CaseSanitizer

Studienprojekt von

Hanna Peters und Youssef Sammari

Abstract

Der CaseSanitizer ist ein Programm, das durch die Identifizierung der in einem Text vorkommenden Abkürzungen und Namen eindeutige Satzgrenzen bestimmt. Das Verfahren arbeitet dokument-zentriert, d.h. unter Berücksichtigung des lokalen Kontextes und der Wortwiederholungen. Die Idee dabei ist, dass man zwischen eindeutigem Wortvorkommen (d.h. das Wort steht nicht nach einem potentiellen Satzende) und mehrdeutiger Wortposition im Satz unterscheidet. Da man die eindeutig vorkommenden Worte als Eigennamen oder gewöhnliche Worte klassifizieren kann, ist es dann möglich, Aussagen über dasselbe Wort in einer mehrdeutigen Situation zu treffen. Der große Vorteil bei diesem Verfahren ist, dass man nicht mehr auf Methoden der Statistik oder auf spezialisierte Grammatiken zurückgreifen muss.

Warum ist Bestimmung von Eigennamen und Abkürzungen für Satzgrenzen so wichtig?

The stock market's main price index rallied 77.12 points to a record 3,401.79 points, with volume at a frenzied 159.89 million shares. This has never happened before in N.Y. With U.S. long-term interest rates expected to remain steady after the Federal Reserve refrained from raising short-term rates on Tuesday.

Ginge man strikt nach dem Muster vor: Satzendezeichen sind ., ! und ?, so hätte man:

The stock market's main price index rallied 77.

12 points to a record 3,401.

79 points, with volume at a frenzied 159.

89 million shares.

This has never happened before in N.

Y.

With U.

S.

long-term interest rates expected to remain steady after the Federal Reserve refrained from raising short-term rates on Tuesday.

Nach Erkennung von Abkürzungen bzw. Datumsangaben :

The stock market's main price index rallied 77.12 points to a record 3,401.79 points, with volume at a frenzied 159.89 million shares.

This has never happened before in N.Y. With U.S. long-term interest rates expected to remain steady after the Federal Reserve refrained from raising short-term rates on Tuesday.

Nach Erkennung von Namen:

The stock market's main price index rallied 77.12 points to a record 3,401.79 points, with volume at a frenzied 159.89 million shares.

This has never happened before in N.Y.

With U.S. long-term interest rates expected to remain steady after the Federal Reserve refrained from raising short-term rates on Tuesday.

Im allgemeinen gilt für die Bestimmung der Satzgrenzen:
Quelle: Mikheev „Periods, Capatilazed Words, etc.“

Fehlerrate unseres Verfahrens	*Bei vollständigem Wissen über*
0.01-0.13%	*Abk. und Namen*
1.20 %	*Abk., aber Namen per lexikon-lookup*
0.45 %	*Namen, aber Abk. heuristisch*

Fehlerrate:= gibt das Verhältnis der nicht-korrekt erkannten zu allen vom System erkannten Satzgrenzen an.

Also: je besser wir Namen und Abkürzungen bestimmen desto genauer können wir Satzgrenzen bestimmen!!!

