**DELIVERABLE SUMMARY SHEET**

Project Number: ***IST–2001-33049***

Project Acronym: **PROTOCURE**

Title: ***Improving medical protocols by formal methods***

Deliverable N°: 3'

Due date: 31/5/2002

Delivery Date: 31/5/2002

**This report is in addition to the demonstration of the KIV formalisation of the reference protocols.**

Short Description:

**Objectives**

To translate the example Asbru protocols Jaundice and Diabetes to a format understandable by KIV.

**Procedure**

How to bridge the gap between the protocol language Asbru and the interactive theorem prover KIV has been examined. It has been decided to translate Asbru to enriched parallel programs, a format understandable by KIV. To ease the translation, additional operators for parallel programs have been defined in KIV. The example Jaundice and Diabetes protocols have been translated to KIV. Rudimentary translation patterns have been derived. Based on this, automation of the translation has been examined and partially implemented.

**Results**

The deliverable contains the translated protocols, along with a report including comments, experiences, translation patterns and future developments concerning the translation process. It also discusses alternatives to bridge the gap between Asbru and KIV.

Partners owning: all

Partners contributed: Universität Augsburg, Technische Universität Wien, Vrije Universiteit Amsterdam

Made available to: Public

# KIV formalisation of reference protocols
# Version 1.0

May 31, 2002

## Contents


**1 Overview** **1**

**2 Asbru in KIV** **2**
2.1 Parallel programs . . . 3
2.2 Additional operators . . . 3
2.2.1 Interrupts . . . 3
2.2.2 Plan ordering . . . 3
2.3 Encodings . . . 3
2.3.1 Continuation condition . . . 4
2.3.2 Time annotations . . . 4
2.4 Open issues . . . 4

**3 Translation of Example Protocols** **4**
3.1 Jaundice . . . 4
3.1.1 Plan hierarchy . . . 4
3.1.2 Plan state . . . 4
3.1.3 Data abstraction . . . 5
3.1.4 Patient record . . . 5
3.2 Diabetes . . . 5

**4 Conclusion** **5**

**5 Future** **6**

**A Translation Patterns** **7**

**B Example Protocols** **12**
B.1 The Jaundice Protocol . . . 12
B.1.1 The Specification Graph . . . 12
B.1.2 The Specifications . . . 14
B.2 The Diabetes Protocol . . . 36
B.2.1 The Specification Graph . . . 36
B.2.2 The Specifications . . . 38


# 1 Overview

Our goal is to try to formally verify properties of medical protocols and to discover errors in the protocol during this attempt. Formal verification is not possible without tool support. As a tool, we have chosen the interactive theorem prover KIV.

As a first step, we need to formalize the informal medical protocol. This has already been done for two example protocols in Deliverable D1. Protocols are now available in the Asbru language. For Asbru to be a formal notation, the semantics of Asbru language constructs need to be clear without ambiguity. Therefore, in a second step, we have defined a formal semantics of (part of) Asbru (see Deliverable D2). The Asbru language is not known to KIV from the start. In a third step, we need to close this gap. This is what we report on in this Deliverable D4.

There are several options to deal with Asbru in KIV:

- Formally define the semantics of Asbru constructs in KIV (e.g. in higher order logic) and construct proofs for protocols on the semantics level (which is a very low and detailed level).
- Implement Asbru constructs in KIV and define (higher level) proof rules for these constructs.
- Translate Asbru protocols to a format already known to KIV (such as predicate logic, higher order logic, temporal logic, parallel programs, or state charts) and construct proofs using existing (higher level) proof rules for these format.

If we want to examine the language itself, the first approach would be a good choice. For the verification of plans written in Asbru, the level of the approach is too low. The resulting proofs are too detailed and thus too large. In order to keep proofs small and simple, the second approach is very promising. On the other hand, it requires a lot of effort on KIV improvements. If translation is not too complicated, the third option should also lead to manageable proof sizes, while minimizing the overall effort. Therefore we have voted for the third option.

As it turned out, translation is not always straightforward. Some Asbru constructs required complex encodings leading to complex proofs. In order to avoid this problem, we have decided to also implement a small number of additional constructs in KIV. We have tried to find a balance between effort to implement these constructs and avoiding difficulties in the translation.

In the following, Section 2 describes general treatment of Asbru plans in KIV. Appendix B contains the translated example protocols Jaundice and Diabetes. Comments on the translation can be found in Section 3. Further sections conclude and point out future possibilities.

# 2 Asbru in KIV

Asbru plans describe concurrent systems: several things may happen at the same time/in parallel. In KIV, parallel programs can be used to specify concurrent systems. The fastest way to receive proof support for Asbru plans is therefore to translate Asbru to parallel programs. However, Asbru operators are different from operators in parallel programs and translation to pure parallel programs is sometimes very complex, requiring to explicitly encode part of the control flow. With complex encodings, constructing formal proofs in KIV would be hard. Also, the gap between Asbru and KIV would be large. If the gap is too large, it is difficult to trace errors which are discovered in KIV back to the original Asbru protocol. Therefore, we have defined some additional operators for parallel programs, in order to directly support features of Asbru, which are most different from pure parallel programs. With these operators, a more direct translation is possible and the gap between Asbru and KIV is reasonably small. It would be possible to support all of the Asbru operators and to verify Asbru plans directly in KIV, but the necessary additions to KIV cannot be implemented within our one year assessment project. For the purpose of investigating, whether formal verification of medical protocols leads to the discovery of errors, the additional operators defined below are sufficient.

In the following, we will describe parallel programs in KIV (Section 2.1), and afterwards introduce the operators which were added (Section 2.2). Features of Asbru which still need to be encoded in KIV are described in Section 2.3. In Section 2.4, we will list features of Asbru which are not yet properly supported.

## 2.1 Parallel programs

For parallel programs in KIV, the following operators can be used:

- assignments $v := \tau$
- sequential composition $\alpha_1; \beta$
- conditionals **if** $\varepsilon$ **then** $\alpha$ **else** $\beta$
- loops **while** $\varepsilon$ **do** $\alpha$
- local variables **var** $v_1 = \tau_1, \ldots, v_n = \tau_n$ **in** $\alpha$
- procedures $p\#(\tau_1, \ldots, \tau_n; v_1, \ldots, v_n)$
- interleaving $\alpha \mid\mid \beta$
- synchronisation **await** $\varepsilon$

## 2.2 Additional operators

### 2.2.1 Interrupts

Some Asbru operators define a control flow which is difficult to translate into parallel programs. For example, an abort condition can be used to interrupt the execution of the plan body in the case of an emergency. In parallel programs as described in Sect. 2.1, there is no operator to properly model interrupts. To encode interrupts using existing operators is complex. Therefore we have added an additional operator

**pbreak** $\alpha$ **if** $\varepsilon$

to KIV. An abort condition can now be directly translated as follows:

**pbreak**
  ⟨*body*⟩
**if** ⟨*abort condition*⟩

### 2.2.2 Plan ordering

Asbru allows to order the execution of sub plans in a plan body in nontrivial ways. Besides sequential execution of sub plans, the plans can be executed in `any order`, in `parallel` and `unordered`. The semantics of these alternatives is described in Deliverable D2. For translating the plan body, two additional operators turned out to be useful:

- $\alpha \mid\mid_a \beta$ which is sequential execution, but in any order. Either $\alpha$ is executed first $(\alpha; \beta)$ or $\beta$ is executed first $(\beta; \alpha)$. The order depends on which sub program can be activated first. If both can be activated at the same time, the order is nondeterministic. This operator is used to translate plan bodies of type `any-order`.
- $\alpha \mid\mid_s \beta$ which is synchronous parallel execution. The sub programs execute steps synchronously (at the same time). This operator can be used to translate plan bodies of type `unordered`, and – in some special cases – `parallel`.

