Attempto – Englisch als (formale) Spezifikationssprache

Norbert E. Fuchs, Uta Schwertel, Rolf Schwitter
Institut für Informatik
Universität Zürich
{fuchs, uschwert, schwitter}@ifi.unizh.ch
http://www.ifi.unizh.ch/staff/fuchs.html

1 Zusammenfassung

An Spezifikationssprachen werden viele – zum Teil sogar widersprüchliche – Anforderungen gestellt. Einerseits sollen die Konzepte eines Anwendungsbereiches einfach und unmittelbar dargestellt werden können, was für *informelle* Spezifikationen in natürlicher Sprache und in graphischen Notationen spricht, andererseits sollen *formale* Spezifikationen die Verifikation und die Validierung erleichtern und den Ausgangspunkt für eine möglichst automatische Programmentwicklung bilden. Das *Attempto* Spezifikationssystem überwindet diesen Widerspruch, indem es zwei semantisch äquivalente Darstellungen einer Spezifikation zur Verfügung stellt – eine in kontrollierter natürlicher Sprache und eine in Prädikatenlogik.

In *Attempto Controlled English (ACE)* – einer Teilmenge der englischen Sprache mit eingeschränkter Grammatik und problemspezifischem Vokabular – können Anwendungsspezialisten Spezifikationen direkt in den Konzepten des Anwendungsbereiches entwickeln [Fuchs & Schwitter]. ACE ist eine mächtige Sprache, in der modellorientierte und eigenschaftsorientierte Spezifikationen eindeutig und klar geschrieben werden können. Die Sprache ACE ist gleichzeitig so einfach, dass sie effizient von einem Computer verarbeitet werden kann.

Um das Erlernen und den Gebrauch der Sprache zu erleichtern, beruht ACE auf einer kleinen Anzahl von Prinzipien. Hier sind einige dieser Prinzipien: einfache Sätze werden durch eine kleine Anzahl von Konstruktoren (z.B. Negation, `if-then`, Konjunktion, Quantifizierung) zu komplexen Sätzen kombiniert; Verben werden nur in der dritten Person Präsens und nur aktiv gebraucht; es gibt keine modalen Verben (z.B. `can`, `may`).

Das interaktive *Attempto Spezifikationssystem* übersetzt ACE Spezifikationen eindeutig in Diskursrepräsentationsstrukturen – eine syntaktische Variante der Prädikatenlogik. Bei der Übersetzung werden Ellipsen und anaphorische Bezüge nach vorgegebenen Prinzipien aufgelöst. Gleichzeitig wird automatisch eine Paraphrase in ACE erzeugt, die den Benutzern zeigt, wie das Attempto System die Eingabe interpretiert hat. In einem zweiten Schritt können Diskursrepräsentationsstrukturen in Prolog übersetzt werden.

Die in Logik übersetzte Spezifikation stellt den spezifizierten Sachverhalt formal dar. Man kann sich über diesen Sachverhalt orientieren, indem man einfache Fragen in ACE stellt, die durch logische Inferenz beantwortet werden. Die übersetzte Spezifikation kann ausgeführt und die Ausführung in ACE beobachtet und überprüft werden. Das führt dazu, dass Benutzer die Spezifikation in anwendungsspezifischen Konzepten validieren können.

Das Attempto System stellt den Benutzern weitere Werkzeuge zur Verfügung, z.B. einen lexikalischen Editor für den inkrementellen Aufbau des Lexikons und einen Spelling Checker, der das Schreiben korrekter Spezifikationen unterstützt.

In ACE wurden bisher ein Geldautomat und eine kleine Datenbank für eine Bibliothek spezifiziert.

