ulm university universität
uulm

Hauptseminar

Aktuelle Themen des Requirements Engineering

Verwaltung großer Anforderungsmengen mit Hilfe von NLP

Martin Rauscher

20. Januar 2010

1 Einführung

Requirements Engineering (RE) ist längst ein integraler Bestandteil der Entwicklungsprozesse vieler Firmen, da es sich in jeder Hinsicht bewährt hat [LM04]. Jedoch bringt die Erstellung und Pflege großer Mengen an bestehenden und oft auch schnell wachsenden Anforderungen neue Herausforderungen mit sich. Insbesondere die unstrukturierte natürliche Sprache, in der Anforderungen meist formuliert sind, stellt ein Problem dar.

Dieses Problem ist längst bekannt. Bereits 1983 schlug Chen in [Che83] eine Methode zur Erstellung von Modellen aus englischen Texten vor.

Diese Arbeit soll eine Einführung in NLP (vgl. Abschnitt 2) im RE-Bereich bieten. Dabei wird insbesondere auf einen Lösungsansatz mittels Ähnlichkeitsanalyse von Anforderungen eingegangen. Zum Schluss werden verschiedene andere Ansätze aufgezeigt, die die Bandbreite an Möglichkeiten in diesem Bereich aufzeigen sollen.

1.1 Warum natürlichsprachliche Anforderungen

Die Vorteile von natürlicher Sprache gegenüber formalen Methoden sind offensichtlich: (nach [Poh07])

Universalität. Natürliche Sprache passt sich jeder Domäne an.

Flexibilität. Natürliche Sprache ermöglicht jeden beliebigen Detaillierungsgrad.

Einfachheit. Natürliche Sprache muss Niemandem beigebracht werden und ist für Jeden verständlich.

Aufgrund dieser Vorteile, und da oft schon beträchtliche Mengen an Anforderungen in textueller Form vorliegen, ist es nicht abzusehen, dass irgendein Formalismus die natürliche Sprache im RE ablösen wird [LM04].

Diesen Vorteilen stellen sich aber auch viele schwerwiegende Probleme entgegen.

1.2 Probleme natürlichsprachlicher Anforderungen

Bei unüberschaubaren Mengen an bestehenden Anforderungen, kann es leicht passieren, dass Anforderungen widersprüchlich oder doppelt erstellen werden. Dies kann zu Inkonsistenz und Redundanz führen. Die scheinbar einfachste Möglichkeit diesen Problemen zu begegnen, wäre entweder die Anzahl der Anforderungen zu verringern oder das Personal aufzustocken. Das ist aber trivialerweise nicht zufriedenstellend, oft nicht möglich und selten zielführend[1] [oDuVG05]. Im Folgenden soll deshalb aufgezeigt werden, wie moderne Methoden des *Natural Language Processing* (NLP, zu Deutsch *Computerlinguistik*) hier Abhilfe schaffen können.

[1] „Adding manpower to a late project makes it later." *Fred Brooks*

2 Natural Language Processing

NLP ist eine Disziplin der Informatik deren eigentliches Ziel es ist Texte, die in uneingeschränkter menschlicher Sprache vorliegen, zu verstehen. Dieses Ziel ist jedoch noch immer eine Herausforderung, die nur in wenigen Spezialfällen zufriedenstellend gelöst werden konnte.

Deshalb unterscheidet man zwischen Natural Language Processing (Die *Verarbeitung* natürlicher Sprache) und Natural Language Understanding (Das *Verstehen* natürlicher Sprache). Für den Großteil der uns hier interessierenden praktischen Problemstellungen ist NLP völlig ausreichend.

E. D. Liddy gibt folgende Definition für NLP an: (übersetzt nach [Lid10])

> NLP beschreibt eine Reihe theoretisch motivierter, rechnergestützer Techniken zur Analyse und zur Repräsentation natürlich auftretender Texte auf einer oder mehrerer Stufen der Sprachanalyse, um menschenähnliche Sprachverarbeitung für eine Reihe von Aufgaben und Anwendungen zu erreichen.

Auch wenn hier also das eigentliche Sprachverständniss ausgeklammert wird, umfasst diese Definition doch zwei große Teilbereiche des NLP: *Sprachverarbeitung* und *Spracherzeugung*.

Sprachverarbeitung, das Thema mit dem sich diese Arbeit beschäftigt, ist der Teilbereich des NLP, in dem es darum geht Texte in eine maschinenverständliche Form, auf Bedeutungsebene, zu bringen. (Das bedeutet *nicht*, dass der PC die Bedeutung *verstehen* muss.)

