Fraunhofer Institut
Experimentelles
Software Engineering

Extraktion von Produktfamilienanforderungen aus Benutzerdokumentation

Autoren:
Isabel John
Jörg Dörr

Veröffentlicht im Buch
"Produktfamilienengineering"

IESE-Report Nr. 111.03/D
Version 1.0
Oktober 2003

Eine Publikation des Fraunhofer IESE

Das Fraunhofer IESE ist ein Institut der Fraunhofer-Gesellschaft.
Das Institut überträgt innovative Software-Entwicklungstechniken, -Methoden und -Werkzeuge in die industrielle Praxis. Es hilft Unternehmen, bedarfsgerechte Software-Kompetenzen aufzubauen und eine wettbewerbsfähige Marktposition zu erlangen.

Das Fraunhofer IESE steht unter der Leitung von
Prof. Dr. Dieter Rombach
Sauerwiesen 6
67661 Kaiserslautern

Abstract

Existierende Software wird oft von Dokumentation wie beispielsweise Benutzerhandbüchern begleitet. Diese Dokumentation ist eine wichtige Quelle für eine neue Produktfamilie. Durch das gezielte Nutzen von existierender Dokumentation können Experten entlastet und existierende Systeme reibungsloser integriert werden. Dieses Kapitel beschreibt eine Methode zur strukturierten Extraktion von Anforderungen an eine Produktfamilie aus der Benutzerdokumentation existierender Altsysteme. Mit Hilfe der Methode können strukturiert Features, Use Cases und andere Anforderungen in Dokumentationen existierender System gefunden und als Anforderungen an eine neue Produktfamilie integriert werden.

Schlagworte: Dokument, Dokumentation, Benutzerdokumentation, Reengineering, Altsysteme, Anforderungen, Use Cases, Features, Gemeinsamkeiten, Variabilität, Extraktion

Copyright © Fraunhofer IESE 2003

Inhaltsverzeichnis

1	**Einführung**	**1**
2	**Software Dokumentation**	**2**
3	**Vorgehen bei der Extraktion**	**4**
3.1	Extraktionspattern	5
3.2	Vorbereitung	6
3.3	Analyse	7
3.4	Auswahl	9
4	**Anwendungsbeispiel**	**10**
5	**Zusammenfassung und Ausblick**	**12**
6	**Referenzen**	**13**

Copyright © Fraunhofer IESE 2003

1 Einführung

Die Einführung von Produktfamilientechnologie ist eine Aktivität, die mit erheblichem Zeitaufwand verbunden sein kann, da in allen Entwicklungsbereichen Änderungen im Entwicklungsprozess und in den Artefakten zu erwarten sind. Involviert sind dabei sogenannte Produktfamilieningenieure (z.B. Produktfamilienanalysten, zuständig für die Domänenanalyse), die sich in der Technologie, nicht aber in der Domäne der Produktfamilie auskennen und Domänenexperten aus der Domäne der Produktfamilie (erfahrene Entwickler, Designer oder Anforderungsingenieure). Ein großes Problem dabei ist der Zeitmangel, die Domänenexperten werden im Unternehmen oft noch an anderen Stellen gebraucht, in kleinen und mittleren Unternehmen sind die Experten oft gleichzeitig Geschäftsführer oder Entwicklungsleiter des Unternehmens. Daher besteht oft nicht die Möglichkeit, innerhalb weniger Wochen mehrtägige Workshops mit allen Beteiligten zur Einführung von Produktfamilientechnologien durchzuführen.

Produktfamilien bauen in der Regel auf existierenden Systemen auf. Kaum ein Unternehmen wird ohne schon Systeme in einem Bereich realisiert zu haben direkt eine Produktfamilie mit mehreren Produktfamilien planen. Ohne Kenntnis des Markts und der technischen Realisierungsmöglichkeiten für die geplanten Produkte ist es sehr schwer, eine neue Produktfamilie aufzusetzen und direkt Variabilitäten zu planen. Deswegen kann man oft davon ausgehen dass schon Systeme in der Domäne entwickelt und dokumentiert wurden.