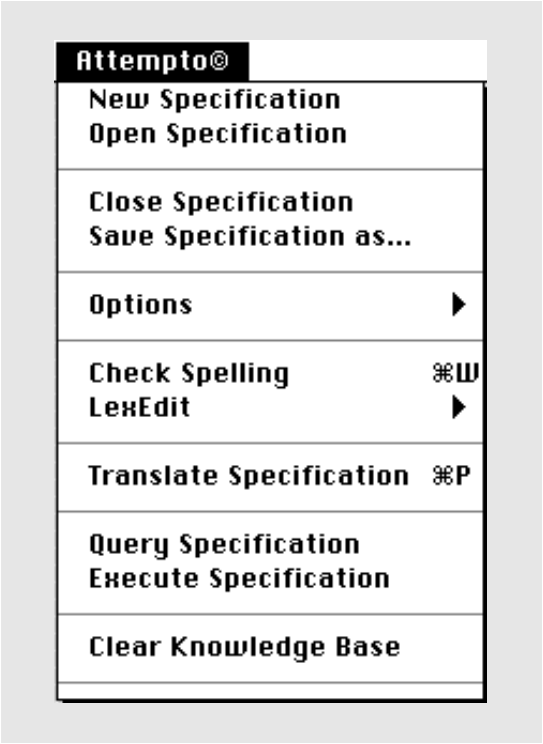
2 Beschreibung des Attempto Systems

Das Attempto Spezifikationssystem zeichnet sich durch eine einfache Handhabung aus. Die verschiedenen Funktionen, die den Benutzern beim Schreiben und Validieren einer ACE Spezifikation zur Verfügung stehen, werden hier kurz vorgestellt. Ausserdem wird die Arbeitsweise des Systems schrittweise an einem kleinen Ausschnitt der ACE Spezifikation für einen einfachen Geldautomaten verdeutlicht.

2.1 Die Funktionalität des Attempto Systems

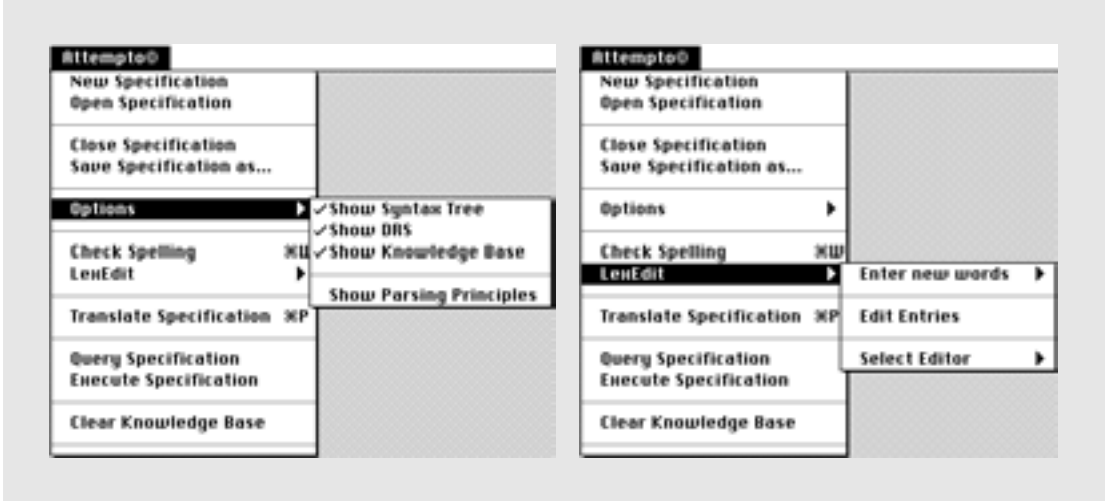
Die vorliegende Version des Attempto Systems ist in LPA MacProlog32 implementiert.

Nach dem Starten von Attempto steht Benutzern die gesamte Funktionalität des Systems auf einen Blick im **Attempto** Menüfenster zur Verfügung.