## 2.3 Encodings

The most important features are directly supported in KIV. Other features still need to be encoded.

#### 2.3.1 Continuation condition

Execution of sub plans can be mandatory or optional. This is specified in the continuation condition (see Deliverable D2 for an explanation). The continuation condition currently cannot be directly translated to KIV. Therefore it is encoded using additional program variables containing the current state. For a plan called `p` a variable named `p-state` is used. Further details are described in Appendix A.

#### 2.3.2 Time annotations

Also time annotations are encoded. A special program variable 'time' is used to refer to the current time. A time annotation is translated into a set of await statements. Because time annotations are rare in our example Asbru plans, we did not investigate a more general translation scheme for time annotations.

### 2.4 Open issues

With the additional operators, translation is rather straightforward. The encodings are not too difficult to handle, and formal proofs for the translated Asbru plans are manageable. For now, the only construct of Asbru which is really used in the example protocols and which cannot be translated very easily is the `do-parallel` operator.

## 3 Translation of Example Protocols

Both example protocols Jaundice and Diabetes have been translated to KIV. The resulting specifications are contained in Appendix B. Here, we summarize our experiences with the translation process.

### 3.1 Jaundice

The Jaundice protocol has been translated first. The translation was manual and contains very specific solutions for various cases. The idea was to keep the resulting parallel programs as simple as possible.

It very roughly took 2 person months to translate Jaundice to KIV.

In the following, we will point out some specialties of the translation.

#### 3.1.1 Plan hierarchy

All the major Asbru plans have been translated into separate specifications, resulting in the specification graph represented by Figures 1 - 4. Asbru plans representing atomic actions have been partially included with other plans in one and the same specification in order to reduce the total number of specifications. This approach turned out to be very useful, as the hierarchy of Asbru plans is preserved and navigation through specifications is very easy.

#### 3.1.2 Plan state

Translation of Asbru plans sometimes requires a parent plan to refer to the current state of its sub plans. For example, a plan is interrupted, if a mandatory sub plans has been aborted. This is known as the continuation condition of Asbru plans. In order to model the continuation condition, the execution state of plans needs to be encoded in additional program variables (see also Section 2.3.1). For the translation of Jaundice a program variable has only been added, if it is really necessary to refer to the execution state, i.e. there does not exist a different translation which avoids referring to plan states.

### 3.1.3 Data abstraction

In Asbru, plans often refer to qualitative abstractions of quantitative measurements. For example, in Jaundice the level of bilirubin in the blood is abstracted to quantitative values `observation`, `phototherapy-recommended`, ... (compare to specification `bilirubin` on page 32). The definition of these qualitative values as well as the abstraction functions are defined in the lower part of Figure 4. For modelling the data abstraction, standard algebraic specifications have been used.

### 3.1.4 Patient record

A number of patient parameters are dealt with throughout the protocol. It turned out, that the parameters are best stored within a central data structure `patient-data`. This keeps parameter lists in procedures smaller and improves readability.

## 3.2 Diabetes

Diabetes was the second protocol to be formalized in KIV after Jaundice. Translation patterns which were derived from jaundice helped to translate Diabetes (see Appendix A). In the following we highlight the more important and/or frequent ones.

In KIV each specification enriches other specifications (if it is not elementary). This can be defined interactively using the graphical user interface of KIV, or it can be defined by adding the appropriate clause into the specification file. Since every Asbru plan is encoded in a separate specification, we automatically produced an enrich statement for each of them. The enrich relations between specifications were deduced from the Asbru plan-schema element plus a standard list of specifications used by every specification. Later, the specifications of user-performed plans (which are very small and do not interact with other plans) which were called by exactly one plan were joint with those of the calling plan.

To implement plan state propagation, each plan has a "var"-parameter, in which it stored its final plan state. The parent defines this value as a local variable. This pattern was also deduced from the plan-schema element in Asbru. Since the simple automations used do not check, whether a childs plan state is accessed by the parent, many unnecessary statements and parameter passings were produced. We assume that this more general approach can be handled in the verification process.

Based on the patterns described in Appendix A we automatically produced first versions of those statements dealing with plan state propagation in the parent. The information needed was extracted from the various attributes of the Asbru element subplans. Due to the simple design of the translation tool, nesting of such elements was not handled correctly. But in simple cases (which are frequent) no further editing was needed.

While translating Diabetes, automation was used to reduce the workload of typing. Based on opportunistic selections of priorities, the most frequent constructs were translated from Asbru to KIV using Perl. With the help of rudimentary automation, the Asbru version of Diabetes was translated to KIV within less than 1 person month. This estimation also includes the effort to actually implement the automation.

# 4 Conclusion

Although the translation of Asbru plans into KIV is not straightforward, the gap between both is narrow compared to the gap between the informal protocol and Asbru. While constructing the Asbru plan is a challenging task, the difficulties in the translation to KIV are technical in nature. As more of these technical issues are solved, translation to KIV becomes easier, more automatic and thus faster. This observation is supported by a comparison of efforts. Construction of Asbru plans takes significantly longer than translation to KIV. Also, the Diabetes protocol was translated faster than Jaundice. This was thanks to "translation patterns" which we could reuse from the earlier Jaundice translation. In addition, part of the Diabetes translation was automated.

Direct support of at least some of the Asbru constructs has been essential for proofs. We have tried to prove some toy Asbru plans, which were hand translated to parallel programs without additional operators. The size of the resulting proofs was larger by magnitudes compared to proof sizes, if the additional operators were used. Thus we think, that it is worthwile to spend effort on additional Asbru operators in KIV.

The translation to KIV is in no way formally correct. For the purpose of our assessment project, the translation is good enough: errors found in the KIV version of the protocol can be manually traced in the Asbru version. Either the error is also present in the Asbru plans, or an error in the translation has been uncovered. However, no correctness guarantees for the Asbru plan can be established without examining the correctness of the translation first.

# 5 Future

It is possible to implement even more constructs of Asbru directly in KIV. Thus the gap between Asbru and KIV would become smaller and translation more straightforward. However, this requires significant effort.