Diese Situation – Domänenwissen existiert im Unternehmen durch das Know-How der zeitlich stark ausgelasteten Experten und in Form von existierenden Systemen und ihrer Dokumentation – ist der Ansatzpunkt des PuLSE™[1]-CaVE Ansatzes (Commonality and Variability Extraction): Der im Café Projekt entwickelte Cave Ansatz nutzt die Benutzerdokumentation existierender Systeme zur Erstellung von gemeinsamen und variablen Anforderungen an eine neue Produktfamilie.

Der CaVE Ansatz ist ein Pattern-basierter Ansatz mit dem mit Hilfe von so genannten Extraktionspattern gemeinsame und variable Teile von Use Cases, Features, Anforderungen etc aus der Benutzerdokumentation von Systemen in einer Domäne extrahiert werden können. Diese Teile dienen dann als Grundlage für ein Domänenmodell. Im folgenden beschreiben wir den Ansatz und seine Grundlagen und geben ein Beispiel aus der industriellen Anwendung.

[1] PuLSE (Product Line Software Engineering) ist ein eingetragenes Warenzeichen des Fraunhofer IESE

Copyright © Fraunhofer IESE 2003

2 Software Dokumentation

Software Dokumentation dient der Kommunikation aller an der Software Entwicklung beteilen Personen. Software Dokumentation wird in [8] definiert als: „Komplex von Informationsträgern zur Beschreibung der Aufgaben, Aktionsträger, Kommunikationswege, Daten- und Arbeitsprozeduren bei der Systementwicklung und Anwendung". Ein Dokument steht dabei für eine Informationsmenge die innerhalb einer Anwendung oder eines Vorgangs als Einheit verstanden wird. Dokumente verfügen über eine Layout Struktur, eine logische Struktur und eine inhaltliche oder semantische Struktur.

Im Laufe der Softwareentwicklung werden viele Arten von Dokumentationen erstellt: Lastenhefte, Pflichtenhefte, Architekturdokumentationen, Dokumentationen im Code, Wartungsdokumente und Benutzerdokumentationen. Eine kürzlich in Deutschland durchgeführte Studie [6] ergab, dass in 87 % der befragten Unternehmen Anforderungsdokumente erstellt werden, in über 90 % der Unternehmen wurden Benutzerdokumentationen erstellt. Dabei war die Benutzerdokumentation die umfangreichste Dokumentation die im Laufe der Entwicklung erstellt wurde (64% der Benutzerdokumentation hatte über 20 Seiten). Gerade bei kleinen und mittleren Unternehmen bleibt oft keine Zeit zur Erstellung von interner Dokumentation, die einzige Dokumentation die erstellt und regelmäßig gewartet wird, ist die Benutzerdokumentation. Benutzerdokumentation beschreibt das System aus Sicht des Benutzers und stellt das System so dar, wie es sein soll. Deshalb ist Benutzerdokumentation eine verfügbare, umfassende und aktuelle Quelle zur Extraktion von Anforderungen an eine Produktfamilie, die auf existierenden Systemen in der Domäne aufbaut. Benutzerdokumentation sollte nach [10] die folgenden Dokumente umfassen:

- Eine funktionale Beschreibung die die Dienste die das System bereitstellt kurz erklärt
- Ein Installationsdokument das erklärt wie das System zu installieren ist
- Ein Einführungshandbuch das eine informelle Beschreibung des Systems und seine ‚normale' Nutzung beschreibt
- Ein Referenzhandbuch das alle Nutzungsmöglichkeiten und Eigenschaften des Systems beschreibt, Fehlermeldungen auflistet und Hilfen zur Fehlerbehebung gibt
- Ein Administrationshandbuch das die Interaktion des Systems mit anderen Systemen und weitergehende Funktionalitäten beschreibt (nicht für alle Systeme sinnvoll)

Copyright © Fraunhofer IESE 2003

In der Praxis sind oft all diese Dokumente oder ein Teil davon in einem Benutzerhandbuch zusammengefasst. All diese Dokumentationsaspekte können, wenn Sie sinnvoll analysiert werden wertvolle Information für ein Domänenmodell liefern. Die funktionale Beschreibung und das Einführungshandbuch können Anforderungen an die Funktionalität der Produktfamilie liefern, das Referenzhandbuch und das Administrationshandbuch können z.B. Wartungsanforderungen liefern.

Im folgenden beschreiben wir, wie man aus der Benutzerdokumentation sinnvolle Informationen für die Scoping- und Domänenanalysephase einer Produktfamilie extrahieren kann. Hierbei können alle oben genannten Arten von Benutzerdokumentation verwendet werden, es werden dann jeweils Anforderungen zu anderen Aspekte des Systems extrahiert.