Mit dem Befehl **New Specification** wird ein leeres **Untitled Specification Window** erzeugt, das auf den gewünschten Dokumentnamen umbenannt und anschliessend mit dem Befehl **Save Specification as**... gesichert werden kann. Der Befehl **Open Specification** aktiviert ein Dialogfenster, aus welchem der Anwendungsspezialist eine bereits existierende ACE Spezifikation auswählen und laden kann. Zum Schliessen einer aktiven Spezifikation wird der Befehl **Close Specification** verwendet. Dieser Befehl schliesst nicht nur das Spezifikationsfenster, sondern löscht auch die aktuelle Wissensbasis. Mit dem Befehl **Check Spelling** kann der ACE Spezifikationstext im aktiven Spezifikationsfenster vor der Übersetzung auf unbekannte Wörter überprüft werden. Der Anwendungsspezialist lässt die ACE Spezifikation mit dem Befehl **Translate Specification** übersetzen. Um die übersetzte Spezifikation durch Fragen zu validieren, wählt er den Befehl **Query Specification** aus, welcher ein Dialogfenster für ACE Fragesätze erzeugt. Um die übersetzte Spezifikation durch Ausführung zu validieren, ruft der Anwendungsspezialist den Befehl **Execute Specification** auf. Der Befehl **Clear Knowledge Base** löscht die Wissensbasis.

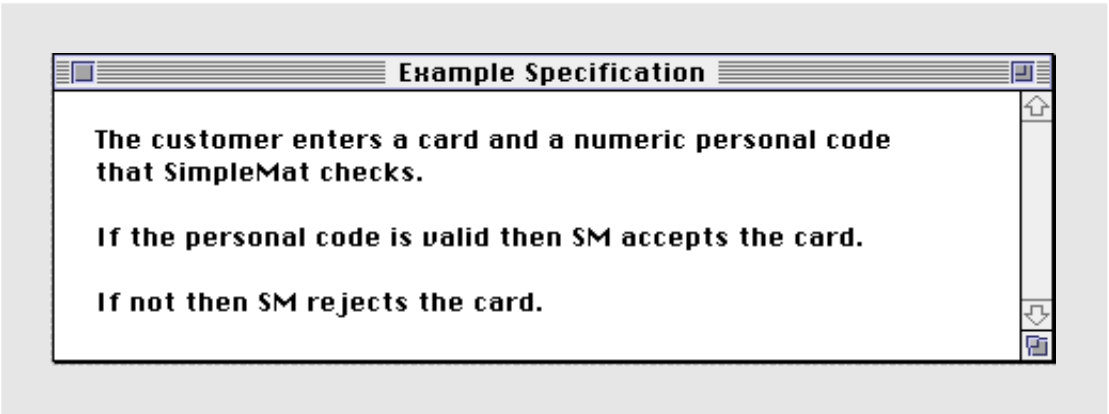
Das **Attempto** Menüfenster enthält zwei Befehle, die in Untermenüs verzweigen. Der Menübefehl **Options** verzweigt zu Befehlen, die optional den aktuellen Syntaxbaum, die Diskursrepräsentationsstruktur und den Inhalt der Prolog Wissensbasis anzeigen, oder die die Parsingprinzipien im paraphrasierten Text einblenden.



Der Menübefehl **LexEdit** verschafft Zugang zum lexikalischen Editor. Der Anwendungsspezialist hat die Möglichkeit, neue Inhaltswörter während des Spezifikationsprozesses ins Lexikon einzugeben und existierende lexikalische Einträge von Inhaltswörtern zu editieren. Gegenwärtig stehen zwei Typen von lexikalischen Editoren zur Auswahl: einer für den Anwendungsspezialisten und einer für den Experten.

2.2 Verarbeitung der *SimpleMat* Spezifikation in ACE

Als Beispiel folgt ein Ausschnitt aus der ACE Spezifikation von *SimpleMat*, einem einfachen Bankautomaten. Dazu wird mit dem Befehl **New Specification** ein neues Spezifikationsfenster erzeugt. Anschliessend verfasst der Anwendungsspezialist den Spezifikationstext, wobei er den ACE Prinzipien folgt, die den Gebrauch der Sprache regeln.