Even without further constructs in KIV, a complete automation of the translation of all elements of Asbru Light into KIV seems possible. Such a script would best be implemented in XSL. The output of any automated translation will be suboptimal with respect to proof size and some manual optimization might be necessary. On the other hand, applying KIVs proof heuristics, potentially unnecessary routine part of the plan library could automatically be dealt with, so no additional labour would be required. There are two big advantages of an automatic translation of Asbru to KIV which exceed the drawbacks described above: The human effort in the translation process would be reduced dramatically and there would be a guarantee that there are no (human) mistakes introduced in the course of the translation. This would also reduce the problem of tracing errors found during verification back to the original Asbru plan. We believe that these advantages make automatic translation worthwhile to be pursued in a future project.

If we want to establish some kind of correctness guarantees for Asbru plans using KIV, we also need to spend effort on the examination of the correctness of the translation bridging the gap between Asbru and KIV.

# Implementation of Asbru Plan Ordering in KIV

## Plan state of parent

After execution of children, the plan state of the parent is set.

### all

```
if child-a-state [.] = completed and child-b-state [.] = completed
then parent-state := completed
else parent-state := aborted
```

### none

```
parent-state := completed
```

### one

```
if child-a-state [.] = completed or child-b-state [.] = completed
then parent-state := completed
else parent-state := aborted
```

### A certain child

```
given plan b is the certain child:
if child-b-state [.] = completed
then parent-state := completed
else parent-state := aborted
```

## Temporal aspects

There are four dimensions to plan ordering to be observed:

1) the ordering proper: sequential, any-order, parallel, unordered
2) the continuation specification proper, i.e., the condition under which the parent successfully completes (if there is no additional complete condition)
3) the inverse of the continuation specification, i.e., the condition, under which successful completion is precluded and the parent immediately aborts
4) modifications of the continuation specifications:
   a) wait-for-optional-subplans,
   b) retry-aborted-subplans

In the following, these dimension are described in a simplest-first approach. Then the complete range of variations is described in detail for sequential and any-order.

### Principal plan ordering

sequentially

```
a ; b
```

any-order

```
a ||a b
```

parallel

not yet implemented but ||s is a good approximation, even perfect if there is no filter condition and activation is automatic.
The precise model of parallel(a, b) would be

```
(: filter condition in parallel, then synchronize :)
    await filter condition of a
||s await filter condition of b;
(: manual activation in parallel, then synchronize :)
    var time0 = time [.], dt = [?] in await time0 [.] + dt [.] \le time [.]
||s var time0 = time [.], dt = [?] in await time0 [.] + dt [.] \le time [.];
(: plan body started together, rest not synchronized :)
    plan body of a
||s plan body of b
```

unordered

```
<child> ||s <child>
```

These simplest cases represent the variety "wait for none, but wait for optional subplans", i.e., neither success nor failure of subplans has any consequence.

## Continuation specification

If wait-for-optional-subplans is not given or false, the parent terminates, as soon as the continuation condition is fulfilled.
This means, that above statements are enclosed in a break-statement:

```
break
  ... above statements ...
if continuation specification
```

The continuation specification is specified in terms of plan states of the subplans.
Wait-for one is encoded as disjunction of all plans.
Wait-for all would be a conjunction of all plans and therefore redundant. So *this aspect* of "wait for all" does not need to be modeled.

### Negative continuation specification

If a continuation specification is given and falsified during execution of some children, e.g., because a mandatory child failed, the parent aborts immediately.
This is implemented as a break (the same break as above). The condition of the break is the logical complement of the continuation condition.

This also holds if wait-for-mandatory-subplans=yes, since they cannot change the failure of the parent.

### Retry aborted subplans

If this option is given (set to yes), each plan enters the selection phase immediately after abortion, independent of any other plans.
This is implemented as a while-loop in KIV

```
while child-state \ne completed do
  child#(...)
```

Such plans must be excluded from the negative continuation specification - otherwise the while loop would be futile.
"Wait for optional subplans" need not be given together with "retry aborted subplans". If it is not "yes", the while-loops of optional subplans are disrupted by the break implementing the continuation condition.

## Principal construction of a plan (BNF-style)

Note that CAPITALIZED words and such under quotes are meta-language, where small letters are KIV code.

```
<parent> :=
   break
      <children>
   if    <pos-cont-cond>
      or <neg-cont-cond>
      or <abort-cond>
      or <complete-cond>;
   if <cont-cond> and <complete-cond>
   then parent-state := completed
   else parent-state := aborted

<children> :=
   IF "ordering = sequentially" THEN
      <child> ; <child>
   ELSE IF "ordering = any-order" THEN
      <child> ||a <child>
   ...

<child> :=
   IF "retry-aborted-subplans" THEN
      while child-state [.] \ne completed
        do child#(...)
   ELSE
      child#(...)

<cont-cond> :=
   IF "all" THEN
      child-state [.] = completed and child-state [.] = completed
   ELSE IF "none" THEN
      true
   ELSE IF "one" THEN
```

```
      child-state [.] = completed or child-state [.] = completed

<pos-cont-cond> :=
   IF "wait-for-optional-subplans=yes" THEN
      false
   ELSE
      <complete-cond>

<neg-cont-cond> :=
   IF "all" THEN
      child-state [.] = aborted or child-state [.] = aborted
   ELSE IF "none" THEN
      false
   ELSE IF "one" THEN
      child-state [.] = aborted and child-state [.] = aborted
```

<complete-cond> and <abort-cond> are defined by the corresponding Asbru elements.

## All combinations for a sequential plan

| wait-for | wait-for-optional-subplans | retry-aborted-subplans | KIV code | comments |
|---|---|---|---|---|
| none | no | no | break<br>a; b; c<br>if <any reason to stop> | as soon as the complete condition holds, execution of children is discontinued |
| none | yes | no | a; b; c | the end of c is waited for |
| none | no | yes | | this is generally a nonsense combination for sequential plans (not for the others): I cannot retry the children an not wait for them |
| none | yes | yes | while a-state [.] \ne completed do<br>  a;<br>while b-state [.] \ne completed do<br>  b;<br>while c-state [.] \ne completed do<br>  c; | |
| all | there are none | no | break<br>  a; b; c<br>if a-state [.] = aborted<br>or b-state [.] = aborted<br>(or c-state [.] = aborted) | The last part is not necessary since/if c-state is only set in the last statement of c.<br>We break, as soon as failure is inevitable. |
| all | yes | yes | as none-yes-yes | |
| one | no | no | break<br>  a; b; c<br>if a-state [.] = completed<br>or b-state [.] = completed<br>(or c-state [.] = completed) | We successfully complete as soon as one child completes and do not wait for the others. |
| one | yes | no | a; b; c | |
| one | yes | yes | as none-yes-yes | |
| plan b | no | no | | c would never be done |
| plan b | yes | no | break<br>  a; b; c<br>if b-state [.] = aborted | |
| plan b | yes | yes | as none-yes-yes | |