Copyright © Fraunhofer IESE 2003

3 Vorgehen bei der Extraktion

Wie beim Suchen von Anforderungen in Benutzerdokumenten vorzugehen ist, beschreibt ein dreistufiger Extraktionsprozess. Im ersten Schritt wählt der Produktfamilienanalyst die Ausgangsdokumente aus und zerteilt sie in handhabbare Stücke, so dass die Extraktion der Anforderungen aus den Dokumenten so effektiv und effizient wie möglich erfolgen kann. Weiterhin wählt er aus einer Sammlung von Extraktionspattern jene aus, die für diese Dokumentationsteile geeignet sind. Diese Extraktionspattern, die bei der eigentlichen Suche verwendet werden, beschreiben wo man typischerweise in der Benutzerdokumentation welche Anforderungselemente findet. Im zweiten Schritt identifiziert der Produktfamilienanalyst mit Hilfe der Extraktionspattern die Elemente in den Dokumentationsteilen, die dann in gemeinsame und variable Anforderungsfragmente überführt werden. Erst im dritten Schritt wird der Domänenexperte involviert. Der Produktfamilienanalyst fügt die in den Benutzerdokumenten identifizierten Elemente zu einem vorläufigen, partiellen Produktfamilienmodell zusammen und legt es dem Domänenexperten zur Begutachtung vor. Dieser überprüft die Korrektheit und Vollständigkeit des Produktfamilienmodells aus seinem Experten-Blickwinkel. Dabei führt er gegebenenfalls Änderungen und Ergänzungen durch. Damit entsteht ein partielles Produktfamilienmodell, das nun weiter detailliert und gegebenenfalls durch andere Spezifikationen ergänzt werden kann, für die keine Elemente extrahiert werden konnten. Abbildung 1 gibt eine Übersicht über die verschiedenen Prozessschritte und deren Ein- und Ausgaben.