Spracherzeugung ist sozusagen der umgekehrte Weg; nämlich aus einem Netzwerk an Informationen einen Text in natürlicher Sprache (auf einem bestimmten Abstraktionsniveau) zu erzeugen.

Auch wenn es theoretisch höchst interessant wäre, z.B. ein Anforderungsdokument das in einer sehr „technisierten" Form vorliegt auf eine auch für Manager verständliche Form zu bringen, ist hier nur die Sprachverarbeitung von Interesse, da im RE hauptsächlich die Eigenschaften bereits bestehender Texte im Vordergrund stehen.

3 NLP als Lösung

Bei der Verwaltung von Anforderungen - egal ob bei der Erstellung, der Pflege oder der Aktualisierung - ist die Information, ob die aktuelle Anforderung inhaltlich mit einer bereits Bestehenden identisch ist oder ob sie mit anderen Anforderungen in Zusammenhang steht, enorm wichtig. Jedoch ist sie aufgrund der meist großen Masse an bereits bestehenden Anforderungen aufwändig zu bestimmen. Mit Hilfe der Zusammanhangsinformation können Duplikate vermieden, implizite Links gefunden, sowie Widersprüche aufgedeckt werden. Diese können dann vom Benutzer behoben werden.

Auch bei der Validierung und der Verifizierung von Anforderungen kann diese Information von großem Nutzen sein, da sich so Zusammenhänge effizient finden lassen. Ebenso spielt sie beim Änderungsmanagement eine herausragende Rolle, da der Zusammenhang von Anforderungen zentraler Bestandteil von Impact-Analysen ist.

3.1 Ähnlichkeit von Anforderungen

Da die aktuelle Forschung noch nicht in der Lage ist die tatsächliche Bedeutung von Texten automatisiert zu erfassen, ist eine naheliegende Möglichkeit sich rein auf die lexikalischen Eigenschaften der Texte zu beschränken. Hierbei ist die Ähnlichkeit zweier Anforderungen durch das Auftreten gemeinsamer Wörter definiert. Es ist daher naheliegend, dass ähnliche Anforderungen entweder Duplikate sind oder in einem engen Zusammenhang stehen.

Bevor jedoch die Ähnlichkeit bestimmen werden kann, müssen die Anforderungen zuerst auf eine Form gebracht werden, in der sie einfacher zu verarbeiten sind.

3.2 Vereinfachen von Anforderungen

Im Wesentlichen ist eine Anforderung eine Aneinanderreihung von Worten. Einige dieser Worte, wie z.B. Artikel, erfüllen rein grammatikalische Zwecke und sind nicht von Interesse. Desweiteren ist nur die Grundform von Verben interessant und Interpunktionszeichen können ebenfalls getrost ignoriert werden.

Den Prozess eine Anforderung in einzelne Wörter zu zerlegen, wobei z.B. auch „can't" und

„requirements engineering" als ein (zusammengesetztes) Wort betrachtet werden, nennt man *Tokenization* [oDuVG05].

In einem nächsten Schritt, dem so genannten *Stemming*, werden die Wörter durch Anwendung bestimmter Regeln (und einem Wörterbuch für die Ausnahmen) auf ihre Grundform gebracht.

Abschließend werden mit Hilfe spezieller Lexika die Worte, die keine Bedeutungsträger sind, entfernt. (*Stop Word Removal*)

3.3 Datenstrukturen für Anforderungen

Die lexikalisch vereinfachten Anforderungen können auf verschiedene Weisen gespeichert werden. Da aber die Reihenfolge der einzelnen Wörter als unbedeutend angenommen werden kann[2], ist ein Gewichtsvektor, dessen einzelne Komponenten die Relevanz eines Tokens angeben, die effizienteste Darstellungsform. Die Datenbank enthält dabei eine Tabelle aller Tokens, die in irgendeiner Anforderung auftreten.

Die Relevanz eines Tokens kann auf die üblichen Arten bestimmt werden, z.B. einfach als die Häufigkeit des Tokens oder mit der bekannten Formel $1 + \log_2 n$ [oDuVG05].

4 Vergleich verschiedener Ähnlichkeitsmaße

Nachdem die Anforderungen auf eine einheitliche Form gebracht wurden, kann nun die Ähnlichkeit von Anforderungspaaren bestimmt werden. Im Bereich des NLP haben sich unter anderem das *Dice*, *Jaccard* und das *Cosinusmaß* etabliert [Man99].

