Doctorial Symposis Material (Unsortiert)

ASR:

* Third-party: Janus, Google Speech API
* Bestenfalls mit Hypothesen und Konfidenzen
  + Hypothesen können in mehreren STRs münden
  + Konfidenzen zur Bewertung der aktuellen Annahme über die Nutzerintension
  + Graphsysteme die parallel berechnet werden (pro Satz, komplette Engabe) bilden belief system für sich und ein state in einem DS-weiten belief system
* 2-step Methode (ASR+NLU) nicht optimal (ZITAT SLU-Buch), aber Ausrichtung des Gesamtforschungsprojekt erfordert es, da auch Text als Eingabe dienen soll
  + Deshalb Nutzung des selben Modells für ASR wie für NLU
  + Grammatik oder Soundex oder ähnliches mit Termen aus der Domäne
  + Terme bekommt man automatisch aus Ontologie, SKILL DB, API
  + Synonyme?
  + Synonyme Phrasen?
* Anforderungen:
  + Niedrige ER (WER, SER, CER)
  + Hypothesen, Konfidenzen
  + Ohne Training
  + Alle domänen
  + Trade-off machen

Initial NLU:

* Umwandlung Text in logische Form (STR)
* Keine vollständige NLU, das passiert Modular auf STR
* Im Grunde Syntax/Semantik-Analyse
  + POS, parseTree, dep. Graph, **sem.Rollen**
* Wie?
  + NLP -> Regelbasiert:
    - POS-Tags stabil. Nur abhängig von WER
    - Parse Trees haben Struktur
    - Dep. Trees & sem roles nicht verwendbar (da wollen wir aber hin)
    - Weltwissen einbeziehen?
    - Chunks funktionieren
    - Evtl. unterschiedl. Tools als Agenten, die Ergebnis mit Konfidenz liefern
    - Dazu Judge (ähnlich Watson)
  + Statistisch
    - Neuronales Netz auf Token
    - HMM
    - MEMM
    - Stanford Classifier selber bauen
    - Als features wortweite Eigenschaften aus Vorverarbeitungsschritt
      * POS
      * Lexname (WordNet)
      * Chunk (Phrasenzugehörigkeit, head or not)
      * Wort an sich
    - Problem: teuer, Aufwendig
  + Möglichkeit: stabiles System regelbasiert bauen
    - Durch statistische Methoden verbessern
    - Wahrscheinlich online
    - Initiales System muss das ermöglichen

SSR (Semantic Speech Representation)

* Graph-basiert
* Enthält alle Wörter als Knoten (Knoten sind Objekte mit Attributen)
* Kanten mit Relationen dazwischen
* Enthält zunächst Ergebnis der Initial NLU
* NLU-Module ändern Gestalt des Graphen
* Graph dient als gemeinsame Wissensgrundlage
* Anforderungen
  + Änderbar
  + Knoten und Relationen haben Attribute
  + Konsistenzprüfung nach Änderung
  + Reasoning auf Graph (anhand von Regeln)
  + Dazu Definition von Vorbedingungen für Eintritt in Knoten
  + Token (zur Markierung aktiver Knoten)?
  + Sollte wissen welche NLU-Module gelaufen sind (state)
* Im Sinne der Disambiguierung könnte es sinnvoll sein mehrere Graphen zu haben auf denen man arbeitet
  + der mit der höchsten Konfidenz wird zuletzt ausgewählt
  + Erfordert den Umgang mit Konfidenzen für NLU-Module (sowohl als Eingabe als auch für die Ausgabe)
  + Problem: Echtzeitanforderung
  + Problem: vielleicht ist Mapping zwischen Graphen sinnvoll, weil sie stückweise hohe Konfidenzen aufweisen

Schnittstelle zu Anwendungsdomäne

* Stellt Fassade zur Eigentlichen Domäne dar
* Gegenüber PARSE-SSR domänenunabhängig
* Darunterliegende Domäne muss auf dieses Interface zugeschnitten werden
* Erleichtert Mapping
* Eventuell zu restriktiv. Schwierig zu entwerfen
* Idee des Service-Brokers adaptieren?
* Es erlauben an Schnittstelle Module/Dienstgeber zu registrieren

NLU-Module

* Liefern die zusätzliche Semantik
* Haben SSR als gem. Wiisen/Schnittstelle
* Agenten-basiert
* Mögliche Module:
  + Kontext-Analyse
    - Konzeptbildung
    - Coref
    - Synonyme
    - Wortfehler-Korrektur durch Kontext
  + Memory
    - Longterm
      * Nutzereigenheiten
      * Grundsätzliches Wissen über Situation
    - Short term
      * Beliefs (Nutzer-Intension)
      * Momentanes Wissen (z.B. über Objekte)
  + Weltwissen (Anbindung von Weltwissenontologien, wie…)
    - Cyc
    - FrameNet
    - ConceptNet
* Module müssen sich am Graphen registrieren (und haben vielleicht andere Module als Vorbedingung -> haben das als Attribut, im Graph vermerkt was schon gelaufen)
* Module können über Lebenszeit des Graphen existieren, müssen sich aber in neuem Graph erneut anmelden (zusätzlich muss die extended lifetime als Attribut vermerkt sein)