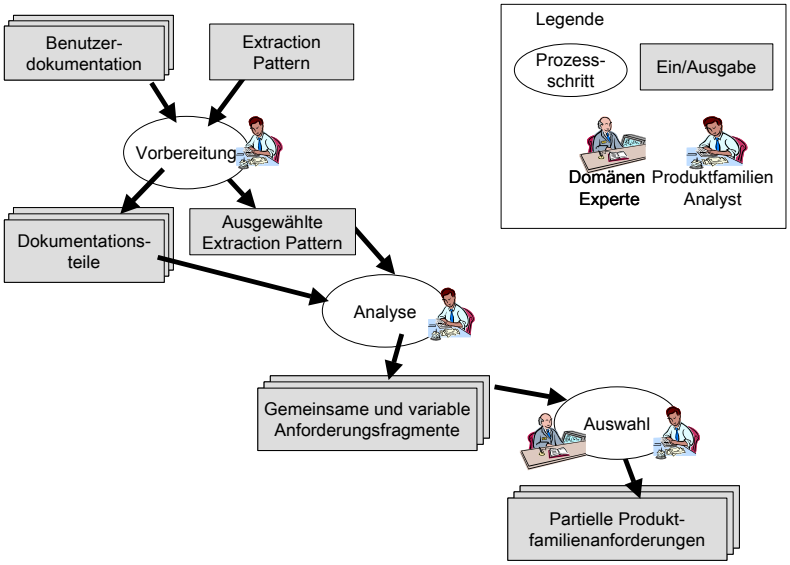

Abbildung 1 Der Extraktionsansatz

Copyright © Fraunhofer IESE 2003

3.1 Extraktionspattern

Das zentrale Element des Extraktionsprozesses sind die Extraktionspattern. Diese wurden auf Basis eines Metamodells erstellt, welches Elemente der Benutzerdokumentation, Anforderungskonzepte, Variabilität und Produktfamilienartefakte beschreibt und Verbindungen zwischen diesen vier Konzepten herstellt. Elemente der Benutzerdokumentation wie beispielsweise Überschriften, nummerierte Listen oder hervorgehobene Wörter werden hier auf Anforderungskonzepte wie Aufgaben, Stakeholder, Aktivitäten, Systemfunktionen und Daten abgebildet. Gleiches oder unterschiedliches Vorkommen dieser Benutzungsdokumentations-Elemente in den verschiedenen Benutzerdokumenten lässt beispielsweise auf Gemeinsamkeiten, Alternativen und Parametrisierungen schließen, welche Elemente des Variabilitätenkonzepts sind. Die um Variabilitäten angereicherten Anforderungskonzepte werden anschließend auf Elemente des Produktfamilienartefaktkonzepts wie beispielsweise variable oder gemeinsame Features, Use Cases, Vorbedingungen oder nicht-funktionale Anforderungen abgebildet. Für den Extraktionsprozess werden vor allem die direkten Extraktionspattern von Elementen der Benutzungsdokumentation (z.B. Überschriften, Listen., Glossareinträge) auf Elemente des Produktfamilienartefaktkonzepts (wie Features, Use Case Namen, Datenanforderungen etc.) verwendet. Abbildung 2 zeigt die Beziehungen zwischen den Konzepten auf. Eine komplette Beschreibung und Diskussion der einzelnen Konzepte befindet sich in [4]. Alle Extraktionspattern sind gemäß einem Template beschrieben. Ein Beispielpattern ist in Tabelle 1 dargestellt. Durch die Verwendung des Templates ist die Erfassung aller Informationen, die für die Charakterisierung eines Pattern notwendig sind, sichergestellt.

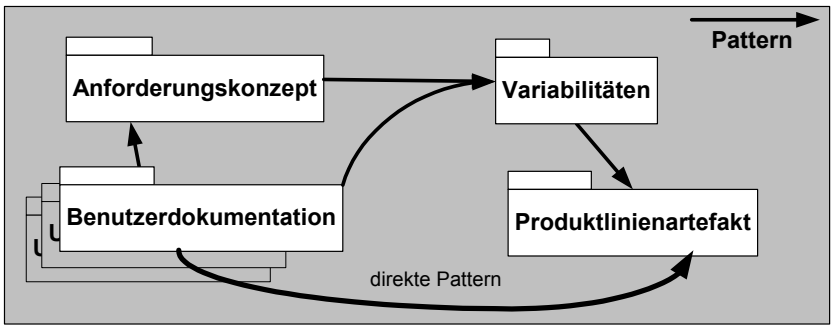

Abbildung 2 1 Beziehungen zwischen Konzepten des Metamodells

Copyright © Fraunhofer IESE 2003

Name :	**#3**	**Heading-to-Feature**
Kurzbeschreibung	Überschriften von Kapiteln und Sektionen sind oder enthalten typischerweise Features.	
Eingabe	Überschrift	
Ausgabe	Feature	
Vollständigkeit	30% (Durchschnittliche Anzahl von gefundenen Elementen in Relation zu der tatsächlichen Anzahl der Elemente)	
Korrektheit	70% (Durchschnittliche Anzahl der relevanten Elemente in Relation zu der Anzahl der gefundenen Elemente)	
Beschreibung	Da Feature Funktionalitäten beschreiben, die für den Benutzer wichtig sind, werden sie an besonders auffälligen Plätzen der Benutzerdokumentation, d.h. in Überschriften gefunden.	

Tabelle 1 Beschreibung eines Extraktionspattern im Patterntemplate

Im Folgenden geben wir Beispiele für Extraktionspattern:

- Phrasen wie "nur bei", "bei Nutzung von", "im Fall von" können Markierungen für Use Case Vorbedingungen sein.
- Nummerierte Listen oder Listen sind Markierungen für geordnete Abarbeitung von sequentiellen Schritten und beschreiben Use Case Beschreibungen (Szenarien).
- Wenn sich numerische Werte in ansonsten identischen Teilen der Dokumentationsteilen unterscheiden deutet dies auf eine Parametrisierung (Parametrische Variabilität) hin.

Insgesamt umfasst die Liste etwa 30 kondensierte Pattern, diese Liste wird aufgrund neuer Erfahrungen stetig erweitert. Bei der Nutzung der Pattern während des Analyse Schritts, wird jeweils die Kurzbeschreibung des Patterns in Form einer Checkliste verwendet.

Im Folgenden werden die drei Schritte des Extraktionsprozesses im Detail beschrieben.

