ComGeneration: die Dienstebeschreibung als Basis für automatisierte Tests

Patrick Wacht, Thomas Eichelmann,
Armin Lehmann, Ulrich Trick
Fachhochschule Frankfurt/M.
University of Applied Sciences,
Nibelungenplatz 1, 60318 Frankfurt/M., Germany
E-Mail: {wacht, eichelmann, lehmann, trick}@e-technik.org;

Rolf Lasch, Marten Fischer, Ralf Tönjes
Hochschule Osnabrück
University of Applied Sciences,
Albrechtstr. 30, 49076 Osnabrück, Germany
E-Mail: {r.lasch, m.fischer, r.toenjes}@hs-osnabrueck.de;

Das dieser Publikation zugrunde liegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung (BMBF) unter dem Förderkennzeichen 1724B09 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

Kurzfassung

Um sicherzustellen, dass ein Mehrwertdienst gemäß den Anforderungen eines Kunden realisiert wurde, ist die Durchführung von funktionalen Tests unerlässlich, welche aus einer vorliegenden Dienstebeschreibung bzw. Spezifikation abgeleitet werden. In diesem Aufsatz wird eine neue Art der Dienstebeschreibung vorgestellt, die nicht den Kontakt zum Kunden verliert, diesen mit einbindet und eine solide Grundlage für die schnelle, kostengünstige und qualitativ hochwertige Dienst- und Testentwicklung gewährleistet. Das Vorgehen beim Erstellen einer solchen Dienstebeschreibung wird vorgestellt und anhand eines konkreten Beispieldienstes gezeigt.

1 Einleitung

Die Nachfrage nach neuen Diensten, speziell Mehrwertdiensten, wächst ständig, daher besteht ein großes Interesse seitens der Netzbetreiber und Diensteanbieter, kostenoptimierte Lösungen zu finden, um neue Dienste einfach und schnell bereitstellen zu können. Durch das IMS (IP Multimedia Subsystem) werden neue Möglichkeiten der einfacheren und schnelleren Dienstebereitstellung gegeben, die zu neuen Einnahmequellen neben der normalen Telefonie führen. Um die Nachfrage der Kunden zu befriedigen, müssen neu entwickelte Dienste nicht nur bereitgestellt, sondern auch getestet werden, bevor sie in den Wirkbetrieb übergehen. Nur so kann die Grundvoraussetzung für die Zufriedenheit der Kunden garantiert werden. Das Testen von Diensten wird durch die steigende Nachfrage und Komplexität der neu entwickelten Dienste zunehmend an Bedeutung gewinnen.
Im BMBF-Projekt ComGeneration (Test-gesteuerte Evolution und automatisierte Bereitstellung von Kommunikationsdiensten) [1] ist daher eine durchgängige Lösung erarbeitet worden, die beginnend mit der Dienstebeschreibung bis hin zum ausführbaren Test reicht.
Im nächsten Kapitel wird der in ComGeneration entwickelte Ansatz näher erläutert, der auf einer wohl definierten Dienstebeschreibung fusst. Diese Dienstebeschreibung wird in Kapitel 3 eingehend beleuchtet und in Kapitel 4 anhand eines Beispiels veranschaulicht. Die Zusammenfassung und geplantes weiteres Vorgehen bilden den Abschluss dieses Aufsatzes in Kapitel 5.

2 ComGeneration-Ansatz

Wie in Bild 1 dargestellt, wird in diesem Ansatz Wert darauf gelegt, dass zu Beginn eine definierte Dienstebeschreibung vorliegt, aus der sowohl die Dienstentwicklung als auch die Testentwicklung hervorgehen.

