1 Die Basistypen

Die Schnittstellen des Projektes sind geeignet zu definieren. Wir benötigen dazu Basistypen zur Darstellung einer Suchmaschine, einer Datenbank, einer Anfrage und eines Ergebnisses:

```
TYPE t_suchmaschine = RECORD
                         nummer      : integer;
                         name        : string;
                         beschreibung: string;
                         help_text   : string
                      END;

TYPE t_datenbank    = RECORD
                         nummer      : integer;
                         name        : string;
                         beschreibung: string;
                         help_text   : string
                      END;

TYPE t_anfrage      = VARIANT-RECORD
                         IQL: anfrage: string;
                       | SQL: anfrage: string;
                      END;

TYPE t_table        = list[list[t_feld]];
```

Eine Anfrage ist dabei ein String in SQL oder IQL und das Ergebnis ist im Prinzip eine Tabelle.

2 Das Frontend

Das Frontend (z.B. die WWW-Oberfläche) kann wie folgt aussehen:

```
PROC frontend ();
VAR db: list[t_datenbank];
    sm: list[t_suchmaschine];
    anfrage: t_anfrage;
    ergebnis: t_table;

BEGIN
   db := query_db ();
   sm := query_sm ();
   built_frontend (sm, db);
   :
   :
   ergebnis := do_query (anfrage, sm[i], db[j]);
   show_erg (ergebnis);
END frontend;
```

Es werden zunächst die vorhandenen Datenbanken und die implementierten Suchmaschinen ermittelt und mit deren Hilfe ein geeignetes WWW-Formular (oder mehrere) generiert.

Nach der Eingabe der Daten wird die Anfrage an einen zentralen Anfrageknoten gestellt. Eine Anfrage muß in einem *genormten* Format vorliegen. Das kann z.B. in IQL-Form sein. Direkte SQL-Anfragen sind ebenfalls zulässige Anfragen, die an die globale Suchmaschine gestellt werden können.

3 Die Suchmaschine

Zunächst gibt es die beiden Basisfunktionen `query_sm` und `query_db`, die die vorhandenen Suchmaschinen bzw. die vorhandenen Datenbanken zurückliefern:

```
PROC query_sm (): list[t_suchmaschine];
VAR sm: list[t_suchmaschine];
BEGIN
   :
   :
   RERURN (sm);
END query_sm;

PROC query_db (): list[t_datenbank];
VAR db: list[t_datenbank];
BEGIN
   :
   :
   RETURN (db);
END query_db;
```

Eine vom Frontend gestellte Anfrage wird über den zentralen Anfrageknoten `do_query` an die entsprechende Suchmaschine weitergeleitet. Mit einem bißchen KI könnte der Anfrageknoten sogar die „beste“ Suchmaschine für diese Anfrage auszuwählen.

```
PROC do_query (anfrage: t_anfrage;
               sm[i]  : t_suchmaschine;
               db     : list[t_datenbank]): t_table;
BEGIN
   CASE sm[i] OF
      1: RETURN (suchmaschine_1 (anfrage, db));
     |2: RETURN (suchmaschine_2 (anfrage, db));
      :
      :
     |N: RETURN (suchmaschine_N (anfrage, db));
   END; (*CASE*)
END do_query;
```

Die einzelnen Suchmaschinen (durch die Prozedur `suchmaschine_I` angedeutet) wandelt die IQL-Anfrage in eine SQL-Anfrage um. (Dies ist natürlich nicht erforderlich, falls die Originalanfrage bereits eine SQL-Anfrage ist.) Diese Anfrage wird dann an die ausgewählten Datenbanken weitergereicht und das Ergebnis in Form einer Tabelle zusammengefügt.


```
PROC suchmaschine_I (anfrage: t_anfrage;
                     db     : list[t_datenbank]): t_table;
VAR table: t_table;
BEGIN
   :
   :
   sql_anfrage := translate_to_sql (anfrage);
   table := {};
   FOR ALL j IN db DO
      CASE db[j] OF
         1: table := table ++ datenbank_1 (sql_anfrage);
        |2: table := table ++ datenbank_2 (sql_anfrage);
         :
         :
        |N: table := table ++ datenbank_N (sql_anfrage);
      END; (*CASE*)
   END (*FOR*)
   RETURN (table);
END suchmaschine_I;
```

4 Die Datenbank

Die jeweilige Datenbank hat auch noch eine kleine Aufgabe zu übernehmen. Die durch die Suchmaschine erzeugte SQL-Anfrage muß zum Abschluß noch auf die speziellen Verhältnisse der Datenbank angepaßt werden. So sind z.B. Zeilen oder Tabellen anders zu benennen oder eine Anfrage in zwei Teile zu zerstückeln. Dabei kann es natürlich auch vorkommen, daß diese Informationen gar nicht in dieser Datenbank enthalten sind.

```
PROC datenbank_I (sql_anfrage: t_anfrage): t_table;
VAR table: t_table;

BEGIN
   :
   :
   sql_anfrage_neu := translate_to_dbI_sql (sql_anfrage);
   table := do_db_query_I (sql_anfrage_neu);
   return (table);
END suchmaschine_I;
```