3.2 Vorbereitung

In der Vorbereitungsphase sichtet der Produktfamilienanalyst die existierende Dokumentation. Er wählt 2-4 typische Dokumentationen aus. Dabei sind besonders kleine und große Dokumentationen typisch, da diese meist low-end und high-end Geräten entsprechen und somit viele Variabilitäten identifiziert werden können. Die ausgewählten Dokumentationen werden in Kapitel bzw. Teile (z.B. Inhalt, Kapitel, Glossar) aufgespaltet so dass zusammenhängende Teile verschiedener Dokumentationen parallel verglichen werden können. Erfahrungsgemäß eignen sich Teile von fünf bis 30 Seiten. Ausgehend von der ersten

Copyright © Fraunhofer IESE 2003

Sichtung der Teile wird sich für das weitere Vorgehen entschieden: Unterscheiden sich die verschiedenen Dokumentationen stark voneinander (erfahrungsgemäß ab etwa einem Drittel Unterschiede), wird die eigentliche Analyse zuerst mit einer einzelnen Dokumentation durchgeführt, danach werden die Unterschiede zu den anderen Systemen identifiziert. Unterscheiden sich die Dokumentationen nur wenig kann im weiteren Vorgehen parallel verglichen werden, d.h. die Dokumente werden im Analyseschritt parallel gelesen.

Zusätzlich wählt der Produktfamilienanalyst die anzuwendenden Pattern aus der Sammlung existierender Pattern aus. Dies geschieht anhand verschiedener Kriterien. Beispiele für diese Kriterien sind

- die Notation des Produktfamilienmodells (Möchte ich z.B. Features oder Use Cases extrahieren?),
- die Korrektheit der Pattern (Wenn ich Features extrahiere: wie sicher ist ein identifiziertes Feature wirklich ein Feature?) und
- die Vollständigkeit eines Pattern (Wenn ich Features extrahiere: wie viele Features bleiben unentdeckt?).

Die so ausgewählten Pattern werden in einer Checkliste ausgedruckt, die ausgewählten Teile der Benutzerdokumentationen können ausgedruckt oder am Bildschirm verglichen werden.

3.3 Analyse

In der Analysephase werden Anforderungselemente (z.B. Features, Use Case Namen, Use Case Schritte, nicht-funktionale Anforderungen etc.) mit Hilfe der ausgewählten Extraktionspattern (siehe Sektion 3.1) im Text gesucht. Zusätzlich werden mit Hilfe von Pattern Gemeinsamkeiten und Unterschiede identifiziert. Der Produktfamilienanalyst markiert die identifizierten Elemente im Text und vergibt optional eine Kennzeichnung für das gefundene Element. Abbildung 3 zeigt an einem Bespiel aus der industriellen Anwendung (siehe Abschnitt 4) wie Elemente mit Boxen und Textmarkern markiert werden können. Hier sind als Beispiel Datenelemente, Features und dazugehörige Feature-Werte markiert, ähnlich werden auch andere Elemente und Variabilitäten markiert. Erfahrungsgemäss sind solche Elemente zwischen einem Wort und fünf bis sechs Zeilen lang.

Die hellgrau markierten Elemente sind Daten des Systems, die dunkelgrau markierten Elemente Features und die mit Boxen markierten Elemente mögliche Feature-Werte. Es gibt prinzipiell zwei Arten, wie das Dokument mit Hilfe der Pattern untersucht werden kann:

Copyright © Fraunhofer IESE 2003

Die Software ORBIS ® -Dienstplan zeichnet sich durch folgende Merkmale aus:
• hohe Güte der generierten Dienstpläne durch die Verwendung von Software nach dem neuesten Stand der Technik. Der Constraint-Solver **ConSolve ®** dient hierbei als Planungskomponente,

• Unterstützung der Arbeitszeitflexibilisierung durch die Definition individueller Arbeitszeiten und die Führung von Zeitkonten nach verschiedenen Verfahren (vorgegebene Ausgleichsfrist, Jahresarbeitszeit, ...)