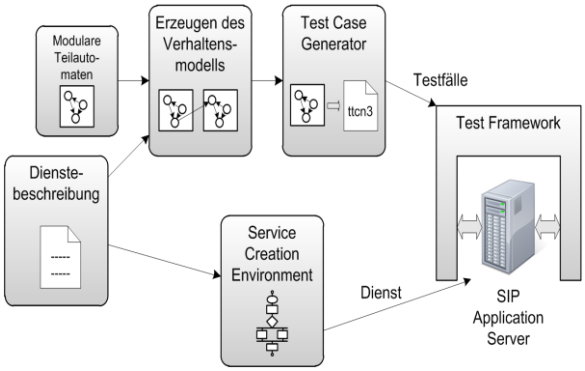

Bild 1 ComGeneration Framework

Für die Dienstentwicklung kann theoretisch jedes existierende Service Creation Environment (SCE) eingesetzt werden. Ebenfalls denkbar wäre eine manuelle Implementierung durch einen Dienstentwickler. Im Gegensatz dazu ist der Ablauf der Testentwicklung durch einen Testentwickler im ComGeneration-Ansatz wohldefiniert [2; 3]. Dieser muss zunächst die testspezifischen Informationen aus der ihm vorliegenden Dienstebeschreibung herausfiltern. Danach selektiert er aus einem Repository an vorgefertigten wiederverwendbaren modularen Teilautomaten die für den Dienst relevanten Bausteine. Diese modularen Teilautomaten beschreiben typische dienstspezifische Abläufe, z.B. bestimmte Sequenzen von Nachrichten für Protokolle wie SIP (Session Initiation Protocol) oder HTTP (Hypertext Transfer Protocol). Nach der Auswahl der relevanten modularen Teilautomaten komponiert der Testentwickler diese entsprechend den Angaben aus der Dienstebeschreibung und erhält in der Folge das sogenannte Verhaltensmodell, bei dem es sich um einen komplexen Zustandsautomaten handelt. Alle in diesem Zustandsautomaten möglichen zu durchlaufenden Zustände und Zustandsübergänge inklusive der diese hervorrufenden Events (Eingangssignale) und resultierenden Actions (Ausgangssignale) vom Startzustand bis zum Endzustand des Zustandsautomaten beschreiben die zu testenden Nachrichtenabläufe. Diese Nachrichtenabläufe stellen das mögliche Verhalten aus der Sicht des Dienstes dar. Um nun mittels des Test Case Generators aus dem Verhaltensmodell TTCN-3-Testfälle generieren zu können, welche anschließend in einem Test Framework auf das "System under Test", dem deployten Dienst, ausgeführt werden können, wird die Sicht invertiert. So handelt es sich bei den Eingangssignalen für den Dienst im Verhaltensmodell um Nachrichten, die von Testseite bzw. von den Testkomponenten gesendet werden müssen. Im Gegensatz dazu handelt es sich bei den Ausgangssignalen vom Dienst um Nachrichten, welche von der Seite des Tests empfangen werden.

3 Dienstebeschreibung

Wie bereits erwähnt, ist die Grundlage für die Entwicklung eines neuen Dienstes und dessen Tests eine hinreichend genaue Dienstebeschreibung. Diese Dienstebeschreibung muss mehrere Kriterien erfüllen. Sie sollte möglichst unmissverständlich und eindeutig sein. Hierfür sollte ein eingeschränkter Wortstamm (mit Schlüsselwörtern) genutzt werden, um Doppeldeutigkeit an Begriffen zu vermeiden. Darüber hinaus muss die Dienstebeschreibung so geartet sein, dass sowohl der Test- als auch der Dienstentwickler diese verstehen und interpretieren können. Beide Entwickler sollten möglichst das gleiche Bild des Dienstes vor sich haben, um spätere Differenzen zu vermeiden. Eine weitere Anforderung an die Dienstebeschreibung ist, dass diese hinreichend genau den Dienst bzw. das Dienstverhalten beschreibt. Hierzu bedarf es speziell geschultem Personal (sogenannte Service Agents), welches zusammen mit dem Kunden eine Dienstebeschreibung erstellt.