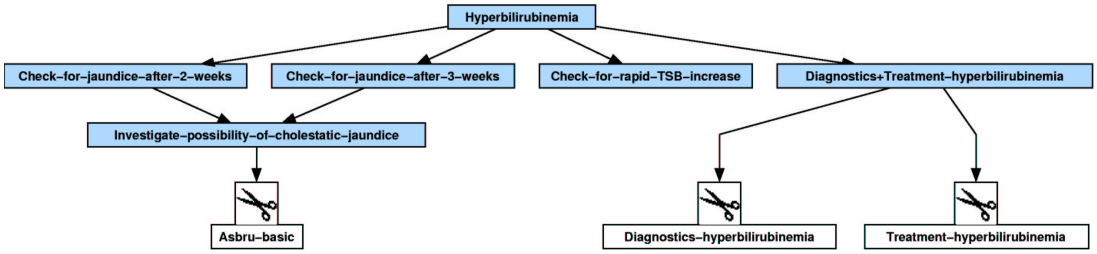


Figure 1: Top-level of Jaundice

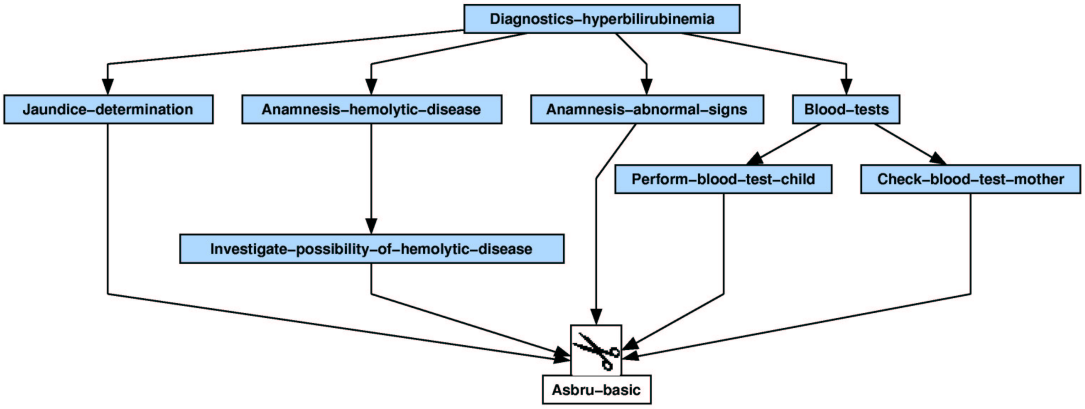


Figure 2: Diagnosis part of Jaundice

# B Example Protocols

## B.1 The Jaundice Protocol

### B.1.1 The Specification Graph

For better readability we have split the specification graph into several loosely coupled parts: Top-level (Fig. 1), Diagnosis (Fig. 2), Treatment (Fig. 3), and basic Asbru concepts (Fig. 4). The splits are indicated by a scissor symbol in the specification graphs.

