Zeitschrift: Bulletin des Schweizerischen Elektrotechnischen Vereins, des Verbandes Schweizerischer Elektrizitätsunternehmen = Bulletin de l'Association suisse des électriciens, de l'Association des entreprises électriques suisses

Herausgeber: Schweizerischer Elektrotechnischer Verein ; Verband Schweizerischer Elektrizitätsunternehmen

Band: 80 (1989)

Heft: 3

Artikel: KEE : eine Entwicklungswerkzeug für Expertensysteme

Autor: Trum, P.

DOI: https://doi.org/10.5169/seals-903634

Nutzungsbedingungen

Die ETH-Bibliothek ist die Anbieterin der digitalisierten Zeitschriften. Sie besitzt keine Urheberrechte an den Zeitschriften und ist nicht verantwortlich für deren Inhalte. Die Rechte liegen in der Regel bei den Herausgebern beziehungsweise den externen Rechteinhabern. Siehe Rechtliche Hinweise.

Conditions d'utilisation

L'ETH Library est le fournisseur des revues numérisées. Elle ne détient aucun droit d'auteur sur les revues et n'est pas responsable de leur contenu. En règle générale, les droits sont détenus par les éditeurs ou les détenteurs de droits externes. Voir Informations légales.

Terms of use

The ETH Library is the provider of the digitised journals. It does not own any copyrights to the journals and is not responsible for their content. The rights usually lie with the publishers or the external rights holders. See Legal notice.

Download PDF: 17.03.2025

ETH-Bibliothek Zürich, E-Periodica, https://www.e-periodica.ch

KEE – ein Entwicklungswerkzeug für Expertensysteme

P. Trum

KEE (Knowledge Engineering Environment) ist ein hybrides, sehr mächtiges und inzwischen auf verschiedenen Rechnern lauffähiges Werkzeug zur Realisierung wissensbasierter Systeme. Strukturen, Funktionen und Art der Benutzung von KEE werden beschrieben, gefolgt von einer auf Erfahrung beruhenden Bewertung des Werkzeugs.

KEE (Knowledge Engineering Environment) est un outil hybride très puissant disponible maintenant sur divers ordinateurs, permettant de réaliser des systèmes à base de connaissances. Après une description des structures, des fonctionnalités et de la méthode d'utilisation de KEE suit une appréciation guidée par l'expérience de cet outil.

Adresse des Autors

Dr. *Peter Trum*, Dipl.-Informatiker, Insiders-Gesellschaft für angewandte Künstliche Intelligenz (BDU), D-6500 Mainz.

KEE (Knowledge Engineering Environment) von Intellicorp ist ein hybrides Expertensystem-Entwicklungswerkzeug, das sowohl die objektorientierte als auch die regelorientierte Programmierung unterstützt. Darüber hinaus bietet KEE ab der Version 3.0 ein Weltenkonzept an, das - wohl als Intellicorp-Antwort auf den ART-Viewpoint-Mechanismus - eine Untersuchung alternativer Lösungen (hypothetisches Schliessen) erlaubt. Den prinzipiellen Aufbau von KEE zeigt die Graphik in Figur 1. In Tabelle I sind einige sehr wichtige Begriffe näher erklärt.

Objektorientierte Programmierung

Objekte in KEE (hier als Units bezeichnet) dienen zwei Funktionen:

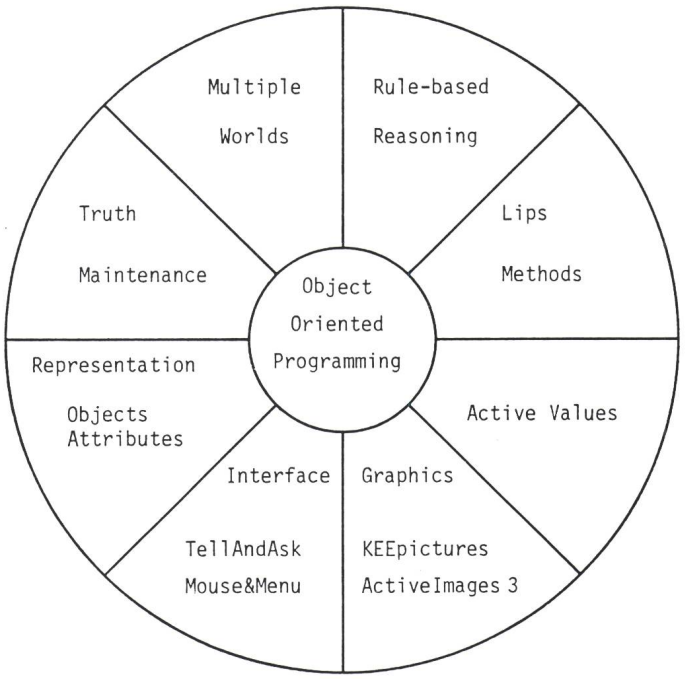
- Definition eines Modells der Wirklichkeit in Form einer Objekthierarchie (statische Struktur),
- Definition des den einzelnen Objekten zugeordneten prozeduralen Wissens in Form von Methoden und Active Values (dynamisches Verhalten).