Dieser Spezifikationsausschnitt enthält die folgenden ACE Konstruktionen:

- zusammengesetzte Sätze, die aus einfacheren Sätzen mit Hilfe der Konstruktoren `and`, `that`, `if-then` und `not` gebildet werden
- zusammengesetzte Wörter, z.B. `personal code`

- syntaktische Ellipsen

  `... and [the customer enters] a numeric personal code`

  `If [the personal code is not valid] then ...`
- anaphorische Bezüge von definiten Nominalphrasen auf indefinite Nominalphrasen

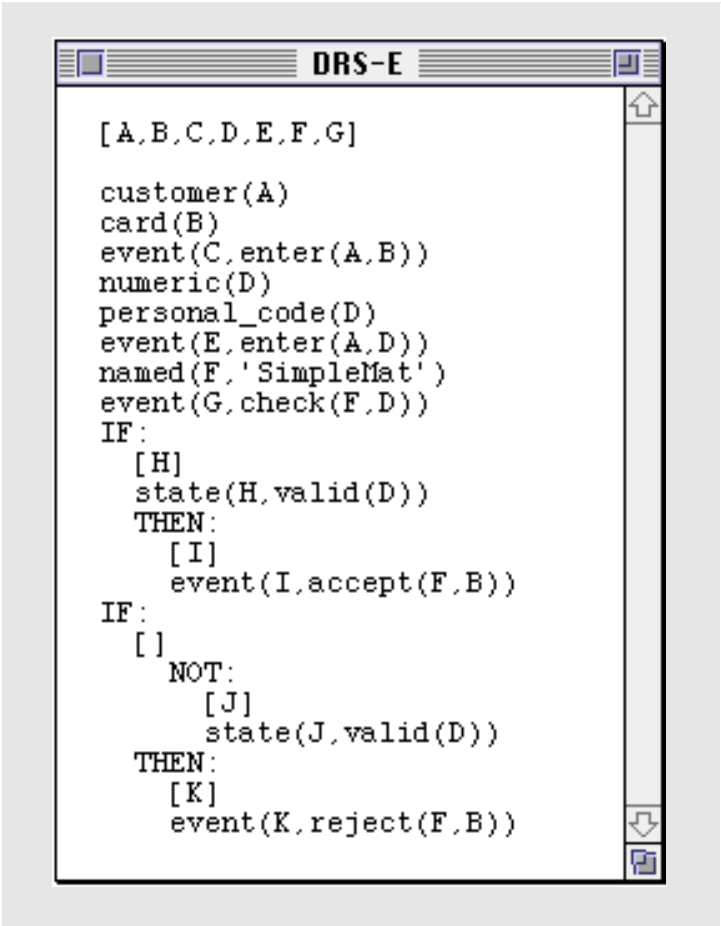
  `the card` ⇒ `a card`

  `the personal code` ⇒ `a numeric personal code`
- Abkürzungen

  `SM` steht für den Namen `SimpleMat`

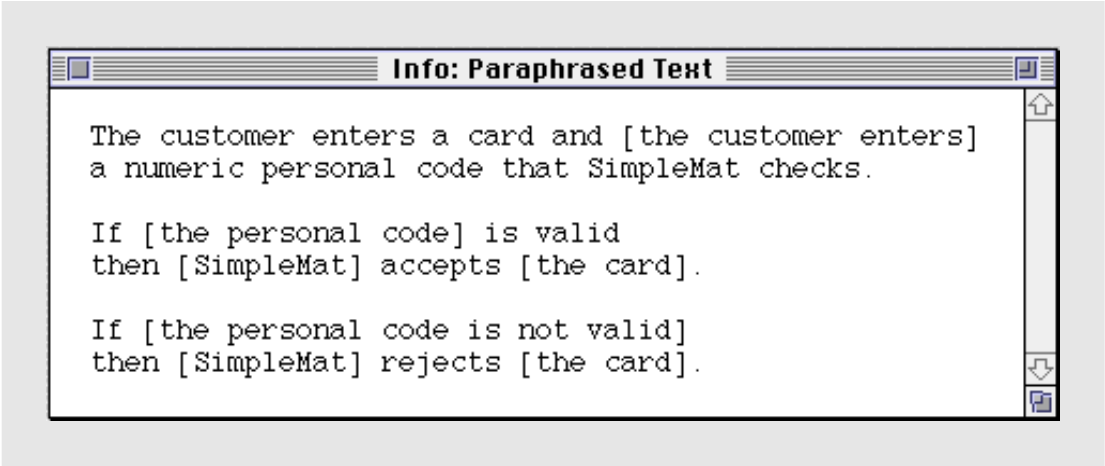
Der Anwendungsspezialist startet den Übersetzungsprozess, indem er den **Translate Specification** Befehl aufruft. Der Chart Parser nimmt den Spezifikationstext als Liste von Worten entgegen und erzeugt Syntaxbäume als grammatische Repräsentation der einzelnen Sätze, eine um Ereignisse und Zustände erweiterte Diskursrepräsentationsstruktur (DRS-E) als formale Repräsentation der spezifizierten Sachverhalte und eine Paraphrase in ACE als Feedback für die Benutzer.

