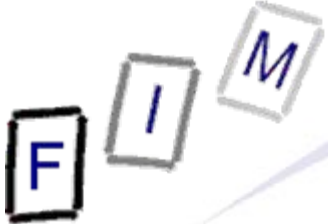


# Automatische Generierung von Übungsfällen

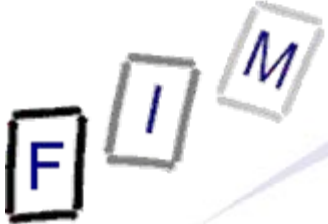
**IRIS 2008, Salzburg, 21.-23.3.2008**

Institut für Informationsverarbeitung und  
Mikroprozessortechnik (FIM)  
Johannes Kepler Universität Linz, Österreich

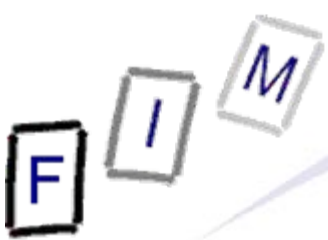
E-Mail: [sonntag@fim.uni-linz.ac.at](mailto:sonntag@fim.uni-linz.ac.at)  
<http://www.fim.uni-linz.ac.at/staff/sonntag.htm>



- E-Learning: Verhältnis Studenten/Betreuer steigt
  - Weniger persönliche Betreuung möglich
  - Aber auch schwieriger, Übungsbeispiele zu korrigieren
- Resultat: Übungen müssen automatisiert werden
- Dies ist jedoch im juristischen Bereich schwierig:
  - Erstellen von Übungsfällen ist aufwendig
  - Automatische Korrektur durch Computer (derzeit) illusorisch
- Ansatz zur Reduktion des Problems:
  - Software erstellt automatisch einen Übungsfall
- Ablauf:
  - Software erstellt Übungsfall zusammen mit Musterlösung
  - Lernende lösen den Fall selbständig
  - Lernende vergleichen ihre Lösung mit der Musterlösung

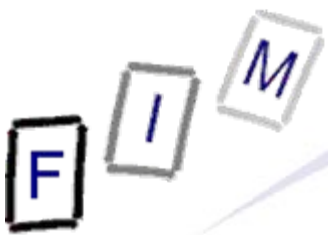


- Domainstreitigkeiten nach der UDRP
  - Uniform Domain-Name Dispute-Resolution Policy
    - » Initiator: ICANN
- Praktisch große Bedeutung
  - >2000 Fälle/Jahr
  - Muster für eine Vielzahl ähnlicher Streitschlichtungsverfahren
  - Wird dennoch kritisiert für bestimmte Schwächen, insb. die Bevorzugung von Markeninhabern in der Realität
- Einschränkung der Software: Nur materielle Probleme
  - Verfahrensrecht ist auch möglich, wird aber in den Kursen, in denen die Software eingesetzt werden soll, nicht unterrichtet!

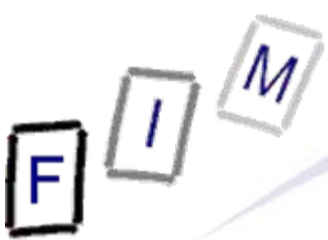


- Sowohl Fall als auch Musterlösung werden aus zufällig ausgewählten Textfragmenten zusammengesetzt
  - Jedem Angaben-Textfragment ist Lösungstext zugeordnet
- Damit nur **mögliche** Fälle generiert werden, erfolgt eine Art "Plausibilitätsprüfung":
  - Textfragmente sind entsprechend einer Ontologie klassifiziert
  - Ontologie enthält Regeln, welche Klassen nicht kombinierbar sind, d.h. nicht gleichzeitig in einem Fall vorkommen dürfen
    - » Bsp.: Domaininhaber verlangt € 100 und € 10.000 gleichzeitig
    - » Manche Textfragmente determinieren etwa zusätzlich den Domainnamen oder den Markeninhaber
  - Ergebnis: Nur rechtlich/sachlich mögliche Fälle
    - » Allerdings nicht immer auch sinnvolle Fälle!