3.1 Aufbau der Dienstebeschreibung

Die Erstellung einer Dienstebeschreibung gliedert sich wie folgt:

1. Erstellung einer textuellen Dienstebeschreibung.
2. Identifizieren der am Dienst beteiligten Rollen.
3. Spezifizieren der Anforderungen aus Sicht der Benutzer/Kunden.
4. Ergänzen der spezifizierten Anforderungen
5. Identifizieren der Kommunikationsschnittstellen zwischen dem Dienst (System) und den Nutzern.
6. Entwickeln der Tests und des Dienstes.

Die genannten Punkte 1 bis 3 werden werden durch den Service-Agenten mit dem Kunden erarbeitet. Die Schritte 4 und 5 werden darauf durch den Service-Agenten fortgeführt und Schritt 6 wird dann durch die Entwickler (Test- bzw. Dienstentwickler) durchgeführt. Bis zum Schritt 6 muss die Dienstebeschreibung allen Anforderungen genügen. Die einzelnen Schritte der Dienstebeschreibung bis hin zur Erstellung der Tests und des Dienstes werden im Folgenden näher erläutert.

3.2 Textuelle Dienstebeschreibung

In der textuellen Beschreibung wird der Dienst mit all seinen Eigenschaften aus Sicht des Benutzers beschreiben. Hier werden auch Vereinbarungen genauer definiert wie zum Beispiel die SIP URI (Uniform Resource Identifier) des Dienstes. Während der Erstellung der textuellen Beschreibung muss der Service Agent genau darauf achten, dass keine Ungenauigkeiten in der Beschreibung auftreten. Ein Beispiel hierfür wäre der Satz: „Alle weiteren Fehler werden ebenfalls übermittelt“. Dieser Satz ist unzureichend genau und muss ergänzt bzw. geändert werden in z.B.: „Bei Nichterreichbarkeit und falsch angegebener Zieladresse wird dem anfragenden Teilnehmer über Statusmeldungen der aufgetretene Fehler signalisiert“.

3.3 Identifikation der beteiligten Rollen

Aus der bereits erstellten textuellen Beschreibung des Dienstes können Rollen (z.B. SIP-Endgerät) identifiziert werden. Alle Rollen müssen aus der textuellen Beschreibung ableitbar sein. Hierfür sorgt der Service Agent, indem er mit dem Kunden die Beschreibung komplettiert. Die Rollen müssen immer aus Sicht auf das System definiert werden, wobei unter dem System der erwünschte Dienst zu verstehen ist.

3.4 Spezifikation der Anforderungen

Dieser Abschnitt der Erstellung der Dienstebeschreibung unterteilt den zu entwickelnden Test und Dienst in einzelne Bereiche, die als sogenannte Dienstpfade verstanden werden können. Unter einen Dienstpfad ist ein bestimmter Ablauf bzw. eine Variante innerhalb eines Tests bzw. eines Dienstes zu verstehen. Zum Beispiel wird auf Grund eines Ereignisses (z.B. Eintreffen einer SIP-Nachricht INVITE) ein bestimmter Pfad betreten. Beim Eintreffen eines anderen Ereignisses (andere SIP-Nachricht) wird ein anderer Pfad eingeschlagen. Die Dienstpfade dienen dazu, den Test- bzw. Dienstentwickler zu unterstützen. Beide Entwickler können sich anhand dieser Spezifikationen orientieren um den Dienst bzw. die Tests zu erstellen.
Die Spezifikation der Anforderungen besteht aus vier Elementen (in Tabelle 1 exemplarisch dargestellt).