Die DRS-E besteht aus einer Menge von Diskursreferenten (`[A,B,C, …]`) und einer Menge von Bedingungen für die Diskursreferenten. Anaphorische Referenzen, Ellipsen und Abkürzungen, die im Spezifikationstext auftauchen, werden in der DRS-E durch den DRS-Konstruktionsalgorithmus aufgelöst.



Die DRS-E bildet als formale Repräsentation der ACE Spezifikation den Ausgangspunkt für die Validierung und die automatische Programmentwicklung. Optional kann die DRS-E nach Prolog übersetzt werden.

Zusätzlich zu diesen internen Repräsentationen, welche normalerweise für den Anwendungsspezialisten nicht sichtbar sind, erzeugt der Chart Parser eine Paraphrase des Spezifikationstextes. Die Paraphrase zeigt den Benutzern an, welche Interpretation das Attempto System aufgrund der Sprachprinzipien und der deterministischen Parsingstrategie gewählt hat. Eckige Klammern `[]` verdeutlichen alle Ersetzungen und Interpretationen, die der Parser vorgenommen hat. Entfernt man die eckigen Klammern, entsteht ein gültiger ACE Text, der vom System wiederum verarbeitet werden kann.



Aufgrund der Paraphrase entscheidet der Anwendungsspezialist, ob er die vom System vorgelegte Interpretation akzeptiert, oder ob er den Spezifikationstext umformulieren möchte, um zur intendierten Interpretation zu gelangen. Wenn die Eingabe mehrdeutig ist, dann schlägt das System seiner deterministischen Parsingstrategie folgend immer eine – durch die Prinzipien festgelegte – Interpretation vor. Relativsätze werden beispielsweise in ACE immer an das letzterwähnte Nomen angebunden. Mit dem Befehl **Show Parsing Principles** kann sich der Anwendungsspezialist die Parsingprinzipien anzeigen lassen. Geschweifte Klammern `{}` markieren die Anbindung.

```
The customer enters a card and [the customer enters] a {numeric personal code
that SimpleMat checks}.
```

Ist der Anwendungsspezialist mit dem Ergebnis nicht zufrieden, dann muss er die Eingabe umformulieren. Wenn er einen Sachverhalt beschreiben möchte, bei dem der Geldautomat sowohl die Karte als auch den Personalcode – nacheinander – überpüft, dann erreicht er das durch den folgenden Text:

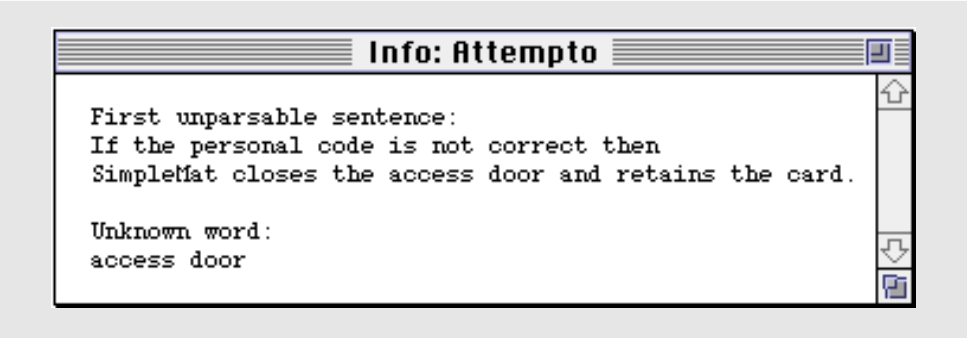
```
The customer enters a card and a numeric personal code.
SimpleMat checks the card and the personal code.
```