• Darstellung der tatsächlich geleisteten Arbeitsstunden, Auswertung der Istdaten in einer Ausfallstatistik,

Markierte Artefakte: Daten | Features | Feature-Werte

Abbildung 3 Markierung der Elemente in Benutzerdokumentation

- Pattern für Pattern
  Der Produktfamilienanalyst konzentriert sich auf ein oder eine Gruppe von Pattern. Dabei kann er beispielsweise nach Ein- bzw. Ausgaben der Pattern (siehe Tabelle 1) gruppieren. Dies würde bedeuten, dass er z.b. alle Pattern, die Features als Ausgabe liefern gruppiert, oder alle Pattern, die Überschriften als Eingaben erhalten. Mit diesen Pattern bzw. Gruppen von Pattern arbeitet er dann den aktuellen Dokumentationsteil durch. Dabei markiert er die identifizierten Elemente. Dies ist, speziell wenn man es Pattern für Pattern durchgeht, sehr zeitaufwändig, hat jedoch den Vorteil, dass man damit alle mit dem Pattern erhebbaren Elemente mit hoher Wahrscheinlichkeit findet.
- Integriert
  Der Produktfamilienanalyst liest sich alle Pattern durch und arbeitet dann die Dokumente durch wobei er die identifizierten Elemente markiert. Dies erfordert weniger Zeitaufwand als die Dokumente mit nur wenigen Pattern und mehrmals durchzugehen, jedoch ist die Wahrscheinlichkeit, dass man damit alle mit dem Patterns erhebbaren Elemente findet ist dafür niedriger.

Beide Arten haben ihre Vor- und Nachteile, was den Zeitaufwand und die Vollständigkeit bzw. Korrektheit angeht. Es wird also immer anhand der Faktoren

- Menge der ausgewählten Pattern
- Länge und Anzahl der Dokumentationsteile und
- Erfahrung des Produktfamilienanalysten

abgewägt werden müssen, welche Art verwendet werden soll.

In den ausgewählten Textabschnitten werden dann nacheinander oder parallel alle Elemente die mit Hilfe der Pattern gefunden werden markiert. Das Ergebnis dieses Schrittes sind also gemeinsame und variable Anforderungsfragmente, repräsentiert durch markierte Elemente in den Dokumentationsteilen.

Copyright © Fraunhofer IESE 2003

3.4 Auswahl

In der Nachbereitung setzt der Produktfamilienanalyst die identifizierten Elemente zunächst zu vorläufigen Produktfamilienartefakten (z.B. Use Cases, Feature Listen, textuelle Anforderungen) zusammen. Die Information zur Zusammenstellen kann aus dem Text kommen (z.B. dadurch dass eine nummerierte Liste als Liste von Use Case Schritten identifiziert wurde) oder sich aus dem Zusammenhang der Texte oder aus einem Glossar oder aus Vorwissen ergeben. Weiterhin kann die Information, welches variable Feature in welchen Benutzungsdokumenten vorkommt, beim Erstellen einer Produkt-Feature Matrix, also einem Produktfamilienartefakt dass beschreibt, welche Features in welchen Produkten und Subdomänen des Systems vorkommen genutzt werden. Der Produktfamilienanalyst legt nun dieses vorläufige partielle Produktfamilienmodell dem Domänenexperten zur Auswahl, Änderung und zum Hinzufügen von neuen Artefakten vor. Die Attribute Korrektheit und Vollständigkeit der Extraktionspatterns geben dabei wertvolle Hinweise, wo evtl. verstärkt nach weiteren Elementen gesucht werden muss, bzw. Elemente besonders kritisch zu betrachten sind. Resultat des Ansatzes ist ein partielles Produktfamilienmodell welches dann in einem folgenden Modellierungsschritt weiter analysiert, vervollständigt und verifiziert werden kann. Die erzeugten Artefakte können direkt als Eingabe für verschiedene Produktfamilienmodellierungstechniken verwendet werden. Die identifizierten Features z.B. können als Basis einer Feature-orientierten Modellierung mit FODA/FeatuRSEB [7] oder als Basis fürs Scoping dienen, variable textuelle Anforderungen können in FAST [11] eingesetzt werden. Wie die aus der Benutzerdokumentation extrahierten Elemente zu Produktfamilien Use Cases zusammengesetzt werden können, ist in [3] beschrieben.

Der Ansatz befindet sich momentan am Fraunhofer IESE im Rahmen der Domänenanalyse mit dem PuLSE™CDA-Ansatz in der Erprobung, erste industrielle Fallstudien lieferten gute Resultate.