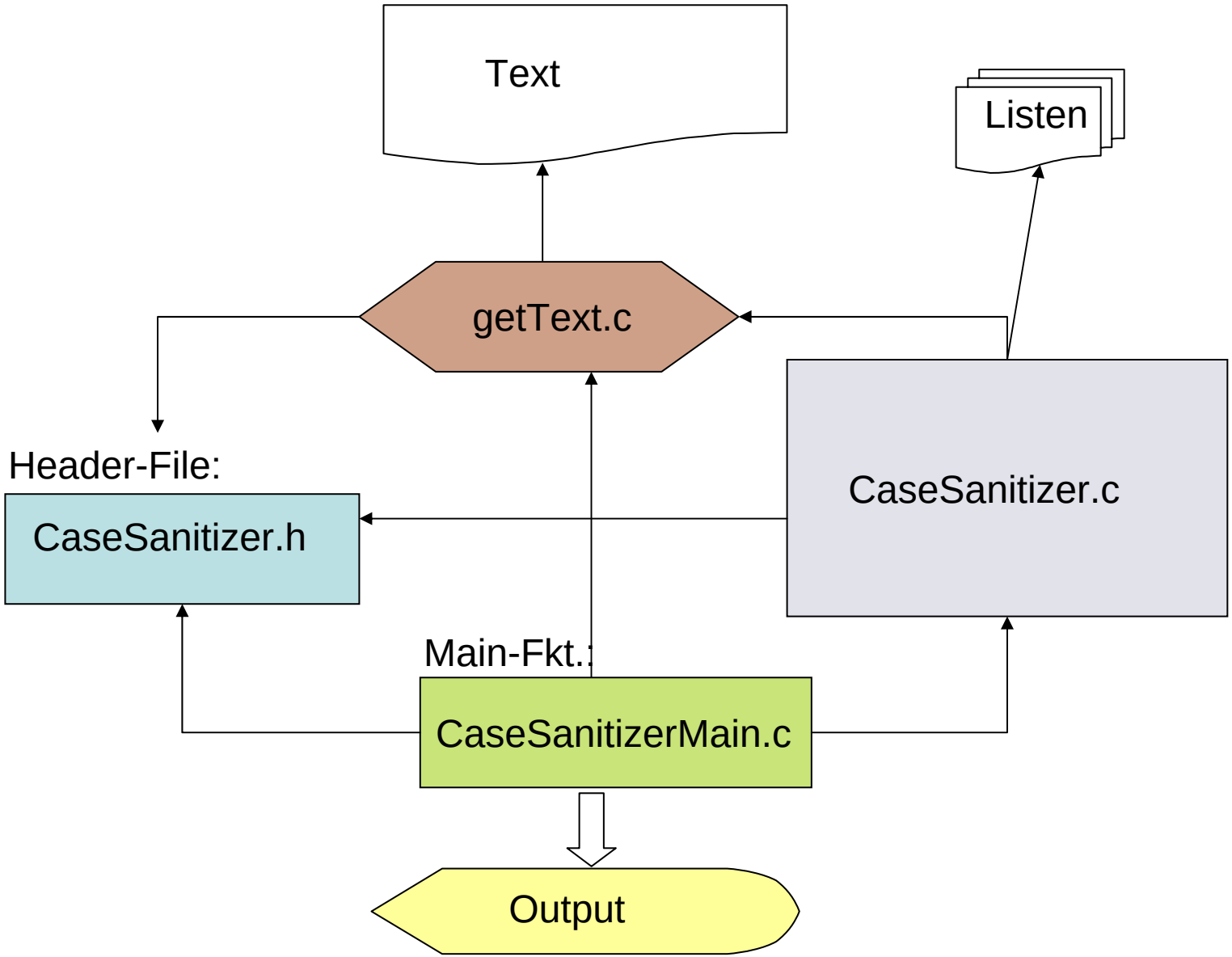
Probleme:

Text darf nicht zu klein sein (< 50 Wörter)
Text darf nicht zu groß sein (> 4000 Wörter)