Wenn der Parser einen Satz nicht analysieren kann, dann erzeugt das Attempto System eine Fehlernachricht. Wenn das Inhaltswort `access door` zum Zeitpunkt der Analyse des Satzes

```
If the personal code is not correct then SimpleMat closes the access door and
retains the card.
```

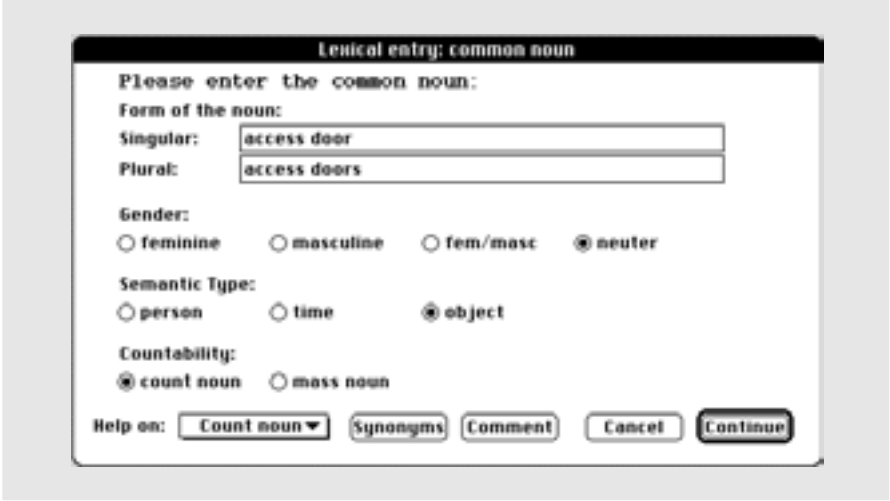
noch nicht im Lexikon verfügbar ist, schlägt die Analyse des Satzes fehl.

Das Attempto System ruft automatisch den Spelling Checker auf, der auf das unbekannte Wort `access door` stösst und die folgende Fehlernachricht ausgibt:



Da es sich beim Wort `access door` um ein unbekanntes, aber korrektes geschriebenes, Inhaltswort handelt, ruft der Anwendungsspezialist den lexikalischen Editor mit dem Befehl **Enter new word** auf und fügt das Wort zum Lexikon hinzu. Dazu braucht der Anwendungsspezialist nur schulgrammatisches Wissen.

Das Dialogfenster zeigt, wie der Anwendungsspezialist das Nomen `access door` in das linguistische Lexikon einträgt. Er muss die Wortformen im Singular und Plural eingeben und das Geschlecht des Nomen bestimmen. Weiter ist zu entscheiden, ob das Nomen zur Klasse der zählbaren Nomen oder Massennomen gehört. Zusätzlich erhält das Nomen einen semantischen Typ zugewiesen. Optional können Benutzer jedes Inhaltswort mit Synonymen und Abkürzungen assoziieren und ausserdem informelle Erkennungsregeln für den Begriff in einem Kommentarfeld formulieren. Aus dieser Information wird der komplette Lexikoneintrag abgeleitet.

