MOQARE = “Misuse-oriented Quality Requirements Engineering” – Über den Nutzen von Bedrohungsszenarien beim RE von Qualitätsanforderungen

Dr. Andrea Herrmann, Prof. Barbara Paech
Universität Heidelberg, Fakultät für Mathematik und Informatik, Software Engineering Group, 69042 Heidelberg;
{herrmann;paech}@informatik.uni-heidelberg.de

Das Ziel der hier vorgestellten Arbeit war die Entwicklung einer Methode zur systematischen Herleitung und Dokumentation von nicht-funktionalen Anforderungen (auch Qualitätsanforderungen genannt) und deren Konkretisierung in einer Form, dass sie realisierbar und prüfbar sind. Beispielsweise: Wie kommt man von der vagen Anforderung „Die Daten sollen richtig sein“ zu Anforderungen wie „Use Case ‚Eingabe von Patientenstammdaten’ soll in 99% der Fälle fehlerfrei ausführbar sein“ (Usability-Anforderung an Use Case), „Bei der Eingabe der Daten erfolgt eine Plausibilitätsprüfung“ (neue funktionale Anforderung) oder „Der Administrator führt regelmäßig eine Datenbereinigung durch“ (neue funktionale Anforderung)? Eine solche Methode würde zusätzlich die Herkunft solcher Anforderungen nachvollziehbar und Wissen wieder verwendbar machen.

Im Bereich der Sicherheit benutzt man für die Konkretisierung des Qualitätsattributes „Sicherheit“ oft Bedrohungsszenarien oder Misuse Cases [1-5] die beschreiben, was nicht passieren bzw. wie das System auf Angriffe reagieren soll. Nur indem Missbrauchsfälle betrachtet werden, werden hier die Anforderungen vollständig. Wir verfolgten nun die Idee, Misuse Cases auch für alle anderen Qualitätsattribute zu verwenden, mit geringfügigen Konzeptänderungen. Beispielsweise ist der Aktor eines Misuse Cases, der Misuser, im allgemeinen Fall nicht unbedingt ein Angreifer, sondern kann auch ein normaler Mitarbeiter sein, der nur seine Arbeit machen will, dabei aber versehentlich ein Problem verursacht, oder auch ein Entwickler oder Administrator.

Für unsere neue Methode MOQARE brauchen wir die im Folgenden beschriebenen Konzepte (siehe auch Abbildung 1). Wir gehen von den zu erreichenden Geschäftszielen (Business Goals) aus. Diese werden einerseits durch Geschäftsschäden (Business Damages) bedroht, andererseits durch Qualitätsziele (Quality Goals) des IT-Systems unterstützt. Ein Qualitätsziel besteht nach unserer Definition stets aus einem zu schützenden Wert (Asset) und einem Qualitätsattribut, z.B. „Integrität der Daten“. Die Qualitätsziele werden durch Qualitätsmängel (Quality Deficiencies) bedroht, die zugleich auch die Geschäftsschäden verursachen. Die Auslöser für diese Qualitätsmängel sind die Misuse Cases, die beschrieben werden durch eine Bedrohung (Threat = eine Aktion), einen Misuser, oft auch durch eine Schwachstelle (Vulnerability), die die Bedrohung ermöglicht oder erleichtert, und durch den verursachten Qualitätsmangel. Um das Existieren der Bedrohung oder die Durchführung des Misuse Cases zu verhindern, abzuschwächen oder wenigstens zu entdecken werden Gegenmaßnahmen (Countermeasures) definiert. Diese Gegenmaßnahmen sind entweder neue Qualitätsziele oder konkrete nicht-funktionale Anforderungen, Architekturanforderungen, neue funktionale Anforderungen u.v.m..