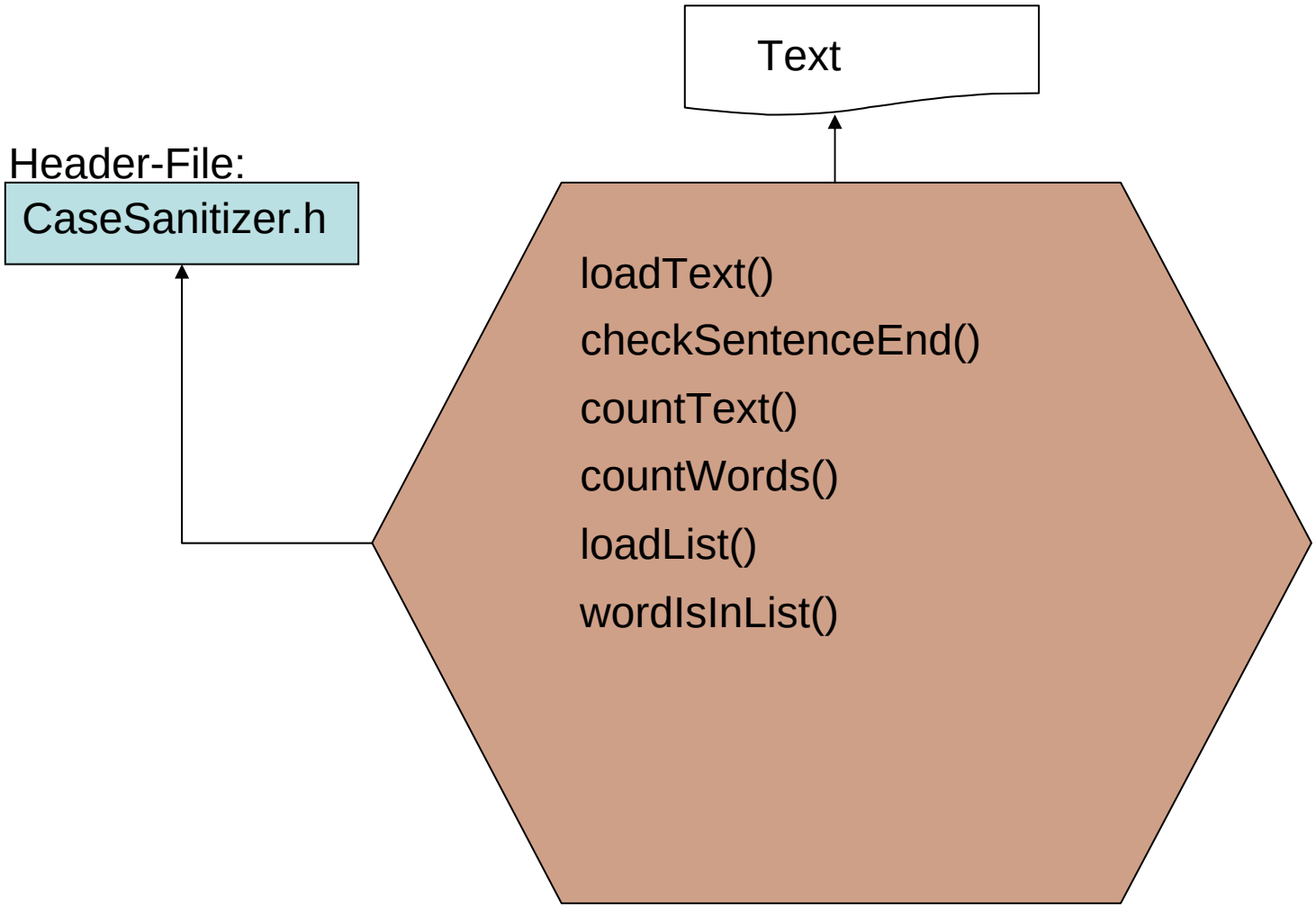
Ziele des Projekts

- **Textnormalisierung**
  - Bestimmung von Satzgrenzen ✓
  - Identifizierung von Abkürzungen ✓
  - Identifizierung von Eigennamen ✓
- **Robustes und flexibles Programm**
  - Insgesamt möglichst geringe Fehlerrate, d.h. hohe Erkennungsrate von Eigennamen und Abkürzungen ✓
  - Nur von wenigen externen Ressourcen abhängig ✓
  - Adaptivität an neue Sprachen
  - Domänenunabhängig ✓
- **Adaptierbarkeit**
  - Verwendung als Ressource für andere Projekte ✓

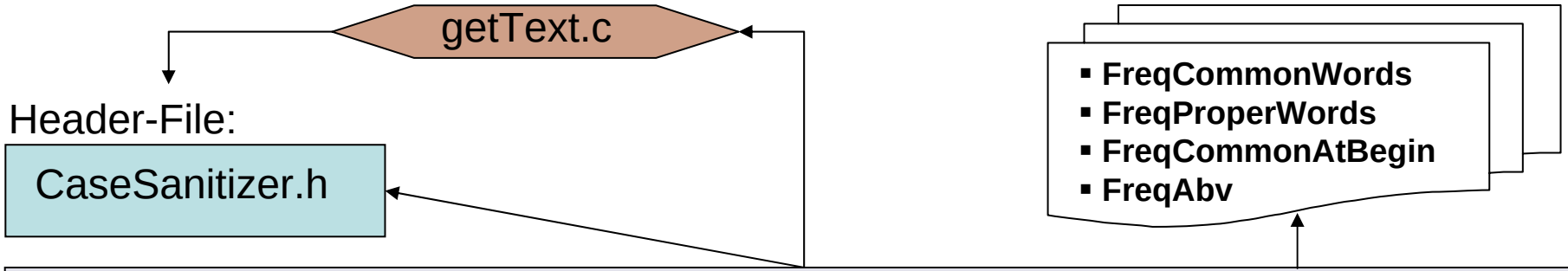
Struktur der Komponenten - Module



Modul : getText.c



Modul: CaseSanitizer.c



Abkürzungen:

