Flache Verfahren zur Textgenerierung

Stephan Busemann
DFKI GmbH
Bereich Sprachtechnologie

Worum geht es?

ÜBERBLICK

- Probleme für die Entwicklung von Anwendungen in der Sprachgenerierung
- Oft Einsatz speziell entwickelter Systeme
- Definitionen: Tiefe versus flache Generierung
- Einige Konsequenzen für flache Generierung (z.B. aus TEMSIS)
- TG/2 unterstützt unterschiedliche Granularität der Grammatikmodellierung
- Benutzerorientierte Textgenerierung mit TG/2
- Vor- und Nachteile flacher Generierungsverfahren

Anwendungssysteme für NLG müssen schnell und benutzerorientiert entwickelt werden

- **Anforderungen der Anwendung**
  - Benutzer: Anforderungen erkennen und formulieren
  - Entwickler: sich mit der Domäne vertraut machen
  - Zusammenarbeit: Korpus von Beispieltexten erstellen und anpassen
- **Anforderungen an die Software**
  - Anpaßbarkeit an neue Aufgaben und Domänen
  - Skalierbarkeit (geringe Kosten der nächsten Regel)
  - Modularisierung (Interpreter, Daten, Wissen, Schnittstellen)

Hohe Entwicklungseffizienz ist mit traditionellen Generierungsverfahren nicht erreichbar

Nichttriviale Generierungssysteme erfordern hohen Anpassungsaufwand

- **Beispiele**
  - KPML (Bateman et al.), systemische Grammatiken, Entwicklungsumgebung
  - FUF/Surge (Elhadad/Robin), Funktionale Unifikationsgrammatik, Interpreter
- **Charakteristika**
  - große multi-linguale Systeme
  - detaillierte, einzelsprachspezifische Semantikkonstruktionen als Eingabe
  - decken viele sprachliche Phänomene ab (Ziel: je mehr, desto besser)
- **Anpassungsaufwand**
  - Schnittstelle zur Eingabesprache des Systems (logische Form, SPL)
  - Generierung der abgedeckten Unterscheidungen

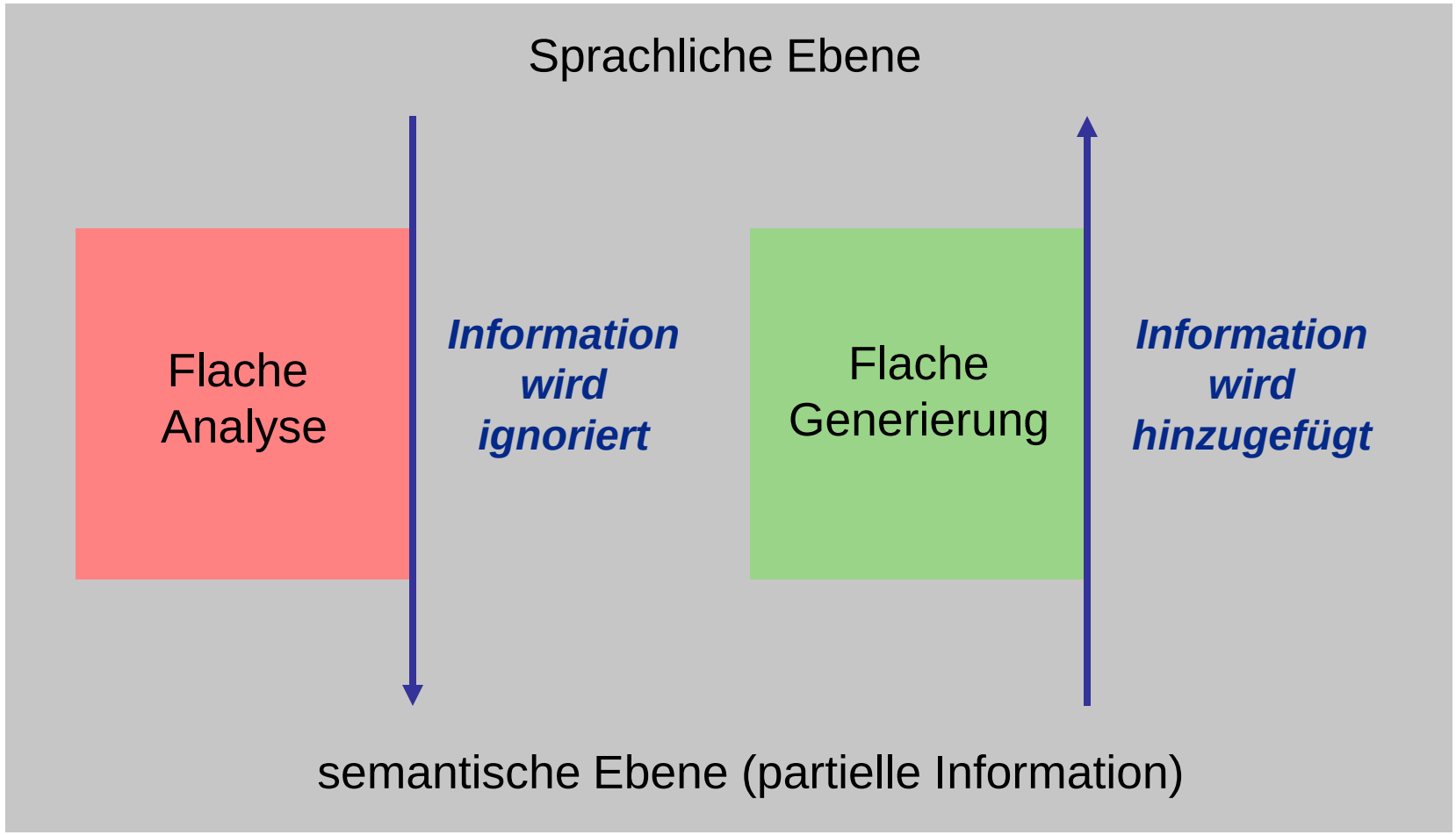
Der hervorragende Leistungsumfang allgemeiner Ressourcen kann oft nicht genutzt werden

