org, Stand: 21.07.2009.

239