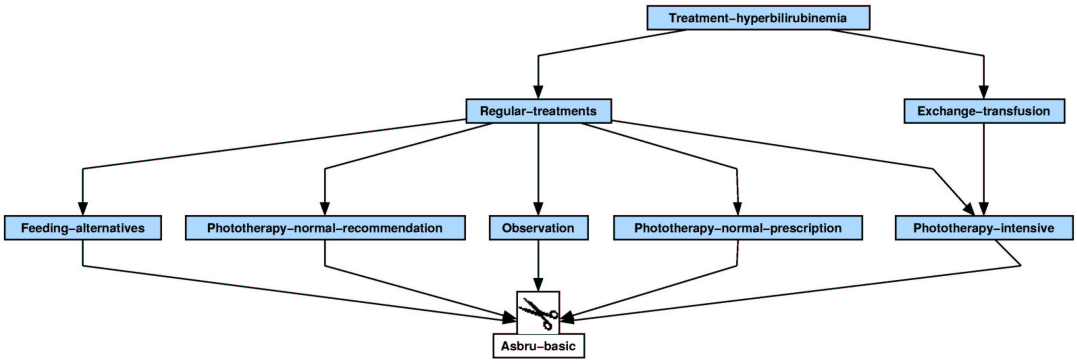


Figure 3: Treatment part of Jaundice

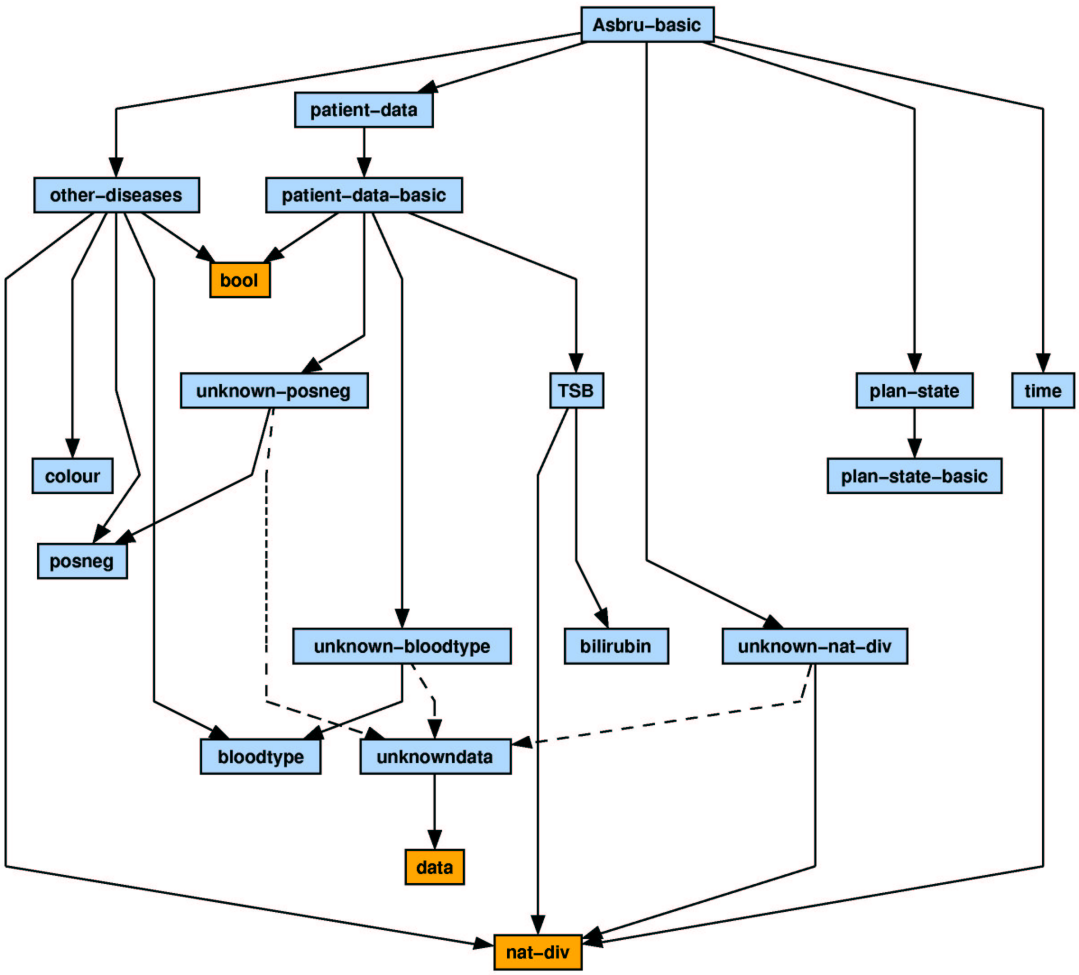


Figure 4: Basic Asbru parts of Jaundice

### B.1.2 The Specifications

— The following text was automatically generated from KIV. —

Hyperbilirubinemia, 15
Check-for-jaundice-after-2-weeks, 15
Check-for-jaundice-after-3-weeks, 16
Check-for-rapid-TSB-increase, 16
Diagnostics+Treatment-hyperbilirubinemia, 17
Diagnostics-hyperbilirubinemia, 18
Investigate-possibility-of-cholestatic-jaundice, 18
Treatment-hyperbilirubinemia, 19
Anamnesis-abnormal-signs, 20
Anamnesis-hemolytic-disease, 20
Blood-tests, 21
Exchange-transfusion, 22
Jaundice-determination, 22
Regular-treatments, 23
Check-blood-test-mother, 24
Feeding-alternatives, 25
Investigate-possibility-of-hemolytic-disease, 26
Observation, 26
Perform-blood-test-child, 26
Phototherapy-intensive, 27
Phototherapy-normal-prescription, 28
Phototherapy-normal-recommendation, 28
Asbru-basic, 29
other-diseases, 29
patient-data, 30
plan-state, 30
time, 31
unknown-nat-div, 31
colour, 31
patient-data-basic, 31
plan-state-basic, 31
TSB, 32
unknown-bloodtype, 32
unknown-posneg, 32
bilirubin, 32
bloodtype, 33
posneg, 33
unknowndata, 33
bool, 33
nat-div, 34
data, 34
nat-mult, 34
nat, 34
nat-basic2, 35
nat-basic1, 35

```
Hyperbilirubinemia =
enrich Diagnostics+Treatment-hyperbilirubinemia, Check-for-rapid-TSB-increase, Check-for-jaundice-after-2-weeks, Ch
    procedures hyperbilirubinemia#() : (patient-data, nat);
    variables time, time0, dt, dir-serum-bilirubin, tot-urine-bilirubin: nat;
    declaration

        hyperbilirubinemia# (patient-data, time)
        var var-possibility-of-hemolytic-disease = ?,
            var-possibility-of-cholestatic-disease = ?,
            jaundice-clinically-significant = ?,
            diagnostics+treatment-hyperbilirubinemia-state = inactive
        in
        break
              check-for-rapid-tsb-increase#(; patient-data,
                                              time,
                                              var-possibility-of-hemolytic-disease)
          ||s check-for-jaundice-after-2-weeks#(; patient-data,
                                                  jaundice-clinically-significant,
                                                  time,
                                                  var-possibility-of-cholestatic-disease)
          ||s check-for-jaundice-after-3-weeks#(; patient-data,
                                                  jaundice-clinically-significant,
                                                  time,
                                                  var-possibility-of-cholestatic-disease)
          ||s diagnostics+treatment-hyperbilirubinemia#(; patient-data,
                                                          time,
                                                          jaundice-clinically-significant,
                                                          diagnostics+treatment-
hyperbilirubinemia-state)
        if   diagnostics+treatment-hyperbilirubinemia-state[.] = completed
          ∨ var-possibility-of-hemolytic-disease[.]
end enrich
```

---

```
Check-for-jaundice-after-2-weeks =
enrich Investigate-possibility-of-cholestatic-jaundice with
    procedures
          check-for-jaundice-after-2-weeks#      ()   : (patient-data, bool, nat, bool);
          provide-routine-care-and-follow-up#    ()   : nat;
    variables
        time, time0, dt, dir-serum-bilirubin: nat;
        patient-data: patient-data;
        jaundice-clinically-significant, var-possibility-of-cholestatic-disease: bool;
    declaration

        check-for-jaundice-after-2-weeks# (patient-data, jaundice-clinically-significant, time,
var-possibility-of-cholestatic-disease)
        begin
          await jaundice-clinically-significant[.] ∧ patient-data[.].age > 336;
          var phy-exam-ok = ?, col-stools = ?, col-urine = ? in
          begin
            phy-exam-ok := ? ||a col-stools := ? ||a col-urine := ?;
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
            if possibility-of-cholestatic-disease(phy-exam-ok[.], col-stools[.], col-urine[.])
            then
```

```
            var dir-serum-bilirubin = ? in
            begin
              dir-serum-bilirubin := ?;
              var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
              patient-data := set-dir-serum-bilirubin(patient-data[.], dir-serum-bilirubin[.]);
              var-possibility-of-cholestatic-disease := true;
              investigate-possibility-of-cholestatic-jaundice#(; time)
            end
          else
            begin
              var-possibility-of-cholestatic-disease := false;
              provide-routine-care-and-follow-up#(; time)
            end
        end
    end;

    provide-routine-care-and-follow-up# (time)
    begin
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
    end
end enrich
```

---

```
Check-for-jaundice-after-3-weeks =
enrich Investigate-possibility-of-cholestatic-jaundice with
  procedures check-for-jaundice-after-3-weeks#() : (patient-data, bool, nat, bool);
  variables
    jaundice-clinically-significant, var-possibility-of-cholestatic-disease: bool;
    tsb-value, dir-serum-bilirubin, tot-urine-bilirubin: nat;
  declaration

    check-for-jaundice-after-3-weeks# (patient-data, jaundice-clinically-significant, time,
var-possibility-of-cholestatic-disease)
    begin
      await jaundice-clinically-significant[.] ∧ patient-data[.].age > 504;
      var tsb-value = ?, tsb = ?, dir-serum-bilirubin = ?, tot-urine-bilirubin = ? in
      begin
          ⟨tsb-value := ?;
            tsb := set-age-level(patient-data[.].tsb, patient-data[.].age, tsb-value[.])⟩
        ||a dir-serum-bilirubin := ?
        ||a tot-urine-bilirubin := ?;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        patient-data := set-tot-urine-bilirubin(set-dir-serum-bilirubin(set-tsb(patient-data[.],
tsb[.]), dir-serum-bilirubin[.]), tot-urine-bilirubin[.])
      end;
      var-possibility-of-cholestatic-disease := true;
      investigate-possibility-of-cholestatic-jaundice#(; time)
    end
end enrich
```

---

```
Check-for-rapid-TSB-increase =
enrich Investigate-possibility-of-hemolytic-disease with
```

```
    procedures
        check-for-rapid-tsb-increase#                          ()   : (patient-data, nat, bool);
        investigate-possibility-of-g6pd-hemolytic-disease#     ()   : nat;
    variables
        time, time0, dt: nat;
        patient-data: patient-data;
        var-possibility-of-hemolytic-disease: bool;
    declaration

      check-for-rapid-tsb-increase# (patient-data, time, var-possibility-of-hemolytic-disease)
      begin
        await ¬ get-decrease(patient-data[.].tsb) ∧ get-change(patient-data[.].tsb) > 10;
        var-possibility-of-hemolytic-disease := true;
        if possibility-of-g6pd(patient-data[.].age) then
          investigate-possibility-of-g6pd-hemolytic-disease#(; time)
        else
          investigate-possibility-of-hemolytic-disease#(; time)
      end;

      investigate-possibility-of-g6pd-hemolytic-disease# (time)
      begin
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end
end enrich
```