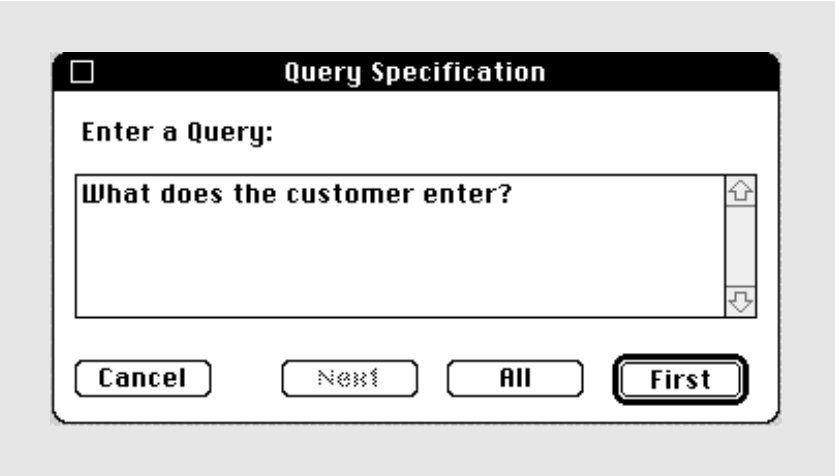


Neben der Schnittstelle für Anwendungsspezialisten gibt es eine Expertenschnittstelle, die nicht nur das Eintragen und Verändern von Inhaltswörtern zulässt, sondern dem Experten erlaubt, alle lexikalischen Einträge auf der Ebene der Merkmalstrukturen zu modifizieren.

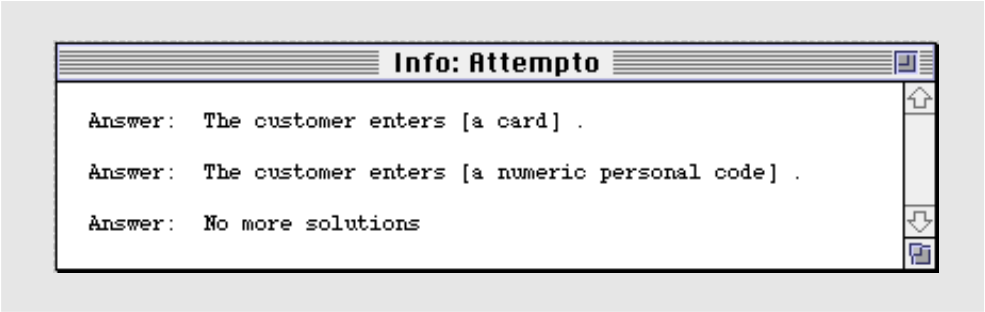
2.3 Fragebeantwortung

Der Anwendungsspezialist kann einfache Fragesätze in ACE verwenden, um sich über die spezifizierten Sachverhalte zu orientieren. Mit *Entscheidungsfragen* lässt sich feststellen, ob ein bestimmter Sachverhalt wahr oder falsch ist. Mit *Ergänzungsfragen* lassen sich bestimmte Aspekte eines Sachverhalts überprüfen. Der Befehl **Query Specification** erzeugt ein separates Dialogfenster, in welches der Anwendungsspezialist die Frage

eingibt. Hier das Beispiel einer Ergänzungsfrage:



Technisch werden Fragen in *QUERY* DRSen übersetzt und durch Inferenz beantwortet. Wenn es auf eine Frage mehr als eine Antwort gibt, dann können die Antworten nacheinander oder auf einmal generiert und angezeigt werden. Alle Antworten auf Fragen sind wiederum Sätze in ACE.