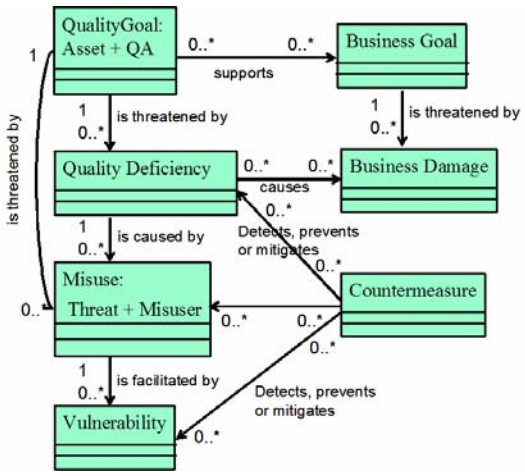

Abbildung 1: Konzepte der MOQARE Methode

Das Ergebnis dieser Analyse wird in einem hierarchischen „Misuse Tree“ visualisiert (siehe Abbildung 2). Unterstützt wird die Methode durch Listen von Bedrohungen, Schwachstellen und Gegenmaßnahmen, die wir durch Literaturrecherche zusammengestellt haben. Gefunden werden hierbei nicht nur Anforderungen an die Software, sondern auch an den Softwareentwicklungsprozess und den Betrieb. Da MOQARE teilweise zu neuen funktionalen Anforderungen oder zu nicht-funktionalen Anforderungen an funktionale Anforderungen führt (wie z.B. die oben genannte Usability-Anforderung an einen Use Case), müssen diese in die funktionale Beschreibung des IT-Systems integriert werden. Dies haben wir erfolgreich anhand der Methode TORE [6] durchgeführt, die bei uns am Lehrstuhl verwendet wird.

MOQARE und TORE trennen zwischen der Anforderungsbeschreibung auf der technik-unabhängigen Geschäftsebene und der technischen Ebene. Auf der Geschäftsebene fügen sich die Geschäftsziele, Geschäftsschäden und Qualitätsmängel ein, außerdem die Use Cases, die in ihrer Granularität den Misuse Cases entsprechen. Auf technischer Ebene finden die Anforderungen an die Architektur, Benutzeroberflächen und weitere technische Anforderungen ihren Platz.

Diese Methode wurde anhand einer Fallstudie anhand einer Patienten- und Behandlungsdatenbank des Uveitiszentrums Heidelberg getestet und verbessert. Aus unserer Sicht hat sich MOQARE dabei durch folgende Vorzüge ausgezeichnet:

- Die systematische Detaillierung in Verbindung mit Checklisten führte zu einer besseren Vollständigkeit.
- Das Ergebnis waren nicht nur Anforderungen IT-System, sondern auch an den Software Engineering Prozess und den Betrieb des Systems.
- Die Methode ist auch für nicht-technische Stakeholder anschaulich.
- Die Geschäftsziele erlauben eine Priorisierung von Anforderungen (je nachdem ob sie die Geschäftsziele direkt oder indirekt unterstützen) und dokumentieren eine Begründung von Anforderungen.
- Die integrierte Beschreibung zusammen mit funktionalen Anforderungen führt zu einer Gleichwertigkeit und besserer gemeinsamer Konfliktlösung zwischen Anforderungen.

Weitere geplante Arbeiten sind die qualitative Bewertung und Priorisierung der Bedrohungen und daraus abgeleiteten Anforderungen, die Lösung von Konflikten zwischen Anforderungen und wie man aus diesen Anforderungen Testfälle ableiten kann.

Danksagungen: Diese Arbeit ist Teil des Forschungsprojekts SIKOSA, das vom Ministerium für Wissenschaft, Forschung und Kunst Baden-Württemberg gefördert wird. Wir danken den Kolleg/innen innerhalb des Lehrstuhls, innerhalb von SIKOSA und am Uveitiszentrum für zahlreiche wertvolle Diskussionen.

[1] McDermott, J., Fox, C.: Using Abuse Case Models for Security Requirements Analysis. 15th Annual Computer Security Applications Conference (1999) 55-6

[2] Sindre, G., Opdahl, A.L.: Eliciting Security Requirements by Misuse Cases. Proceedings of TOOLS Pacific 2000 (2000) 120-131

[3] Sindre, G., Opdahl, A.L.: Templates for Misuse Case Description. REFSQ - Proc. Of Internat. Workshop on Requirements Engineering for Software Quality (2001) 125-136

[4] Allenby, K., Kelly, T.: Deriving Safety Requirements Using Scenarios. Proceedings of the 5th International Symposium on Requirements Engineering (2001) 228-235

[5] Alexander, I.: Initial Industrial Experience of Misuse Cases. Proc. Of IEEE Joint Internat. Requirements Engineering Conf. (2002) 61-68

[6] Paech, B., Kohler, K.: Task-driven Requirements in object-oriented Development. In J. Leite, J. Doorn, (eds.) Perspectives on Requirements Engineering, Kluwer Academic Publishers, 2003

Abbildung 2: Misuse Tree