---

```
Diagnostics+Treatment-hyperbilirubinemia =
enrich Treatment-hyperbilirubinemia, Diagnostics-hyperbilirubinemia with
    procedures diagnostics+treatment-hyperbilirubinemia#() : (patient-data, nat, bool, plan-state);
    variables
        time, time0, dt: nat;
        diagnostics+treatment-hyperbilirubinemia-state: plan-state;
        term: bool;
    declaration

      diagnostics+treatment-hyperbilirubinemia# (patient-data, time,
jaundice-clinically-significant, diagnostics+treatment-hyperbilirubinemia-state)
      var pathologic-reason = ?,
          treatment-hyperbilirubinemia-state = inactive,
          term = ?,
          age = ?
      in
      begin
        break
          term := ?;
          age := ?;
          var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
          patient-data := set-age(set-term(patient-data[.], term[.]), age[.]);
          diagnostics-hyperbilirubinemia#(; patient-data,
                                            time,
                                            jaundice-clinically-significant,
                                            pathologic-reason);
          treatment-hyperbilirubinemia#(; patient-data,
                                          time,
```

```
                                                    treatment-hyperbilirubinemia-state)
        if   ¬ jaundice-clinically-significant[.]
           ∨ treatment-hyperbilirubinemia-state[.] = completed
           ∨ ¬ term[.]
           ∨ age[.] ≤ 24
           ∨ pathologic-reason[.];
        if   ¬ jaundice-clinically-significant[.]
           ∨ treatment-hyperbilirubinemia-state[.] = completed
        then
           diagnostics+treatment-hyperbilirubinemia-state := completed
        else
           diagnostics+treatment-hyperbilirubinemia-state := aborted
     end
end enrich
```

---

```
Diagnostics-hyperbilirubinemia =
enrich Anamnesis-abnormal-signs, Blood-tests, Anamnesis-hemolytic-disease, Jaundice-determination with
   procedures diagnostics-hyperbilirubinemia#() : (patient-data, nat, bool, bool);
   variables time, time0, dt: nat;
   declaration

      diagnostics-hyperbilirubinemia# (patient-data, time, jaundice-clinically-significant,
pathologic-reason)
      var anamnesis-abnormal-signs-state = inactive,
          blood-tests-state = inactive,
          anamnesis-hemolytic-disease-state = inactive
      in
      break
        pathologic-reason := false;
        anamnesis-abnormal-signs#(; time, pathologic-reason, anamnesis-abnormal-signs-state);
        blood-tests#(; patient-data, time, pathologic-reason, blood-tests-state);
        anamnesis-hemolytic-disease#(; time,
                                       pathologic-reason,
                                       anamnesis-hemolytic-disease-state);
        jaundice-determination#(; time, jaundice-clinically-significant)
      if   anamnesis-abnormal-signs-state[.] = aborted
        ∨ blood-tests-state[.] = aborted
        ∨ anamnesis-hemolytic-disease-state[.] = aborted
end enrich
```

---

```
Investigate-possibility-of-cholestatic-jaundice =
enrich Asbru-basic with
   procedures investigate-possibility-of-cholestatic-jaundice#() : nat;
   variables time, time0, dt: nat;
   declaration

      investigate-possibility-of-cholestatic-jaundice# (time)
      begin
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end
end enrich
```

---

```
Treatment-hyperbilirubinemia =
enrich Regular-treatments, Exchange-transfusion with
    procedures treatment-hyperbilirubinemia#() : (patient-data, nat, plan-state);
    variables
        time, time0, dt, tsb-value, age: nat;
        treatment-hyperbilirubinemia-state: plan-state;
        tsb: tsb;
    declaration

      treatment-hyperbilirubinemia# (patient-data, time, treatment-hyperbilirubinemia-state)
      var regular-treatments-state = inactive,
          exchange-transfusion-state = inactive,
          phototherapy-intensive-activated = ⊥
      in
      begin
        break
            break
                begin
                  regular-treatments#(; patient-data,
                                        time,
                                        regular-treatments-state,
                                        phototherapy-intensive-activated);
                  if regular-treatments-state = aborted then
                    exchange-transfusion#(; phototherapy-intensive-activated,
                                            patient-data,
                                            time,
                                            exchange-transfusion-state)
                end
              ||a exchange-transfusion#(; phototherapy-intensive-activated,
                                          patient-data,
                                          time,
                                          exchange-transfusion-state)
            if   regular-treatments-state[.] = completed
               ∨ exchange-transfusion-state[.] = completed
          ||s while true do
                begin
                  var time0 = time[.] in
                  await 12 ≤ time[.] - time0[.] ∧ time[.] - time0[.] ≤ 24;
                  var tsb-value = ?, age = ?, tsb = ? in
                  begin
                    age := ?;
                    ⟨tsb-value := ?;
                     tsb := set-age-level(patient-data[.].tsb, age[.], tsb-value[.])⟩;
                    var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
                    patient-data := set-tsb(set-age(patient-data[.], age[.]), tsb[.])
                  end
                end
        if   regular-treatments-state[.] = completed
           ∨ exchange-transfusion-state[.] = completed;
        if   regular-treatments-state[.] = completed
           ∨ exchange-transfusion-state[.] = completed
        then
          treatment-hyperbilirubinemia-state := completed
      end
```

**end enrich**

---

Anamnesis-abnormal-signs =
**enrich** Asbru-basic **with**
**procedures** anamnesis-abnormal-signs#() : (nat, bool, plan-state);
**variables**
time, $time_0$, dt: nat;
pathologic-reason, let, apn, tac, ins-temperature, beh-changes, hep, vom, fee-difficulty, exc-weight-loss: bool;
anamnesis-abnormal-signs-state: plan-state;
**declaration**

anamnesis-abnormal-signs# (time, pathologic-reason, anamnesis-abnormal-signs-state)
**begin**
**break**
**var** let = ?,
apn = ?,
tac = ?,
ins-temperature = ?,
beh-changes = ?,
hep = ?,
vom = ?,
fee-difficulty = ?,
exc-weight-loss = ?
**in**
**begin**
let := ?
$||_a$ apn := ?
$||_a$ tac := ?
$||_a$ ins-temperature := ?
$||_a$ beh-changes := ?
$||_a$ hep := ?
$||_a$ vom := ?
$||_a$ fee-difficulty := ?
$||_a$ exc-weight-loss := ?;
**var** $time_0$ = time[.], dt = ? **in await** $time_0$[.] + dt[.] $\leq$ time[.];
pathologic-reason := possibility-of-other-diseases(let, apn, tac, ins-temperature, beh-changes, hep, vom, fee-difficulty, exc-weight-loss)
**end**
**if** pathologic-reason[.];
**if** pathologic-reason[.] **then** anamnesis-abnormal-signs-state := aborted
**end**
**end enrich**