Ähnlichkeitsmaß	Definition
Dice	$\frac{2\lvert X \bigcap Y\rvert}{\lvert X\rvert \bigcup \lvert Y\rvert}$
Jaccard	$\frac{\lvert X \bigcap Y\rvert}{\lvert X \bigcup Y\rvert}$
Cosinus	$\frac{\lvert X \bigcap Y\rvert}{\sqrt{\lvert X\rvert \times \lvert Y\rvert}}$

Tabelle 1: Ähnlichkeitsmaße (Quelle: [Man99, S. 299])

Diese verschiedenen Maßzahlen unterscheiden sich hinsichtlich Performance, Recall, Precision und Acuracy[3], wie man in Abbildung 1 auf Seite 4 erkennen kann. Diese Unterschiede sind allerdings sehr kontextabhängig. Daher soll diese Abbildung nur als Beispiel verstanden werden. Hier soll deutlich werden, dass im speziellen Fall man sehr genau testen sollte welches Maß angebracht ist.

5 Verbesserungsmöglichkeiten

Auch wenn die Methode in ihrer bisher vorgestellten Form schon einen großen Fortschritt darstellt, gibt es noch einige Möglichkeiten sie zu verbessern.

5.1 Anforderungscluster

Mit Hilfe der in Abschnitt 4 vorgestellten Maße kann eine binäre Relation über allen Anforderungen erzeugt werden. Wird nun die transitive Hülle dieser Relation gebildet, erhält man Cluster von Anforderungen, die alle in einem gewissen Zusammenhang stehen.

Aufgrund der Tatsache, dass viele dieser so „entdeckten" Anforderungspaare nicht direkt als ähnlich erkannt wurden, sind sie möglicherweise besonders interessant, da sie sehr subtile Verbindungen aufzeigen können.

Allerdings können auf diese Weise sehr große Cluster entstehen (z.B. >50). Deshalb ist es sinnvoll hier eine künstliche Begrenzung einzubauen um sicherzustellen, dass sich diese Cluster noch

[2] Die Reihenfolge könnte nur die Semantik verändern, aber bei einem Vergleich auf lexikalischer Ebene ist sie unwichtig.
[3] Da die Bedeutung der Worte nicht direkt ersichtlich ist, habe ich auf eine Übersetzung verzichtet.

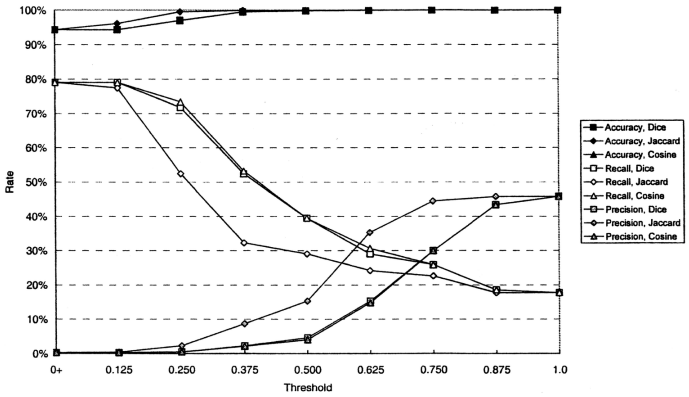

Abbildung 1: Verhalten verschiedener Ähnlichkeitsmaße (Quelle: [oDuVG05])

effizient von einem Menschen beurteilen lassen können.

5.2 Erweiterte Ähnlichkeitsanalysen mittels Thesauri

Eine Limitierung, die auch durch den Clusteransatz nicht beseitigt wird, besteht darin, dass keine Erkennung von Synonymen oder allgemeiner inhaltlich ähnlichen Wörtern erfolgt. Um dieses Problem mit realistischem Aufwand zu beseitigen, bieten sich anwendungsspezifische Thesauri an.

Mit Thesauri lassen sich auch irrtümliche Treffer vermeiden, da sich mit ihrer Hilfe Wörter in speziellere Anwendungskontexte einordnen lassen. Auf diese Weise entfallen Anforderungspaare, die zwar viele gemeinsame Wörter besitzen, jedoch in einem unterschiedlichen Kontext stehen und somit doch nicht in unserem gewünschten Sinn ähnlich sind.

6 Andere NLP-Ansätze

Der hier vorgestellte Ansatz ist nur einer von vielen NLP-basierten Ansätzen im RE. Im Folgenden sollen einige dieser anderen Ansätze vorgestellt werden.