- Rollen: Die bereits identifizierten Rollen, welche aus Sicht auf das System (Dienst) definiert wurden.
- Preconditions: Hier wird die Situation beschrieben, welche zu der vom Kunden gewünschten Folgesituation (Target) führen soll. Voraussetzung für das Erreichen des Targets ist, dass der gesamte Dienstverlauf fehlerfrei ist. Der Begriff Situation soll in diesem Zusammenhang die Summe aller Informationen bezeichnen, die dem Benutzer in seiner momentanen Rolle bezüglich seines Auftrags an das System zugänglich sind (z.B. Rufnummern oder Statusinformationen über sein Endgerät).
- Postconditions: Die Postconditions repräsentieren eine Liste aller möglichen Situationen, die nach Bearbeitung des Benutzerwunsches vorliegen können. Hier lassen sich auch die äusserlich sichtbaren Reaktionen des Systems bei gestörtem oder fehlerhaftem Ablauf des Dienstes beschreiben. Auch das Target zählt zu den Postconditions, es ist allerdings gesondert zu beachten und kenntlich zu machen.
- Prosa: Hier soll nocheinmal das Target beschrieben werden. Dazu sollen allerdings kurze und präzise Formulierungen verwendet werden. Hierdurch erhalten die Entwickler grundlegende Informationen zur Funktionalität des Dienstes.

Tabelle 1 Beispielhafte Anforderunsspezifikation

Rollen	Anrufer [a]	Zuordnung der Rollen
Pre-conditions	Anrufer kann beliebigen Teilnehmer erreichen	[a]
	Anrufer hat keine aktive Verbindung	[a]
Post-conditions	Anrufer wird nicht verbunden	[a]
	Empfänger ist unbekannt	[a]
	Empfänger ist nicht verbindungsbereit	[a]
	Anrufer wird verbunden	[a]
Prosa	Teilnehmer [a] möchte ein Gespräch mit einem anderen Teilnehmer führen	

Wie bereits erwähnt, ist die gewünschte Folgesituation besonders hervorgehoben (hier: „Anrufer wird verbunden“) und wird als Geradeausfall definiert.

3.5 Ergänzung der Anforderungen

Falls nötig definiert der Service Agent Erweiterungen für die einzelnen Anforderungen. Darüber hinaus besteht in diesem Schritt der Erstellung einer Dienstebeschreibung die Möglichkeit, auch dienstinterne Eigenschaften aufzuführen. Dies war in den vorherigen Schritten nicht möglich, da der Dienst immer als Black Box betrachtet wurde. Die folgende Liste soll ein Beispiel für ergänzende Anforderungen sein.

- SIP URIs dürfen keine Sonderzeichen enthalten (ausgenommen @).
- SIP URIs dürfen nicht mehr als 64 Zeichen lang sein.
- Der Inhalt einer Instant Message darf nicht leer sein.

3.6 Identifizierung der Kommunikationsschnittstellen

In diesem Schritt werden die Kommunikationsschnittstellen zwischen dem Dienst und den einzelnen Rollen bestimmt. Auch hierbei ist darauf zu achten, dass die Identifizierung der Schnittstellen immer aus Sicht auf den Dienst geschieht. Hierfür werden vorab definierte Schnittstellen verglichen und ausgewählt. Die Schnittstellen repräsentieren grundlegende Funktionalitäten

verschiedener Protokolle z.B. SIP UAS (User Agent Server) oder SIP UAC (User Agent Client). Das folgende kurze Beispiel soll dies verdeutlichen.
Ein SIP-Endgerät sendet eine Nachricht (Instant Message) an das System. Das SIP-Endgerät repräsentiert hierbei die Rolle, welche auf das System zugreift. Aus Sicht dieser Rolle stellt das System die inverse Funktion dar. Das bedeutet für dieses Beispiel, dass das Endgerät die Funktionalität eines SIP UACs besitzt und das System das dazu passende Gegenstück, den SIP UAS, darstellt.
Jeder der einzelnen Schritte kann dazu führen, einen oder mehrere Schritte zurückgehen zu müssen, da Teilaspekte nicht genügend genau beschrieben wurden. Hierzu kann es auch notwendig sein, den Kunden wieder mit einzubeziehen, um im Sinne des Kunden zu handeln. All diese Schritte unterstützen die Entwickler, den Test- sowie Dienstentwickler bei der Umsetzung. Ein Testentwickler kann viele notwendige Informationen daraus schließen. Er kann z.B. einzelne Testpfade aus den Postconditions zur Erstellung des Zustandautomaten-basierten Verhaltensmodells ableiten. Auch die zu nutzenden Schnittstellen (repräsentiert durch modulare Teilautomaten) und weitere wichtige Parameter (Ergänzungen der Anforderungen) zur Bestimmung der Übergänge zwischen den Zuständen im Verhaltensmodell sind bereits definiert worden und können leicht adaptiert werden.
Anhand eines Beispiels soll das Vorgehen zur Erstellung einer solchen Dienstebeschreibung im nächsten Abschnitt mittels eines Beispieldienstes detailliert erläutert werden.