- hasVowels()
- onlyNumbers()
- isAlwaysBig()
- isSequence()
- nextSmall()
- nextKomma()
- nextNumber()
- singleLetters()
- threePoints()
- abkCopy()
- abkRest()
- findAbkTripel()
- writeValueInHash()
- dateAtSentenceEnd()
- shorterThanMinWordLen()
- HList *list_new()
- Node *fillNode()
- abkTripelinList()
- printListe()
- keyCompare()
- keyCompareNodes()
- Node **getNodeArr()
- sortList()
- lenList()
- sucheBigramm()
- sucheKontext()
- bewerteBigramm()
- identifyAbk()

Namen:

- nameInKlammer()
- nameInZitat()
- nameHatTrennzeichen()
- nameHatSatzenden()
- copyNamen()
- findeNamen()
- writeValueInHashNamen()
- identifyNamen()

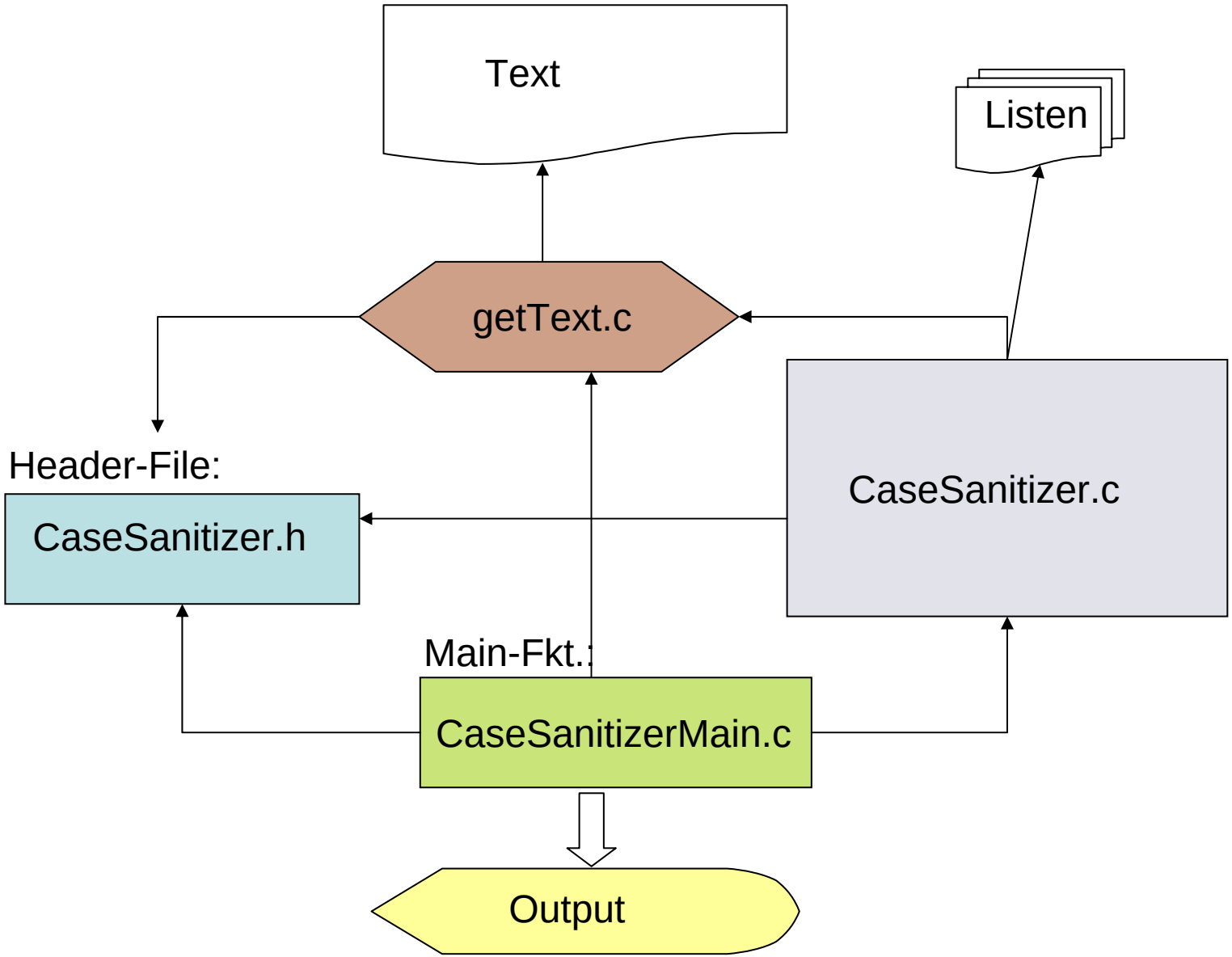
Textausgabe:

- macheAusgabe()
- printSatzenden()

Regeln:

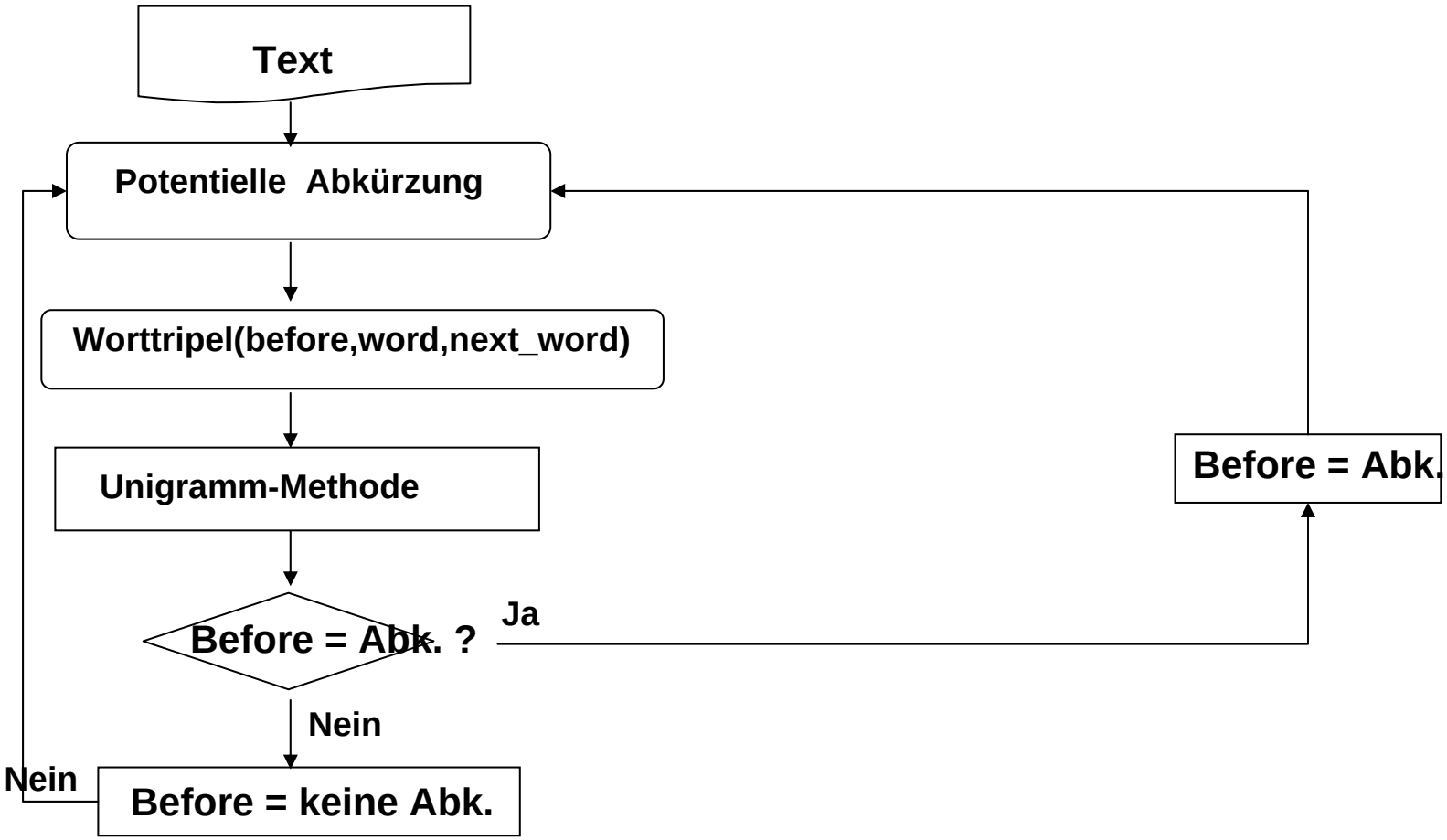
- ruleOne()
- ruleTwo()
- ruleThree()
- ruleFour()
- wendeRegelnAn()
- istAbsatzEnde()
- kleingeschrieben()
- grossgeschrieben()