---

Anamnesis-hemolytic-disease =
**enrich** Investigate-possibility-of-hemolytic-disease **with**
**procedures** anamnesis-hemolytic-disease#() : (nat, bool, plan-state);
**variables**
time, $time_0$, dt: nat;
pathologic-reason, fam-history, eth-origin, geo-origin, ear-jaundice, pallor, hepatosplenomegaly, var-possibility-of
anamnesis-hemolytic-disease-state: plan-state;
**declaration**

anamnesis-hemolytic-disease# (time, pathologic-reason, anamnesis-hemolytic-disease-state)

```
        begin
          break
            var fam-history = ?,
                eth-origin = ?,
                geo-origin = ?,
                ear-jaundice = ?,
                pallor = ?,
                hepatosplenomegaly = ?,
                var-possibility-of-hemolytic-disease = ?
            in
            begin
                fam-history := ?
              ||a eth-origin := ?
              ||a geo-origin := ?
              ||a ear-jaundice := ?
              ||a pallor := ?
              ||a hepatosplenomegaly := ?;
              var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
              if possibility-of-hemolytic-disease(fam-history[.], eth-origin[.], geo-origin[.],
ear-jaundice[.])
                then
                  begin
                    investigate-possibility-of-hemolytic-disease#(; time);
                    var-possibility-of-hemolytic-disease := ?;
                    var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
                  end
                else
                  var-possibility-of-hemolytic-disease := false;
                pathologic-reason :=    var-possibility-of-hemolytic-disease
                                      ∨ possibility-of-inherited-disease(eth-origin[.])
            end
          if pathologic-reason[.];
          if pathologic-reason[.] then anamnesis-hemolytic-disease-state := aborted
        end
end enrich
```

---

```
Blood-tests =
enrich Perform-blood-test-child, Check-blood-test-mother with
    procedures blood-tests#() : (patient-data, nat, bool, plan-state);
    variables
        time, time0, dt: nat;
        pathologic-reason: bool;
        blood-tests-state: plan-state;
        blo-mother: bloodtype;
        rhe-mother, ant-mother: posneg;
    declaration

      blood-tests# (patient-data, time, pathologic-reason, blood-tests-state)
      var check-blood-test-mother-state = inactive, perform-blood-test-child-state = inactive in
      begin
        break
          check-blood-test-mother#(; patient-data, time, check-blood-test-mother-state);
          perform-blood-test-child#(; patient-data,
```

```
                                        time,
                                        pathologic-reason,
                                        perform-blood-test-child-state)
        if      check-blood-test-mother-state[.].failed
              ∧ perform-blood-test-child-state[.].failed
            ∨   check-blood-test-mother-state[.] = completed
              ∧ perform-blood-test-child-state[.].terminated
            ∨   check-blood-test-mother-state[.].terminated
              ∧ perform-blood-test-child-state[.] = completed;
        if    check-blood-test-mother-state[.].failed
            ∧ perform-blood-test-child-state[.].failed
        then
          blood-tests-state := aborted
      end
end enrich
```

---

```
Exchange-transfusion =
enrich Phototherapy-intensive with
    procedures
        exchange-transfusion#              ()   : (unknown-nat, patient-data, nat, plan-state);
        prescribe-exchange-transfusion#    ()   : nat;
    variables
        time, time0, dt: nat;
        exchange-transfusion-state: plan-state;
    declaration

      exchange-transfusion# (phototherapy-intensive-activated, patient-data, time,
exchange-transfusion-state)
      begin
        await   get-bilirubin(patient-data[.].tsb) = transfusion
              ∨   get-bilirubin(patient-data[.].tsb) = phototherapy-intensive
                ∧ phototherapy-intensive-activated[.] ≠ ⊥
                ∧ (     4 ≤ time[.] - phototherapy-intensive-activated[.].value
                      ∧ ¬ get-decrease(patient-data.tsb)
                    ∨   4 ≤ time[.] - phototherapy-intensive-activated[.].value
                      ∧ time[.] - phototherapy-intensive-activated[.].value ≤ 6
                      ∧ get-decrease(patient-data[.].tsb)
                      ∧ get-change(patient-data[.].tsb) < 10);
        prescribe-intensive-phototherapy#(; time) ||s prescribe-exchange-transfusion#(; time);
        exchange-transfusion-state := completed
      end;

      prescribe-exchange-transfusion# (time)
      begin
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end
end enrich
```

---

```
Jaundice-determination =
enrich Asbru-basic with
    procedures
```

```
        jaundice-determination#                                  ()  : (nat, bool);
        blanching-skin-with-digital-pressure-test-manual#        ()  : nat;
        icterometer-test-manual#                                 ()  : nat;
        transcutaneous-jaundice-meter-test-manual#               ()  : nat;
        determine-extent-cephalocaudad-progression-manual#       ()  : nat;
    variables
        time, time0, dt: nat;
        jaundice-clinically-significant: bool;
    declaration

      jaundice-determination# (time, jaundice-clinically-significant)
      begin
            blanching-skin-with-digital-pressure-test-manual#(; time)
        OR icterometer-test-manual#(; time)
        OR transcutaneous-jaundice-meter-test-manual#(; time)
        OR determine-extent-cephalocaudad-progression-manual#(; time);
        jaundice-clinically-significant := ?;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end;

      blanching-skin-with-digital-pressure-test-manual# (time)
      begin
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end;

      icterometer-test-manual# (time)
      begin
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end;

      transcutaneous-jaundice-meter-test-manual# (time)
      begin
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end;

      determine-extent-cephalocaudad-progression-manual# (time)
      begin
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end
end enrich
```

---

```
Regular-treatments =
enrich Phototherapy-intensive, Phototherapy-normal-prescription, Phototherapy-normal-recommendation, Observation
    procedures regular-treatments#() : (patient-data, nat, plan-state, unknown-nat);
    variables regular-treatments-state: plan-state;
    declaration
```

```
        regular-treatments# (patient-data, time, regular-treatments-state,
phototherapy-intensive-activated)
        var phototherapy-normal-prescription-activated = ⊥ in
        begin
          await get-bilirubin(patient-data[.].tsb) ≠ transfusion;
          break
                feeding-alternatives#(; time)
            ||s var observation-state = inactive in
                break
                  while true do
                          phototherapy-intensive#(; phototherapy-normal-prescription-activated,
                                                    patient-data,
                                                    time,
                                                    phototherapy-intensive-activated)
                      OR phototherapy-normal-prescription#(; patient-data,
                                                             time,
                                                             phototherapy-normal-
prescription-activated)
                      OR phototherapy-normal-recommendation#(; patient-data,
                                                               time)
                      OR observation#(; patient-data,
                                        time,
                                        observation-state)
                if observation-state[.] = completed
          if    get-bilirubin(patient-data[.].tsb) = transfusion
            ∨    get-bilirubin(patient-data[.].tsb) = phototherapy-intensive
              ∧ phototherapy-intensive-activated[.] ≠ ⊥
              ∧ (      4 ≤ time[.] - phototherapy-intensive-activated[.].value
                    ∧ time[.] - phototherapy-intensive-activated[.].value ≤ 6
                    ∧ get-decrease(patient-data[.].tsb)
                    ∧ get-change(patient-data[.].tsb) < 10
                  ∨    4 ≤ time[.] - phototherapy-intensive-activated[.].value
                    ∧ ¬ get-decrease(patient-data[.].tsb))
        end
end enrich
```