4 Dienstebeschreibung für den Beispieldienst Click2IM

In dem vorigen Kapitel wurden der Aufbau und die Struktur einer Dienstebeschreibung in einer abstrakten Weise erläutert. Die Konkretisierung des Ansatzes erfolgt nun durch die Anwendung auf einen konkreten Beispieldienst mit dem Namen Click2IM (Click-to-Instant-Message).

Der erste Punkt der Dienstebeschreibung für den Click2IM-Dienst ist die grobe textuelle Beschreibung.

4.1 Textuelle Dienstbeschreibung

Es soll möglich sein, nach dem Aufruf einer Webseite zwei Eingabemasken auf der Seite auszufüllen. Eine Eingabemaske beinhaltet die Adresse bzw. Telefonnummer eines beliebigen Textanzeigegerätes bzw. Softphones, die andere soll einen beliebigen Text aufnehmen. Sobald die Eingaben in den Masken getätigt wurden, können sie durch Betätigung eines dafür vorgesehenen Buttons bestätigt werden. Falls die Zieladresse nicht erreichbar ist oder der Text nicht übermittelt werden konnte, soll dies der Person via Webseite mitgeteilt werden. Auch die erfolgreiche Versendung des Textes soll bestätigt werden.

4.2 Identifikation der am Dienst beteiligten Rollen

In der textuellen Dienstbeschreibung sind die beiden beteiligten Rollen bereits implizit erwähnt:

- Rolle 1: Webseite (Browser)
- Rolle 2: Textanzeigegerät (Softphone)

4.3 Spezifikation der Anforderungen aus der Sicht des Benutzers

In der nachfolgenden Tabelle 2 wird die für den Click2IM-Dienst relevante Anforderungsspezifikation angegeben. Die Rolle des Browsers wird nachfolgend als Initiator bezeichnet, da diese den Dienst ursprünglich konsumiert. Die Rolle des Textanzeigegerätes bzw. Softphones wird hingegen als Empfänger beschrieben.

Tabelle 2 Anforderunsspezifikation für Click2IM

Rollen	Browser [b], Softphone [s]	Zuordnung der Rollen
Preconditions	Initiator kann beliebige Zieladresse angeben	[b]
	Initiator kann Textinhalt angeben	[b]
	Initiator bestätigt Eingaben	[b]
Postconditions	1. Empfänger ist unbekannt	[b]
	2. Empfänger erhält keine Textnachricht	[b]
	3. Empfänger erhält Textnachricht	[b,s]
Prosa	Initiator möchte eine Textnachricht an ein Softphone senden.	

In der Tabelle sind ferner die relevanten Preconditions und Postconditions aufgeführt. Die Preconditions umfassen hierbei alle für den Click2IM-Dienst relevanten Schritte, um zu der geforderten Folgesituation bzw. Postcondition zu kommen, welche zuvor als Target definiert wurde. Neben dem eigentlichen Target werden noch weitere Postconditions spezifiziert, welche die sichtbaren Reaktionen des Dienstes bei gestörtem

oder fehlerhaftem Ablauf beschreiben. Sowohl dem Dienstentwickler als auch dem Testentwickler sollte klar sein, wie es zu derartigen Postconditions kommen kann. Hierfür werden die nachfolgenden ergänzenden Anforderungen angegeben.