Gegenwärtig werden neben tiefen auch flache Entwicklungsansätze verfolgt

- **Tiefe Generierung**
  - wissensbasiert (Modelle der Domäne, von Autor und Addressat, der Sprachen)
  - theoretisch motiviert, strebt generelle, wieder verwendbare Technologie an
  - Frage der allgemeinen Systemarchitektur ungelöst
- **Flache Generierung**
  - opportunistische Modellierung relevanter Aspekte der Anwendung
  - unterschiedliche Modellierungstiefen, wie von der Anwendung vorgegeben
  - Methoden oft als „Short Cuts“ bei ungelösten Fragen der tiefen Generierung

Flache Generierung kann analog zur flachen Analyse definiert werden

Flache Verfahren liefern oder verarbeiten partielle Information



Eingabestrukturen für TG/2 spezifizieren nur variable Textinhalte

```
"[(COOP wertueberschreitung)
   (TIME [(PRED season)
          (NAME [(SEASON summer)
                 (YEAR 1997)])])
   (POLLUTANT o3)
   (SITE \"Völklingen-City\")
   (DURATION [(MINUTE 60)])
   (SOURCE [(LAW-NAME bimsch)
            (THRESHOLD-TYPE infowert)])
   (EXCEEDS [(STATUS yes)
             (TIMES 1)])]"
```

En été 1997, à la station de mesure de Völklingen-City, la valeur d'information pour l'ozone pour une exposition de 60 minutes (180 µg/m³ selon le decret allemand (Bundesimmissionsschutzverordnung)) a été dépassée une fois.

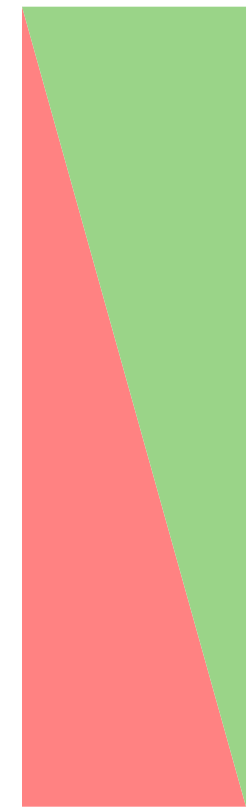
Die generierten Texte werden nicht erdichtet

KORPUSBASIERTE GRAMMATIKENTWICKLUNG

- **Benutzer stellen Beispiele für Zieltexte bereit - je mehr, desto besser**
  - Manuell erzeugte Texte, von Bereichsexperten
- **Initiale Korpusanalyse**
  - Welches Wissen wird von den Autoren verwendet?
  - Was sind die zugrundeliegenden semantischen und rhetorischen Relationen?
  - Wie lassen sich die Texte verbessern?
- **Analyse des revidierten Korpus**
  - Definition der sprachlichen Abdeckung
  - Oberflächenketten und ihre zugrundeliegenden Relationen
  - Test des revidierten Korpus (Wizard of Oz) und ggf. Iteration
- **Generalisierung zu Proto-Beispielen**
  - Grundlage für flache Grammatikentwicklung