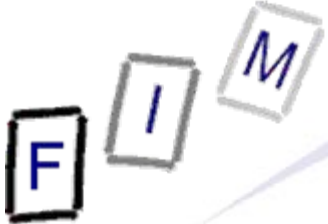




- Durch Aufbau aus Fragmenten sind "Slots" erforderlich
  - Domainname, Marke, Beschwerdeführer, ...
    - » Müssen auch grammatikalisches Geschlecht/Fall enthalten
  - Diese werden am Ende mit den korrekten Werten gefüllt
- Nachteil: Vielfache Wortwiederholungen
  - "Der Beschwerdeführer .... Und der Beschwerdeführer ....  
Allerdings hat der Beschwerdeführer ..."
- Daher Erweiterung um "Querreferenzen" bzw. andere Worte
  - Voller Name, "er"/"sie"/"dieser"/...
  - Auswahl per Zufallsgenerator mit zus. Einschränkungen
    - » Bsp: Sind Beschwerdeführer und Domaininhaber grammatikalisch männlich, so muss bei "dieser" berücksichtigt werden, wer die letzte erwähnte Person war
  - Derzeit in Ausarbeitung!

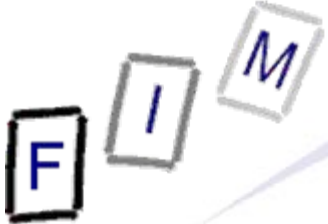


- "Berechnung" des Ergebnisses bei UDRP einfach
  - Drei positive und ein negatives Merkmal
  - Alle Merkmale sind kumulativ erforderlich ( $A \wedge B \wedge C \wedge \neg D$ )
- Ob ein Merkmal erfüllt ist oder nicht, wird durch Addition einer Bewertung aller jeweiligen Textbausteine festgestellt
  - -100: Verhindert alleine jede Erfüllung des Merkmals
  - +100: Reicht alleine zur Erfüllung des Merkmals
  - 0: "Szenario-Text" ohne Bedeutung für den Ausgang
  - -99 .. +99: Aufsummierung und Vergleich mit Schwelle +100
    - » Zwischenwerte: Nur Hinweise; benötigen "Unterstützung"
    - » +100 und -100 dürfen niemals gleichzeitig vorkommen
  - Achtung: Dies bedeutet, dass alle Textbausteine unabhängig voneinander sein müssen
    - » Verboten: X ist egal, Y ist egal, aber X+Y erfüllt das Merkmal!



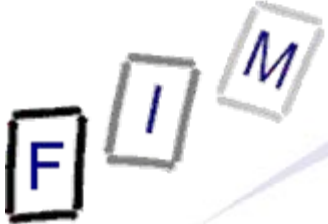
# Derzeitiger Implementierungsstand

- Fall-Generator in der Grundversion ist fertig
  - Produziert Angabe und Musterlösung als Text
    - » Keine Benutzeroberfläche!
  - Berücksichtigt alle Einschränkungen bei Zufallsgenerierung
    - » Daher nur erlaubte Fälle!
  - Querverweise/Slots werden immer konstant gefüllt
    - » Daher viele Wiederholungen
- Implementierungsdetails:
  - Programmiersprache: Java
  - Editor für Ontologie und Daten: Protégé
    - » Repräsentation der Klassen, Regel und Daten in OWL
  - Reasoner (Konsistenzprüfung): Pellet



# Geplante Entwicklungen

- Benutzeroberfläche (in Ausarbeitung)
  - Darstellung von Lösung und Musterlösung nebeneinander und Zuordnung von Satz(teilen) per Drag&Drop
  - Selbst-Bewertung jeweils mit Vollständigkeit und Korrektheit
    - » Musterlösung besteht Teilen → Damit ist auch eine Bewertung der einzelnen Fragmente und deren Klassen verbunden!
- Adaptive Generierung weiterer Fälle
  - Basierend auf schlecht bewerteten Klassen
    - » "Zufällige" Auswahl beinhaltet neue Textfragmente aus diesen
- Kooperation mit anderen Lernende
  - Archivierung von Fall + Lösungen im privaten Bereich
    - » Möglichkeit zur Veröffentlichung und Rating durch Andere
  - Gegenseitige Beurteilung mittels der Benutzeroberfläche
    - » Keine Integration in die Beurteilung der LVA!



- Automatische Generierung von Fällen ist möglich
  - Damit ist verstärktes praktisches Üben durchführbar
  - Ermöglicht Eingehen auf spezifische Schwächen
- Problembereiche:
  - Lesbarkeit: Verbesserung durch Querreferenzen
  - Sinnhaftigkeit: Der Fall soll nicht nur "erlaubt", sondern auch möglich und plausibel sein!
    - » Kann aber auch als Initiator für Diskussionen dienen!
  - Automatisierte Berechnung des Endergebnisses
    - » Hier aufgrund der einfachen Struktur und der limitierten Ergebnisalternativen einfach durchführbar
    - » Komplexere Strukturen ebenso möglich
      - Besonders schwierig: Synergieeffekte

Gefördert vom FWF: P20260-N15 ("ASCOLLA")

F I M

# Fragen?

**Vielen Dank für Ihre Aufmerksamkeit!**