4.4 Ergänzung der Anforderungen

Diese Anforderungsergänzungen gelten ausschließlich für die Postconditions und werden jeweils diesen zugeordnet.

a. Standardfehler werden signalisiert (z.B. Timeouts)
b. Maximale Länge der Zieladresse (128 Zeichen)
c. Sonderzeichen in der Zieladresse sind nicht zulässig (exklusive @)
d. Maximale Länge der Texteingabe (256 Zeichen)
e. Leere Texteingaben sind nicht zulässig

Der Bezug zu den in Tabelle 2 definierten Postconditions wird im nachfolgenden Bild 2 deutlich.

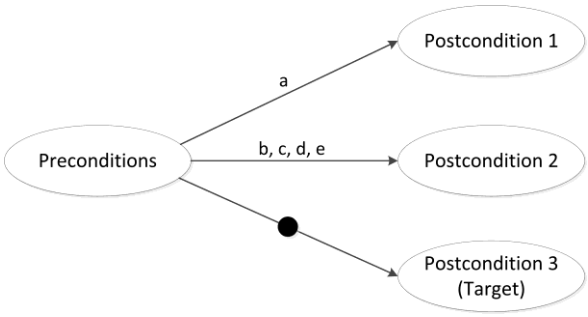

Bild 2 Zuordnung von Anforderungsergänzungen zu Postconditions

Die Postcondition 1 („Empfänger ist unbekannt“) wird hier über die Anforderungsergänzung a erreicht, gleiches gilt bei der Postcondition 2 für die Anforderungsergänzungen b, c, d und e. Wenn also diese fehlerhaften Angaben gemacht werden, wird laut Spezifikation erwartet, dass Postcondition 2 eintritt, der Empfänger sollte also keine Textnachricht erhalten. Postcondition 3 stellt das Target dar und wird hier gesondert dargestellt.

Indem man also die Anforderungsergänzungen den Postconditions zuordnet, erhält man aus der Dienstebeschreibung die relevanten Dienstpfade. Insgesamt können für den Click2IM-Dienst sechs Dienstpfade identifiziert werden, die sowohl der Dienstentwickler als auch der Testentwickler berücksichtigen müssen.

4.5 Identifizierung der Kommunikationsschnittstellen

Die Kommunikationsschnittstellen stellen jeweils das Gegenstück der klassifizierten Rollen dar. Sie repräsentieren die realen Schnittstellen des Dienstes nach außen. Für den Click2IM können die folgenden Schnittstellen identifiziert werden:

- HTTP Server
- SIP UAC (User Agent Client) initRequest

Die Kommunikationsschnittstelle HTTP Server besagt, dass die vom Browser initiierten HTTP Request-Nachrichten (z.B. HTTP POST) dienstseitig empfangen und weiterverarbeitet werden. Die zweite Kommunikationsschnittstelle SIP UAC initRequest regelt hingegen die SIP-Kommunikation zwischen dem Dienst und dem Softphone. Bei der Beschreibung des ComGeneration-Ansatzes in Kapitel 2 wurden bereits die sogenannten Teilautomaten erwähnt, welche als Bausteine zur Erzeugung eines Verhaltensmodells herangezogen werden können. Für die Kommunikationsschnittstellen wurden im Rahmen des Projektes derartige Teilautomaten definiert. Die für den Click2IM-Dienst relevanten Teilautomaten kann der Testentwickler direkt aus den in der Dienstebeschreibung definierten Kommunikationsschnittstellen ableiten und als Basis für das Verhaltensmodell verwenden.

5 Zusammenfassung und Ausblick

In dieser Veröffentlichung wird ein Ansatz zur Spezifizierung und Beschreibung von Mehrwertdiensten vorgestellt. Im Rahmen des Projekts ComGeneration wird eine durchgängige Lösung zum Testen von Mehrwertdiensten von der Dienstebeschreibung bis zum ausführbaren Test entwickelt. Der oben gezeigte Überblick über das Framework lässt bereits auf die Mächtigkeit des hier präsentierten Lösungsansatzes schließen.