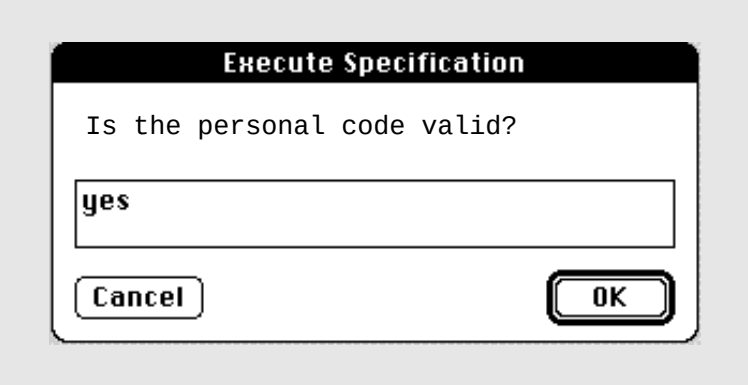


2.4 Ausführung

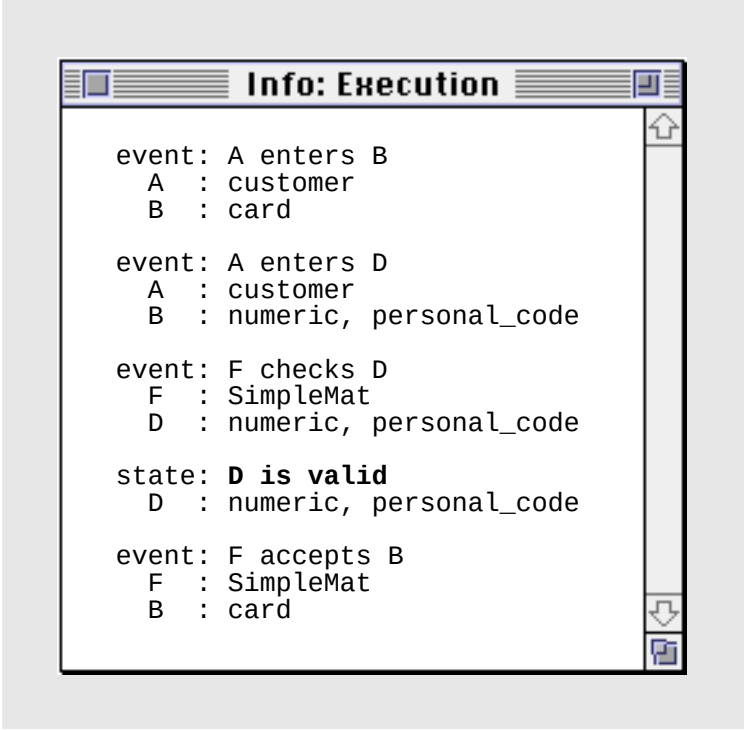
Die Diskursrepräsentationsstruktur bildet auch den Ausgangspunkt für die Validierung der Spezifikation durch Ausführung. Mit dem Befehl **Execute Specification** kann der Anwendungsspezialist die formalisierte Spezifikation durch einen Metainterpreter symbolisch ausführen lassen. Das Verhalten der Spezifikation wird dadurch in den vertrauten Begriffen des Anwendungsbereiches überprüfbar. Die Ausführung der Spezifikation verlangt jedoch zusätzliche Informationen über die Simulationsumgebung und die aktuellen Sachverhalte, welche zur reinen ACE Spezifikation hinzugefügt werden müssen.

In der Diskursrepräsentationsstruktur gibt es Bedingungen, die nicht nur wahrheitsfunktional sind, sondern bei der Ausführung der formalen Spezifikation auch Seiteneffekte verursachen. Diese Seiteneffekte müssen durch Schnittstellenprädikate definiert werden, deren Komplexität von der Simulationsumgebung abhängig ist und die möglicherweise in einer Bibliothek zur Verfügung stehen.

Ausserdem verlangt die Ausführung Information über Sachverhalte, die entweder durch die Benutzer oder durch Fakten einer Wissensbasis zur Verfügung gestellt werden kann. Beispielsweise wird den Benutzern bei der Ausführung des Spezifikationsausschnittes die untenstehende Entscheidungsfrage gestellt. Je nach Beantwortung der Frage führt das zu einem anderen Pfad durch die Spezifikation.



Die Ergebnisse der schrittweisen Ausführung der Spezifikation werden in einem Fenster angezeigt. Die Information, die durch die Benutzer geliefert wurde, ist fett gedruckt.



Der Anwendungsspezialist kann einerseits die chronologische Reihenfolge der Ereignisse und Zustände bei der Ausführung beobachten und sich andererseits selber – in den Begriffen des Anwendungsbereiches – davon überzeugen, dass die Spezifikation korrekt und vollständig ist.

2.5 Programmentwicklung

Nachdem eine Spezifikation vollständig geschrieben und validiert worden ist, kann aus der Diskursrepräsentationsstruktur – oder aus der daraus abgeleiteten Darstellung in Prolog – manuell oder automatisch ein Programm entwickelt werden.

Literatur

[Fuchs & Schwitter] N. E. Fuchs, R. Schwitter, Attempto Controlled English (ACE), Proceedings CLAW 96, First International Workshop on Controlled Language Applications, Centre for Computational Linguistics, Katholieke Universiteit Leuven, Belgium, March 1996, pp. 124-136