Struktur der Komponenten - Module



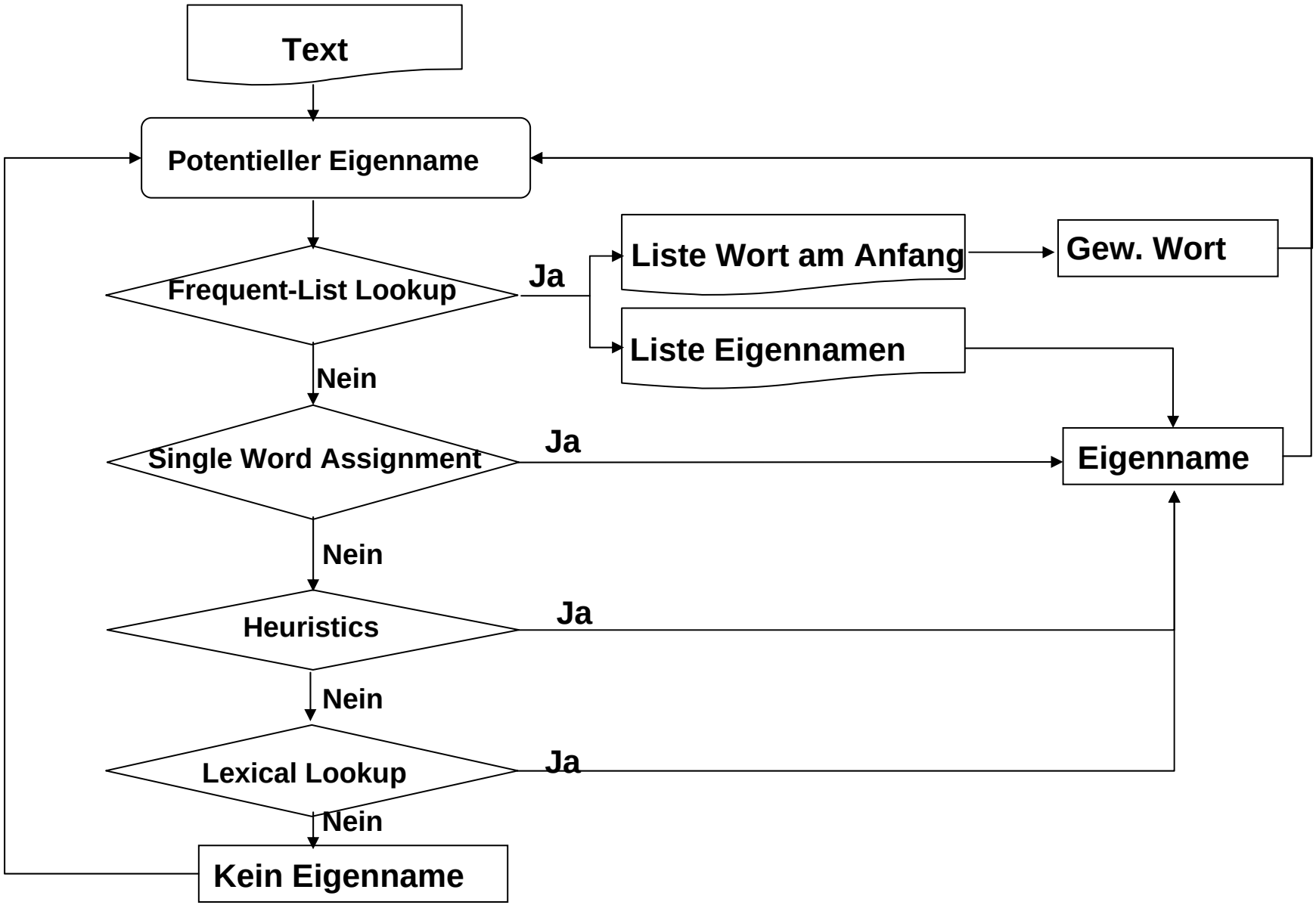
- **Grundregeln zur Bestimmung von Satzgrenzen**
  - Punkt nach Nicht-Abkürzung
    - ➢ Satzgrenze
  - Punkt nach Abkürzung und am Absatzende
    - ➢ Satzgrenze und Abkürzung
  - Punkt nach Abkürzung und darauf folgendes Wort ist nicht großgeschrieben
    - ➢ Abkürzung
  - Punkt nach Abkürzung und vor groß-geschriebenem Wort, das kein Eigenname ist
    - ➢ Satzgrenze und Abkürzung