Beschreibung der statischen Struktur

Eine Unit wird durch eine Menge von Attributen (Slots) beschrieben. Jeder Slot kann durch Facetten weiter verfeinert werden. Diese Facetten beschreiben den Vererbungsmechanismus, die Werteklasse der Slotwerte, Anzahl der Werte, die ein Slot aufnehmen kann, sowie den Slotwert selbst. Zudem können zusätzliche anwendungsspezifische Facetten für einen bestimmten Slot definiert werden. Die Objekte einer Wissensbasis stehen in einer hierarchischen Beziehung zueinander und bilden Klassen, Subklassen und Members, wobei Members Instanzen einer Klasse repräsentieren. Die Slots einer Unit werden in zwei Typen eingeteilt:

- Member Slots beschreiben Attribute von Klassen, die an alle Subklassen und Instanzen dieser Klasse vererbt werden,

Figur 1
Aufbau von KEE



Glossar

Active Values sind Funktionen, die bei einem Zugriff auf ein Objektattribut automatisch ausgefiltert werden.
Facetten beschreiben zusätzliche Eigenschaften eines Objektattributs wie z.B. Typ, Kardinalität usw.
Mit **Instanz** (Member) wird ein Objekt bezeichnet, das ein Element einer Klasse repräsentiert.
Methoden sind Funktionen, die auf Klassenebene ein bestimmtes prozedurales Verhalten beschreiben und für jede Instanz einer Klasse aufgerufen werden können.
Unter **Slot** wird ein Attribut (Eigenschaft) eines Objektes verstanden. Ein Objekt wird somit durch eine Menge von Slots beschrieben.
Unit ist eine allgemeine Bezeichnung für ein Objekt (Klassen und Instanzen) in KEE.

Tabelle I

- Own Slots beschreiben Attribute eines Objektes, die nicht weitervererbt werden.

Aufgrund der hierarchischen Struktur einer Wissensbasis erbt ein Objekt alle Member Slots seiner übergeordneten Klassen (Superklassen). Je nach Angabe des Vererbungsmechanismus können auch die entsprechenden Slotwerte vererbt werden. Der Benutzer hat somit die Möglichkeit, die Art der Vererbung von Slotwerten zu steuern.

Beschreibung des dynamischen Verhaltens

Das dynamische Verhalten von Objekten kann sowohl durch Methoden als auch durch Active Values beschrieben werden.
Methoden sind Lisp-Funktionen, die entsprechenden Methodenslots eines Objektes zugeordnet sind (d.h. der Wert dieser Slots ist eine Lisp-Funktion). Dadurch ist es möglich, auch Methoden auf Subklassen und Instanzen zu vererben. Methoden können sowohl über das Menüinterface der Entwicklungsoberfläche als auch von Programmen oder aus Regeln heraus aufgerufen werden. Auf der anderen Seite können Methoden andere Methoden aktivieren oder eine Regelauswertung initiieren. Somit sind die objektorientierte und regelorientierte Programmierung voll miteinander integriert.
Active Values realisieren in KEE das Dämonenkonzept, wie es aus anderen framebasierten Systemen bekannt ist. Active Values erlauben, beim Zugriff auf Slotwerte automatisch bestimmte Aktionen auszulösen, wobei noch die Art des Zugriffes (Lesen, Schreiben, Hinzufügen und Entfernen) unterschieden werden kann.