Die in diesem Aufsatz vorgeschlagene Dienstebeschreibung stellt die Grundlage für die Dienstentwicklung dar, ganz gleich mit welcher SCE der Entwickler arbeitet oder ob er den Dienst sogar manuell erstellen muss. Aus der geforderten Unabhängigkeit von einer bestimmten SCE folgt eine gewisse universelle Einsetzbarkeit des hier vorgestellten Ansatzes.

Auch für den Testentwickler stellt die Dienstebeschreibung die Grundlage für die Entwicklung von Testfällen dar. Der Testentwickler ermittelt aus der Dienstebeschreibung die benötigten modularen Teilautomaten und verknüpft diese, wie in der Dienstebe-

schreibung aus den Dienstpfaden ablesbar ist, miteinander. Daraus entsteht das Verhaltensmodell des Dienstes, aus dem automatisch TTCN3-Testfälle abgeleitet werden. Ein weiterer Vorteil dieses Vorgehens ist die automatische Generierung der Testfälle. Der Testentwickler muss nur die relevanten Teilautomaten auswählen und miteinander verknüpfen. Je umfangreicher und vollständiger der Pool an vordefinierten Teilautomaten ist, desto geringer fällt der Aufwand für den Testentwickler aus. Modulare Teilautomaten können dabei auch vom Testentwickler aus einfacheren Teilautomaten komponiert werden, hier entfaltet dieser Ansatz sein größtes Potential.

Dieser Aufsatz fokussiert hauptsächlich auf die Dienstebeschreibung. Der Aufbau, die Regeln und Anforderungen für die Erstellung dieser werden definiert und erläutert. Das Vorgehen beim Erstellen einer Dienstebeschreibung wird vorgestellt und anhand eines Beispiels gezeigt.

Für das hier vorgestellte Framework wird ein Prototyp implementiert. Der momentane Stand der Framework-Entwicklung erlaubt dem Testentwickler bereits die Erstellung der Statemachine für das Verhaltensmodell. Modulare Teilautomaten werden im Moment noch nicht unterstützt. Die automatische Generierung von TTCN-3-Testfällen aus dem Verhaltensmodell konnte aber bereits für einfache Dienste ähnlich dem hier vorgestellten durchgeführt werden. Der Test Case Generator kann das Verhaltensmodell analysieren und findet die möglichen Testfälle aus dem Verhaltensmodell heraus. Für jeden gefundenen Testfall generiert der Test Case Generator den entsprechenden TTCN-3 Code.

Die Dienstbeschreibung muss hinreichend genau sein. Eine zu genaue Beschreibung des Dienstes führt zu umfangreichen Dokumenten, andererseits kann eine ungenaue Dienstbeschreibung zu Fehlinterpretationen durch den Dienstentwickler oder den Testentwickler führen. Durch die bisherige Formalisierung werden diese Schwierigkeiten weitgehend vermieden. Dennoch soll in der nächsten Entwicklungsstufe ein Meilenstein- und Versionskonzept in den ComGeneration-Ansatz aufgenommen werden.

An der Integration der modularen Teilautomaten wird im Moment gearbeitet. Der Testentwickler kann dann direkt aus der Dienstbeschreibung die benötigten Teilautomaten erkennt, diese im Framework auswählen und zu einem Verhaltensmodell zusammenfügen. Damit wäre das Ziel einer durchgängigen Lösung erreicht, die nebenbei noch viele reizvolle Vorteile bietet.

6 Literatur

[1] http://www.ecs.fh-osnabrueck.de/27619.html

[2] Wacht, P.; Eichelmann, T.; Lehmann, A.; Trick, U.: "A New Approach to Design Graphically Functional Tests for Communication Services". Fourth IFIP International Conference on New Technologies, Mobility and Security, Paris, Frankreich, 2011

[3] Wacht, P.; Lehmann, A.; Eichelmann, T.; Fuhrmann, W.; Trick, U.; Ghita, B.: "Integration of Model-Based Functional Testing Procedures within a Creation Environment for Value Added Services", Sixth Collaborative Research Symposium on Security, E-learning, Internet and Networking (SEIN 2010), Plymouth, United Kingdom, 2010