Datenstruktur der Abkürzungen



- **Unigram DCA**
  - Lese bei jedem pot. Satzende das Worttripel(before,word,next_word) ein
  - Schreibe Worttripel in Liste (für Bigramm-Methode nötig)
  - Es handelt sich bei *before* um eine Abk. falls:
    - **Es in der Liste der Abk. steht**
    - **Es keine Vokale besitzt (Mr., kg)**
    - **Es nur aus einem oder mehreren Großbuchstaben besteht (Akronyme, Firmennamen)**
    - **Es aus einer Folge von Großbuchstaben besteht, die je durch einen Punkt getrennt sind (Y.M.C.A)**
    - **Es nur aus Ziffern besteht**
    - ***Word* nur eine Nummer, ein Komma oder ein kleingeschriebenes Wort ist**
    - **Weniger als 5 Buchstaben besitzt und *word* nur eine Nummer, ein Komma oder ein kleingeschriebenes Wort ist**

Datenstruktur der Eigennamen



- **Regeln zur Identifizierung von Eigennamen**
  - **Frequent-List Lookup Strategie**
    - Es werden 2 Listen aus dem Text generiert
      - Eine Liste mit normalen Wörtern, die häufig am Satzanfang stehen
      - Eine Liste mit häufig vorkommenden Eigennamen
  - **Single Word Assignment**
    - Falls ein Wort in nicht-ambigen Positionen immer großgeschrieben wird → Eigenname
    - Falls ein Wort in nicht-ambigen Positionen immer kleingeschrieben wird → normales Wort

- Heuristisches Verfahren

- Folgende heuristische Regeln bestimmen einen Eigennamen
  - **Ein einzelnes groß geschriebenes Wort in Klammern oder in Zitat**
    - » John (Cool) Lee
  - **Falls nach Kleinschreibung, Nummer oder Komma ein Klammer folgt und das erste Wort ist groß geschrieben**
    - » *… happened(Moscow…) but …*
  - **Nach einer groß geschriebenen Abk. folgt ein groß geschriebenes Wort und dieses Wort ist nicht in der Liste der häufig am Satzanfang vorkommenden Wörtern enthalten**
    - » *Dr. Peters*

- Lexical-Lookup Strategie

- Restliche ambige Wörter, die nicht in der Liste der gewöhnlichen Wörter auftauchen → Eigennamen

Pro:

- **Gute Zusammenarbeit**
- **Gute Ergänzung**

Kritik:

- **Teilweise zu große Arbeitsteilung**
- **Zeitmanagement-Probleme**
- **Teilweise Abstimmungs- / Koordinationsprobleme**

→ Insgesamt gutes Teamwork

Ausblick:

- **Code-Optimierung**
- **Automatische Listenerstellung bzw. Listenextraktion aus Corpus**
- **Umsetzung in andere Sprachen**
- **Graphische Oberfläche zur Verwendung als Stand-alone Anwendung**

…CaseSanitizer

Studienprojekt von

Hanna Peters und Youssef Sammari