Zwischen flachen und tiefen Verfahren besteht ein fließender Übergang

- Vorgefertigte Texte
- „Fill in the slots“
- mit flexiblen Templates
- mit Aggregierung
- mit Satzplanung
- mit Dokumentplanung



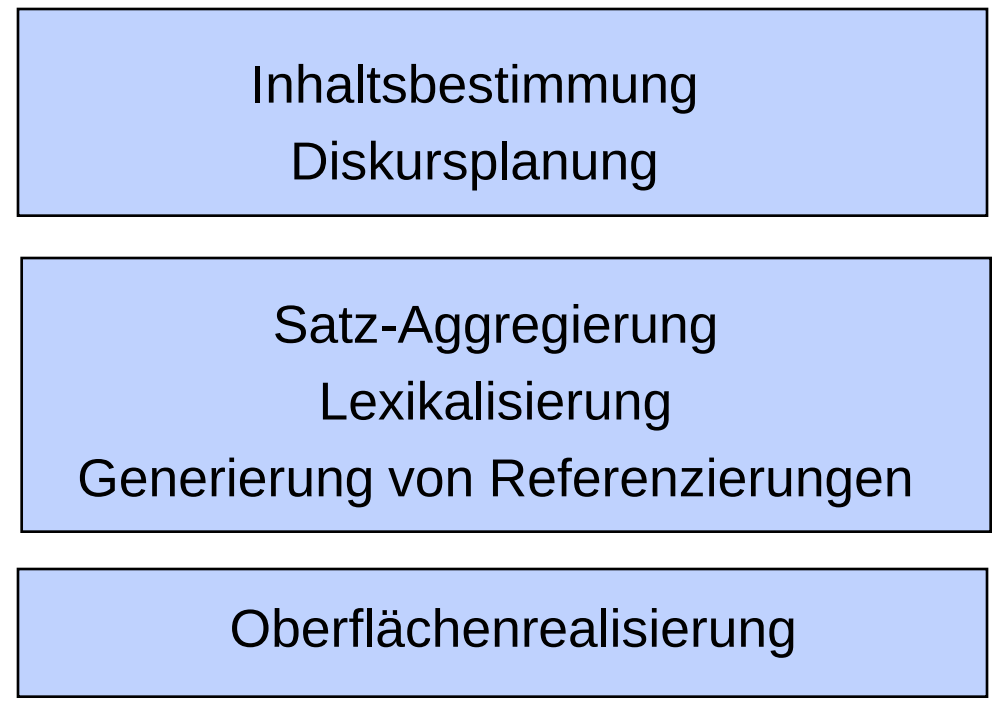
flach

tief

Flache Verfahren haben eine einfache Systemarchitektur

- **tiefe Verfahren beruhen auf Dokument- und Textplanung**
  - einzelne Komponenten können „flach“ sein
- **flache Verfahren kommen ohne Planung aus**
  - einzelne Komponenten können „tief“ sein