Copyright © Fraunhofer IESE 2003

4 Anwendungsbeispiel

Eine dieser industriellen Fallstudien wurde bei der SIEDA GmbH durchgeführt. Im Rahmen der Produktfamilienausweitung bei SIEDA sollte, aufbauend auf den existierenden Produkten ein neues Produkt mit reduzierter Funktionalität auf den Markt gebracht werden. Dazu war es als erster Schritt notwendig, die Variationen in den Features der existierenden und geplanten Produkte zu erfassen und nach Teildomänen des Produkts zu gliedern. Dazu sollte als erster Schritt eine Produkt-Feature Matrix (siehe Abschnitt 3.4) erstellt werden.

Zur Erstellung dieser Matrix wurde die knapp 30 seitige Benutzerdokumentation von 2 Produkten der Produktfamilie mit dem CaVE Extraktionsansatz analysiert, dabei wurde wie folgt vorgegangen:

- Vorbereitung
  Da die Gesamtdokumentation der beiden Produkte nur je 30 Seiten umfasste, musste diese nicht weiter unterteilt werden. Primäres Ziel der Extraktion war die Erstellung einer Produkt-Feature Matrix, zusätzlich sollten Informationen über Teildomänen und Daten gefunden werden. Deshalb wurden nur Pattern ausgewählt, die sich auf Features und Domänen beziehen. Zusätzlich wurden noch Pattern für textuelle Anforderungen die Daten betreffen können ausgewählt (z.B. „Sätze in denen das Teilwort „-daten“ vorkommt enthalten Datenrequirements“) und natürlich die Variabilitätspattern.
- Analyse
  Mit Hilfe der ausgewählten Pattern wurden die Dokumentationen parallel analysiert und Elemente markiert. Dabei wurde das in Abschnitt 3.3 beschriebene integrierte Vorgehen gewählt, da nur relativ wenige Pattern ausgewählt wurden und der Produktfamilienanalyst die Pattern sehr gut kannte. Einen Ausschnitt aus der so markierten Dokumentation zeigt Abbildung 3. Bei der Analyse konnten etwa 130 Features verschiedener Granularität, etwa 12 Subdomänen des Systems und etwa 20 Datenelemente identifiziert werden. Da die beiden analysierten Systeme sehr homogen sind, konnten nur drei Variabilitäten gefunden werden.
- Auswahl
  Die identifizierten Features und Subdomänen wurden in einer vorläufigen Produkt-Feature-Matrix zusammengestellt (Ausschnitt siehe Abbildung 4). Diese vorläufige Matrix wurde in einem knapp 2-stündigen Workshop den Domänenexperten präsentiert, gemeinsam korrigiert und mit den Featurewerten für das geplante Produkt aufgefüllt. Es wurde also schon im Auswahlschritt ein fast vollständiges Produktfamilienartefakt erzeugt.

Copyright © Fraunhofer IESE 2003

Domäne	Sub-domäne	Feature	Werte	Orbis 1	Orbis2	Orbis neu
Systemver-waltung	allgemein		X	X		
		Unterstützung ver-schiedener Diensttypen Frühdienst, Tagesdienst, Spätdienst, Nachtdienst....	Alle	Alle	Alle	
		Bereitschaftsdienste	X	X		
	Konfiguration		X	X	X	
		Tarifgruppenbezug	X	X	N	
		Freie Fehlzeiten	X	X	X	

Abbildung 4 Ausschnitt einer extrahierten Produkt Feature Matrix

Die Verwendung der Benutzerdokumentation und des Extraktionsansatz hat sich in diesem Umfeld bewährt. Es konnte in sehr kurzer Zeit ein relativ vollständiges Bild der Systeme aus der Dokumentation der Systeme extrahiert werden. Über 80% der letztendlichen Features konnten mit Hilfe der Extraktionspattern in der Benutzerdokumentation korrekt identifiziert werden, von diesen knapp 20% fehlenden Features konnte etwa die Hälfte nicht gefunden werden da sie nicht im Text vorkamen, die andere Hälfte konnte mit Hilfe der bisherigen Pattern im Text nicht identifiziert werden. Nur etwa 5% der identifizierten Features haben sich im gemeinsamen Workshop als falsch herausgestellt, dies waren größtenteils Features die doppelt identifiziert wurden oder Features die unter einem anderen Namen schon in der Produkt Feature Matrix aufgenommen waren.