Regelorientierte Programmierung

Regeln werden intern ebenfalls als Units repräsentiert. Dadurch können Regeln in Gruppen (durch ihre Klassenzugehörigkeit) strukturiert und so die Menge der Regeln, die zu einem bestimmten Zeitpunkt anzuwenden sind, eingeschränkt werden (Fokussierung). Als Inferenzmechanismen stellt KEE sowohl die Rückwärtsverkettung als auch die Vorwärtsverkettung zur Verfügung, wobei diese beiden Verkettungsarten auch miteinander kombiniert werden können. Innerhalb der Regeln können Variablen benutzt werden (volles Patternmatching). Insgesamt gibt es drei Regeltypen:

- Same World Action Rules zur Beschreibung von Regeln, die innerhalb einer Welt angewandt werden,
- New World Action Rules zur Beschreibung von Regeln, die neue Welten kreieren und
- Deduction Rules zur Formulierung allgemeingültiger Abhängigkeiten (Constraints).

KEE Worlds

Die simultane Untersuchung alternativer Lösungen wird durch den KEE-Worlds-Mechanismus unterstützt. Jede Welt beschreibt dabei eine Menge von (vorgegebenen und hergeleiteten) Fakten, die jeweils eine (u.U. noch unvollständige) Lösung einer gegebenen Problemstellung repräsentieren. Welten können entweder durch Funktionen (z.B. Methoden) oder durch New World Action Rules generiert werden. Des weiteren werden die Constraints, die durch die Deduction Rules erzeugt werden, fortlaufend überprüft (Verwaltung dieser Abhängigkeiten geschieht durch ein Truth-Maintenance-System).

Benutzeroberfläche

Zur Erzeugung einer problembezogenen Endbenutzeroberfläche stellt KEE zwei Systemwissensbasen zur Verfügung:

- Die Wissensbasis Active Images enthält eine Menge von graphischen Anzeigeelementen. Diese Graphiken werden an Slots gebunden und dienen zur Anzeige und zur interaktiven Eingabe von Slotwerten (Fakten),
- die Wissensbasis KEE Pictures enthält eine Menge von Graphikgrundelementen, um z.B. Objekte bzw. Objektbeziehungen graphisch anzeigen zu können.

Die einzelnen Units innerhalb dieser Wissensbasen sind durch den Benutzer modifizierbar, so dass Änderungen und Erweiterungen dieser Graphikelemente bzw. ihres Verhaltes (Maussensitivität) jederzeit möglich sind.

Bewertung

Die folgende Bewertung von KEE basiert auf Erfahrungen von Insiders bei der Realisierung eines Expertensystems zur Konfiguration von technischen Standardmaschinen.

Generell lässt sich sagen, dass KEE ein sehr mächtiges Entwicklungswerkzeug darstellt. Zur Ausnutzung der vollen Funktionalität von KEE sind allerdings sehr gute Lisp-Kenntnisse und ein nicht zu unterschätzender Einarbeitungsaufwand nötig. Die wesentlichen Vorteile sind:

- Alle gängigen Wissensrepräsentations- und -verarbeitungsmöglichkeiten stehen dem Anwender zur Verfügung,
- mit Hilfe der Wissensbasen Active Images und KEE Pictures lässt sich sehr leicht eine dedizierte Endbenutzeroberfläche realisieren,
- die Offenheit des Systems (z.B. Modifizierung von Systemwissensbasen) erlaubt, Standardauswertungsmechanismen (z.B. Agendaverwaltung bei Vorwärtsverkettung, spezielle Vererbungsmechanismen usw. gezielt zu modifizieren.

Als wesentlicher Nachteil von KEE ist eine mangelnde Unterstützung bei der Erarbeitung der Dokumentation einer Wissensbasis zu nennen. Beim Sichern einer Wissensbasis wird lediglich die Lisp-interne Darstellung der einzelnen Objekte in einer Datei abgespeichert; eine spezielle Aufbereitung dieser Daten - ähnlich zu den Mechanismen, die in der KEE-Entwicklungsumgebung zur Verfügung stehen - findet jedoch nicht statt. Des weiteren ist anzumerken, dass die derzeitige PC-Version von KEE noch relativ instabil ist, was sich negativ auf den Entwicklungsaufwand auswirkt.

Literatur

[1] KEE software development system. Training manual. Palo Alto/California, Intellicorp, 1988.
[2] KEE software development system. Technical manuals. Palo Alto/California, Intellicorp, 1987.