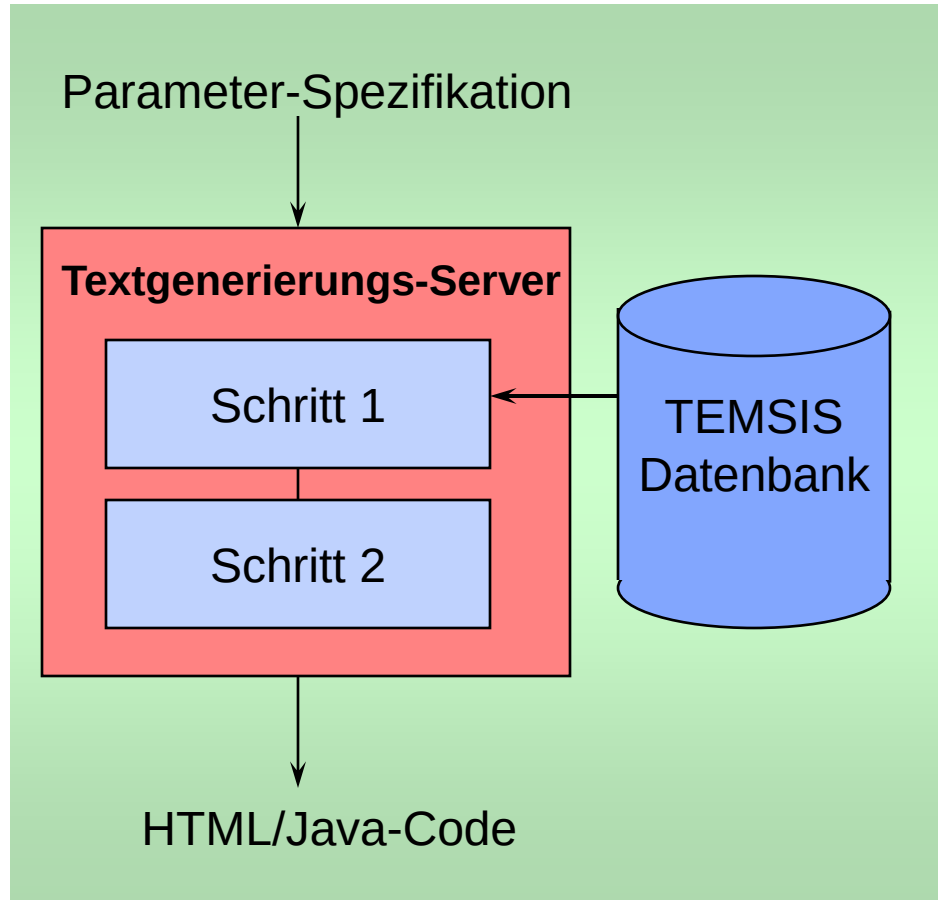
“Tiefes” Architekturmodell



Die Textgenerierung in TEMSIS erfolgt in zwei Schritten

GENERIERUNG VON LUFTGÜTEBERICHTEN

- **Parameterauswahl durch den Benutzer**
  - Sprache (D, E, F, P, J)
  - Schadstoff und Meßstation
  - Relevante Zeitperiode
- **Schritt 1: Aufbau eines Textschemas**
  - Anfrage an die Datenbank
  - Komposition der Berichtsstruktur
  - Textkohärenz
- **Schritt 2: Verbalisierung mit TG/2**
  - Auswahl von Satzmustern
  - Wörter, Phrasen, Grammatik
- **HTML-Nachverarbeitung**



Flache Generierungssysteme sind oft zu unflexibel

- **Beispiele**
  - Ana (Kukich), Börsenberichte
  - IDAS (Reiter et al), technische Anweisungen
- **Charakteristika**
  - kleine bis mittlere Abdeckung, monolingual
  - gehen von außersprachlichen Eingabedaten aus (z.B. Statistiken)
  - Techniken auf Domäne und Kommunikationssituation abgestimmt
- **Mangelnde Flexibilität**
  - vorgefertigte Texte mit eingebetteten Referenzierungen (z.B. *Carefully slide [x] out along its guide*)
  - Kasusrahmen mit textuellen Slotfüllern (z.B. *(manner: „gently“)*)

Flache Technologien müssen tiefer gelegt werden!

Die Granularität der Modellierung sollte von der Anwendung bestimmt werden

- **Nichttriviale Domänen sind inhomogen hinsichtlich der erforderlichen Modellierungstiefe**
- **Tiefe Verfahren streben hohe Homogenität der Modellierung an**
- **Flache Verfahren orientieren sich überwiegend an der Anwendung**

	Feine Modellierung	***Grobe Modellierung***	***Keine Modellierung***
Luftgüteberichte	Zeitintervalle	Zustände, Aktionen	Autor, Adressat Nominalphrasen
Termin-vereinbarung	Zeitausdrücke Autor, Adressat	Nominalphrasen	Pronomen 3. Prs.

TG/2-Grammatiken integrieren vorgefertigte Texte, Templates und kontextfreie Regeln

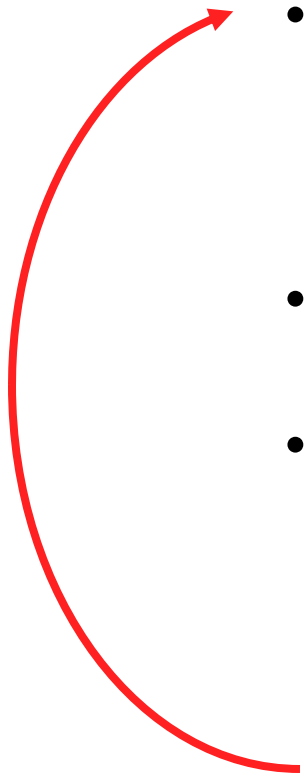
```
My category is DECL.
IF input COOP is 'threshold-passing
   AND input LAW-NAME is specified
THEN apply PPtime from input TIME
     apply THTYPE from *SELF*
     utter "("
     apply LAW from input LAW-NAME
     utter ") "
     apply EXCEEDS from input EXCEEDS
     utter "."
WHERE THTYPE AND EXCEEDS agree in GENDER
```