Weitere Anwendungsbeispiele und Fallstudien sind in [4] und [3] beschrieben.

Copyright © Fraunhofer IESE 2003

5 Zusammenfassung und Ausblick

Dokumentation ist eine wertvolle Quelle zur Anforderungsermittlung. Benutzerdokumentation beschreibt ein System aus Benutzersicht und kann daher gut zur Extraktion von Anforderungen an neu zu erstellende Produktfamilien verwendet werden. Existierende Ansätze zur Domänenmodellierung wie FODA [7], die Commonality Analse in FAST [11] oder der schon etwas ältere aber ausführlich beschriebene Ansatz ODM [9] bieten zwar Unterstützung bei der Analyse und Modellierung von Produktfamilienanforderungen, gehen jedoch nicht darauf ein, wie und wo die benötigten Information zu finden oder zu erheben sind. Mit Hilfe des hier vorgeschlagenen CaVE Ansatz der auf Basis von Extraktionspattern Anforderungsfragmente in Dokumentationen findet, können Anforderungen in beliebigen Notationen gefunden und in eine Produktfamilie integriert werden. Dieser dokumentationsbasierte Ansatz entlastet die Domänenexperten. Der Ansatz wurde in industriellen Fallstudien erfolgreich angewendet. Momentan wird eine Toolunterstützung des Ansatz realisiert, die mit Hilfe von Information Retrieval Methoden [1] relevante Dokumentteile findet und eine Suche mit Hilfe der Extraktionspattern durchführt.

Copyright © Fraunhofer IESE 2003

6 Referenzen

1. Ricardo Baeza-Yates, Berthier Ribeiro-Neto. "Modern information retrieval". Addison-Wesley, 1999
2. Joachim Bayer, Dirk Muthig, and Tanya Widen. „Customizable Domain Analysis". In Proceedings of the First International Symposium on Generative and Component-Based Software Engineering (GCSE '99), Erfurt, Germany, September 1999
3. Alessandro Fantechi, Stefania Gnesi, Isabel John, Giuseppe Lami, Jörg Dörr. "Elicitation of Use Cases for Product Lines" In Proceedings of the Fifth International Workshop on Product Family Engineering, PFE-5, Siena, Italy, November 4-6, 2003.
4. Isabel John, Jörg Dörr, "Extracting Product Line Model Elements from User Documentation", Fraunhofer IESE Report 2003, erscheint Nov. 2003.
5. Isabel John, Jörg Dörr, "Elicitation of Requirements from User Documentation", Proceedings of REFSQ '03, Klagenfurt, June 2003
6. Isabel John, Dirk Muthig "The State of the Practice of Systematic Software Development and Product Line Development in Germany" IESE Report, erscheint December 2003
7. Kyo Chul Kang, Jaejoon Lee, Patrick Donohoe. "Feature-Oriented Product Line Engineering". IEEE Software, 19(4):58--65, Juli/August 2002
8. Klaus-U.Rautenberg, Oldrich Sova: „Dokumentation computergestützter Informationssysteme". Verlag Saur, K.G., Müchen, 1983.
9. Software Technology for Adaptable, Reliable Systems (STARS). "Organization Domain Modeling (ODM) Guidebook", Version 2.0, June 1996
10. Ian Sommerville. "Software Engineering". Addison-Wesley, 2001.
11. David M. Weiss and Chi Tau Robert Lai. "Software Product-Line Engineering: A Family-Based Software Development Process". Addison-Wesley, 1999

Copyright © Fraunhofer IESE 2003

Dokumenten Information

Titel:	Extraktion von Produktfamilien-anforderungen aus Benutzerdo-kumentation
Datum:	Oktober 2003
Report:	IESE-111.03/D
Status:	Final
Klassifikation:	Öffentlich

Copyright 2003, Fraunhofer IESE.
Alle Rechte vorbehalten. Diese Veröffentlichung darf für kommerzielle Zwecke ohne vorherige schriftliche Erlaubnis des Herausgebers in keiner Weise, auch nicht auszugsweise, insbesondere elektronisch oder mechanisch, als Fotokopie oder als Aufnahme oder sonstwie vervielfältigt, gespeichert oder übertragen werden. Eine schriftliche Genehmigung ist nicht erforderlich für die Vervielfältigung oder Verteilung der Veröffentlichung von bzw. an Personen zu privaten Zwecken.