6.1 Eingeschränkte Sprache

Ein verbreiteter Ansatz um dem Problem der inhärent komplexen menschlichen Sprache zu begegnen, ist die Verwendung von eingeschränkten Sprachen. Hierbei werden typischerweise Schablonen verwendet, die eine feste Satzstruktur vorgeben.

Auf diese Weise ist es leicht möglich den genauen Kontext einer Anforderung zu bestimmen, und somit einfacher zu erkennen, ob sich zwei Anforderungen tatsächlich auf die selbe Entität beziehen.

6.2 (Halb-)Automatische Erzeugung formaler Spezifikationen

In [ea02] schlagen die Autoren eine Methode vor, bei der aus einer Spezifikation eine stark annotierte Version erzeugt wird, indem Phrasen aus dem ursprünglichen Dokument mit einem Domänenlexikon verglichen werden. Dieses Dokument kann dann mit beträchtlich geringerem Aufwand von einem Experten in eine formale Zwischensprache übersetzt werden, welche noch relativ nah an der natürlichen Sprache liegt. Aus dieser Zwischensprache kann dann automatisiert ein Modell erzeugt werden.

Ein ähnlicher Ansatz, der jedoch noch eine Verbindung zum Model-Driven-Development herstellt, wird in [con09] vorgestellt. Um einen vollständig automatisierten Ansatz zu erreichen schlagen die Autoren den Einsatz einer eingeschränkten Sprache vor(vgl. 6.1). Da die Autoren Validierung und Verifikation direkt auf den Dokumenten in (eingeschränkter) natürlicher Sprache betreiben wollen, wird es für die Stakeholder viel einfacher die tatsächliche Korrektheit der Dokumente zu überprüfen, da diese in einer für die Stakeholder verständliche Sprache bleiben.

6.3 Traceability zwischen Modellen und Anforderungen

Auch wenn natürliche Sprache vorherrschend ist, setzt ein Großteil der Firmen an irgendeinem Punkt ihres Entwicklungsprozesses Modelle ein. Da diese Modelle meist mit anderen Tools erzeugt werden als die Anforderungen dazu, gibt es keine direkten Links zwischen Anforderungen und Modell.

Diesem Problem nehmen sich Cerbah und Euzenat in [FC01] an. Ihr Ansatz besteht aus zwei Tools:

- ein Werkzeug, das aus den Ursprungsdokumenten eine hierarchische Terminologiestruktur sowie Rück-Links erzeugt.
- und einem Modellmanager, der eine rudimentäre Klassenstruktur aus den Daten des ersten Tools erzeugen kann.

Diese Klassenstruktur kann dann vom Benutzer vervollständigt werden, während sie noch immer zurückverfolgbar bleibt.

6.4 Unterstützung von Dokumentenanalysen

In der industriellen Praxis findet sich ein (Anforderungs-)Entwickler oft großen Mengen an Anforderungen oder Dokumentationen in anderer Form gegenübergestellt. In [PR01] wird deshalb ein Ansatz vorgestellt, der mit Hilfe von probabilistischen NLP Techniken versucht herauszufinden, welche Teile eines Dokuments besonders „interessant" sind. Die Autoren vertreten die Auffassung, dass es nicht möglich ist irgendeinen Teil des RE Prozesses vollständig zu automatisieren. Deshalb beschränkt sich dieser Ansatz rein auf die Suche und Präsentation der gefundenen Stellen.

Der Anfang dieses Ansatzes besteht darin, eine Häufigkeitsanalyse durchzuführen. Dabei werden die Häufigkeiten der Wörter in einem Text mit den Häufigkeiten in einem sehr großen Standardtext verglichen. Besondere Häufungen lassen dann darauf schließen, dass ein Wort im aktuellen Kontext eine andere Bedeutung hat als normalerweise. In Kombination mit einer Part-Of-Speech-Analyse, kann dann erkannt werden, wann etwas Wichtiges mit etwas Wichtigem passiert. Bei der Part-Of-Speech-Analyse wird den Wörtern ihre jeweilige Wortart beigefügt, um besser semantische Eigenschaften bestimmen zu können.

6.5 Anwendungsfallanalysen mittels erweiterter Petri Netze

In [WJL98] beschreiben die Autoren sogenannte *Constraints-based Modular Petri Nets* (CMPNs), mit deren Hilfe sie Anwendungsfalldiagramme formal beschreiben können, um sie so auf Vollständigkeit und Konsistenz überprüfen zu können.