En été 1997
la valeur limite autorisée
(
selon le decret ...
)
a été dépassée une fois
.

```
My category is THTYPE.
IF no input THRESHOLD-TYPE is specified
THEN utter "la valeur limite autoris&e2e "
WHERE THTYPE has value 'fem for GENDER
```

Der Interpreter basiert auf dem kontextfreien Rückgrat von TG/2-Grammatiken

DREISTUFIGER AUSWERTUNGSZYKLUS



- **Matching**
  - Wähle alle Regeln mit der aktuellen Kategorie aus
  - Führe für jede die Tests über der Eingabestruktur durch (IF-Teil)
  - Füge diejenigen, die ihre Tests erfüllen, der *Konfliktmenge* hinzu
- **Konfliktlösung**
  - Wähle ein Element der Konfliktmenge aus
- **Feuern**
  - Führe Updates dynamischer Wissensquellen durch (sofern erforderlich)
  - Werte die Constraints der Regel aus (WHERE-Teil)
  - Für jedes Element des THEN-Teils der Regel lies die neue Kategorie und bestimme die neue Eingabestruktur durch Auswertung der assoziierten Zugriffsfunktion

Explizite Konfliktlösung in TG/2 basiert auf benutzerdefinierten Parametern

- **Adressaten der Luftgüteberichte**
  - Umweltverwaltung, gewöhnlich Experten
  - Öffentlichkeit, gewöhnlich Laien
- **Texteigenschaften gemäß Korpus**
  - Experten: Jargon, termini technici, implizites Verständnis z.B. der Beziehung zwischen Gesetzesvorschriften und Schwellenwert-Typen
  - Laien: kein Jargon, Umschreibung technischer Ausdrücke, ausführlichere Kennzeichnungen

Hyperlinks erwiesen sich in beiden Fällen als wünschenswert

Parameter		
Expertise	*expert*	*novice*
Hyperlink	*ja*	*nein*

Parameter und alternative TGL-Regeln bewirken die Generierung unterschiedlicher Texte

- TGL-Regeln sind annotiert mit möglichen Parameterwerten
- Benutzer bestimmt, welche Werte gelten
- Konfliktlösung bevorzugt Regeln, deren Annotationen den Benutzerwerten am besten entsprechen

- Expertise: expert
  - **In summer 1997, the information value for ozone (180 µg/m³ according to the German decree Bundesimmissionsschutzverordnung) has been exceeded once.**
- Expertise: novice
  - **Between April and September 1997, the lowest threshold for ozone called information value, which is at 180 µg ozone per m³ air, has been exceeded once.**

TG/2 wird unter Forschungslizenzen in verschiedenen Anwendungen eingesetzt

- **Personalisierte Nachrichten**
  - für autonome Agenten bei Terminabsprachen im Dialogsystem COSMA (D; DFKI)
  - für tragbare Orientierungshilfe in Konferenz-Szenarios, EU-Projekt COMRIS (E; VU Brüssel)
- **Reportgenerierung**
  - Luftgüteberichte; Prototyp in fünf Sprachen, EU-Projekt TEMSIS (D, F, E, P; DFKI); **`http://www.dfki.de/service/nlg-demo`**
  - Tourismusinformation, EU-Projekt MIETTA (D, E, I; DFKI, CELI).
    Einsatz von JTg2Light (Java)
- **Zusammenfassungen**
  - begriffliche Repräsentation, EU-Projekt MUSI (D; DFKI)
- **Weitere untersuchte Einsatzmöglichkeit**
  - zur Berechnung von Konfigurationen von Softwaresystemen, EU-Projekt RAINBOW-II (nicht-sprachlich, DFKI)

Flache Generierung hat Vor- und Nachteile

EINSCHÄTZUNGEN

Vorteile

- Geringer Entwicklungsaufwand von Anwendungen durch Plattform
- wiederverwendbare Interpreter und Subgrammatiken
- sehr schnelle Verarbeitung
- leichte Einführung weiterer Sprachen
- problemlose Einfügung von alternativen Formulierungen (Präferenzmechanismus in TG/2)

Nachteile

- TG/2-Eingabestrukturen sind oft anwendungsspezifisch
- Textorganisation prinzipiell unklar
- Skalierbarkeit prinzipiell niedriger als bei tiefen Verfahren
- Erhaltung der Durchschaubarkeit kann bei größeren Anwendungen aufwendig werden