Dieses Verfahren ist zwar noch kaum automatisiert, aber es zeigt bereits, dass aus sehr informalen Modellen, wie z.B. Anwendungsfalldiagrammen, formale Beschreibungen gewonnen werden können. Da diese Arbeit bereits über elf Jahre alt ist, ist wohl zu erwarten, dass inzwischen ein viel größerer Teil dieses Verfahrens automatisiert werden könnte.

6.6 Erzeugung von Spezifikationen aus UML

Den genau umgekehrten Weg wie alle bisher erwähnten Methoden gehen Meziane et al in [FM08]. Um das bereits in 6.2 erwähnte Problem, dass die Stakeholder die formalisierten Anforderungen nicht mehr verstehen zu beseitigen, erzeugen sie aus UML[4]-Klassendiagrammen wieder eine natürlichsprachliche Spezifikation. Um mehr als nur eine triviale textuelle Serialisierung des Diagramms zu bekommen, verwenden die Autoren eine Ontologie in WordNet, um mehr Informationen aus den Namen im Diagramm zu erhalten.

7 Fazit

Wie bereits aus dem kleinen Auszug an verschiedenen Ansätzen klar wird, gibt es reichlich Möglichkeiten mit Hilfe von NLP den RE Prozess zu unterstützen. Da es auch nicht abzusehen ist, dass die Industrie auf formale(-re) Methoden umsteigen wird, sind Fortschritte im Bereich des RE mittelfristig wohl hauptsächlich im NLP Umfeld zu suchen. [LM04] zeigt ganz klar den Bedarf in der Industrie.

Der größte Reiz des NLP-based RE ist sicher die besondere Praxisnähe, da sein Einsatz direkt

[4] UML Version 1.5

Erfolge in Form von beschleunigten Arbeitsprozessen zeigt. Aufgrund dessen ist zu erwarten, dass dieser Forschungstrend sicher auch in der Industrie ankommen wird. Nicht zuletzt deshalb, weil mit diesen Methoden teilweise die Schwächen der Tools sowie der Prozesse ausgeglichen werden können. Denn oft bieten die Tools noch nicht einmal eine vollständige Unterstützung für Links verschiedener Art, und die Prozesse ziehen deshalb weniger Nutzen aus dem RE, als eigentlich möglich wäre.

Literatur

[Che83] Peter Pin-Shan Chen, *English sentence structure and entity-relationship diagrams*, Master's thesis, Graduate School of management, University of California, 1983. 1

[con09] *A controlled natural language approach for integrating requirements and model-driven engineering*, vol. Fourth International Conference on Software Engineering Advances, 2009. 6.2

[ea02] B. Andersson et al. (ed.), *Semi-automated content extraction from specifications*, vol. LNCS 2553, Springer-Verlag, 2002. 6.2

[FC01] J□r□me Euzenat Farid Cerbah, *Natural language processing and information systems*, Lecture Notes in Computer Science, vol. 1959, ch. Using Terminology Extraction to Improve Traceability from Formal Models to Textual Requirements, pp. 115–126, Springer, 2001. 6.3

[FM08] Sophia Ananiadou Farid Meziane, Nikos Athanasakis, *Generating natural language specifications from uml class diagrams*, Requirements Engineering **13** (2008), 1–18. 6.6

[Lid10] E. D. Liddy, *Encyclopedia of library and information science, 2nd ed.*, Marcel Decker, Inc., Stand: Jan. 2010. 2

[LM04] Pierluigi Novi Inverardi Luisa Mich, Mariangela Franch, *Market research for requirements analysis using linguistic tools*, Requirements Engineering **9** (2004), 40–56. 1, 1.1, 7

[Man99] Christopher D. Manning, *Foundations of statistical natural language processing*, Mit Press, 1999. 4, 1

[oDuVG05] Johann Natt och Dag und Vincenzo Gervasi, *Managing large repositories of natural language requirements*, Engineering and Managing Software Requirements, Springer, 2005. 1.2, 3.2, 3.3, 1

[Poh07] Klaus Pohl, *Requirements engineering*, dpunkt-Verlag, 2007. 1.1

[PR01] Roger Garside Pete SawyerS Paul Rayson, Luke Emmet, *Natural language processing and information systems*, Lecture Notes in Computer Science, vol. 1959, ch. The REVERE Project: Experiments with the Application of Probabilistic NLP to Systems Enineering, pp. 288–300, Springer, 2001. 6.4

[WJL98] Yong Rae Kwon Woo Jin Lee, Sung Deok Cha, *Integration and analysis of use cases using modular petri nets in requirements engineering*, IEE Transactions on Software Engineering **24** (1998), no. 12, 1115–1130. 6.5