---

```
Check-blood-test-mother =
enrich Asbru-basic with
    procedures
          check-blood-test-mother#        ()   : (patient-data, nat, plan-state);
          consider-holding-cord-blood#    ()   : nat;
    variables
        time, time0, dt: nat;
        patient-data: patient-data;
        check-blood-test-mother-state: plan-state;
    declaration

        check-blood-test-mother# (patient-data, time, check-blood-test-mother-state)
        if    patient-data[.].blo-mother ≠ ⊥
          ∧ patient-data[.].rhe-mother ≠ ⊥
          ∧ patient-data[.].ant-mother ≠ ⊥
        then
          begin
```

```
            if consider-holding-cord-blood(patient-data[.].blo-mother.value,
patient-data[.].rhe-mother.value, patient-data[.].ant-mother.value)
            then
               consider-holding-cord-blood#(; time);
            check-blood-test-mother-state := completed
         end
      else
         check-blood-test-mother-state := rejected;

      consider-holding-cord-blood# (time)
      begin
         skip;
         var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end
end enrich
```

---

```
Feeding-alternatives =
enrich Asbru-basic with
   procedures
         feeding-alternatives#                  ()   : nat;
         breastfeeding-manual#                  ()   : nat;
         breastfeeding-with-formula-manual#     ()   : nat;
         formula-only-manual#                   ()   : nat;
   variables
      time, time0, dt: nat;
      breastfed-child: bool;
   declaration

      feeding-alternatives# (time)
      var breastfed-child = ? in
      begin
         breastfed-child := ?;
         if breastfed-child[.] then
                breastfeeding-manual#(; time)
            OR breastfeeding-with-formula-manual#(; time)
            OR formula-only-manual#(; time)
      end;

      breastfeeding-manual# (time)
      begin
         var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
         skip;
         var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end;

      breastfeeding-with-formula-manual# (time)
      begin
         var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
         skip;
         var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end;

      formula-only-manual# (time)
```

```
    begin
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
    end
end enrich
```

---

```
Investigate-possibility-of-hemolytic-disease =
enrich Asbru-basic with
  procedures investigate-possibility-of-hemolytic-disease#() : nat;
  variables time, time0, dt: nat;
  declaration

    investigate-possibility-of-hemolytic-disease# (time)
    begin
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
    end
end enrich
```

---

```
Observation =
enrich Asbru-basic with
  procedures
      observation#              ()   : (patient-data, nat, plan-state);
      prescribe-observation#    ()   : nat;
  variables
      time, time0, dt: nat;
      patient-data: patient-data;
      observation-state: plan-state;
  declaration

    observation# (patient-data, time, observation-state)
    begin
      await get-bilirubin(patient-data[.].tsb) = observation;
      break
        prescribe-observation#(; time); observation-state := completed
      if get-bilirubin(patient-data[.].tsb) ≠ observation
    end;

    prescribe-observation# (time)
    begin
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
    end
end enrich
```

---

```
Perform-blood-test-child =
enrich Asbru-basic with
  procedures perform-blood-test-child#() : (patient-data, nat, bool, plan-state);
  variables
      time, time0, dt: nat;
      patient-data: patient-data;
```

```
        pathologic-reason: bool;
        perform-blood-test-child-state: plan-state;
        blo-child: bloodtype;
        rhe-child, dir-coombs-test: posneg;
    declaration

      perform-blood-test-child# (patient-data, time, pathologic-reason,
perform-blood-test-child-state)
      if   patient-data[.].blo-mother = ⊥
        ∨ patient-data[.].rhe-mother = ⊥
        ∨ patient-data[.].ant-mother = ⊥
        ∨ ¬ consider-holding-cord-blood(patient-data[.].blo-mother.value,
patient-data[.].rhe-mother.value, patient-data[.].ant-mother.value)
      then
        begin
          break
            var blo-child = ?, rhe-child = ?, dir-coombs-test = ? in
            begin
              blo-child := ? ||a rhe-child := ? ||a dir-coombs-test := ?;
              var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
              pathologic-reason :=
possibility-of-isoimmune-hemolytic-disease(dir-coombs-test)
            end
          if pathologic-reason[.];
          if pathologic-reason[.] then perform-blood-test-child-state := aborted else
            perform-blood-test-child-state := completed
        end
      else
        perform-blood-test-child-state := rejected
end enrich
```

---

```
Phototherapy-intensive =
enrich Asbru-basic with
    procedures
        phototherapy-intensive#              ()   : (unknown-nat, patient-data, nat, unknown-nat);
        prescribe-intensive-phototherapy#    ()   : nat;
    variables phototherapy-intensive-activated, phototherapy-normal-prescription-activated: unknown-nat;
    declaration

      phototherapy-intensive# (phototherapy-normal-prescription-activated, patient-data, time,
phototherapy-intensive-activated)
      begin
        await   get-bilirubin(patient-data[.].tsb) = phototherapy-intensive
              ∨   get-bilirubin(patient-data[.].tsb) = phototherapy-normal
                ∧ phototherapy-normal-prescription-activated[.] ≠ ⊥
                ∧ 4 ≤ time[.] - phototherapy-normal-prescription-activated[.].value
                ∧ ¬ get-decrease(patient-data[.].tsb);
        phototherapy-intensive-activated := mk-value(time[.]);
        break
          prescribe-intensive-phototherapy#(; time)
        if   get-bilirubin(patient-data[.].tsb) ≠ phototherapy-intensive
          ∨   get-bilirubin(patient-data[.].tsb) = phototherapy-intensive
            ∧ (     4 ≤ time[.] - phototherapy-intensive-activated[.].value
                  ∧ time[.] - phototherapy-intensive-activated[.].value ≤ 6
```

```
                    ∧ get-decrease(patient-data[.].tsb)
                    ∧ get-change(patient-data[.].tsb) < 10
                ∨   4 ≤ time[.] - phototherapy-intensive-activated[.].value
                    ∧ ¬ get-decrease(patient-data[.].tsb))
    end;

    prescribe-intensive-phototherapy# (time)
    begin
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
    end
end enrich
```

---

```
Phototherapy-normal-prescription =
enrich Asbru-basic with
  procedures
      phototherapy-normal-prescription#   ()   : (patient-data, nat, unknown-nat);
      prescribe-normal-phototherapy#      ()   : nat;
  variables phototherapy-normal-prescription-activated: unknown-nat;
  declaration

    phototherapy-normal-prescription# (patient-data, time,
phototherapy-normal-prescription-activated)
    var phototherapy-normal-prescription-activated = ⊥ in
    begin
      await get-bilirubin(patient-data[.].tsb) = phototherapy-normal;
      phototherapy-normal-prescription-activated := mk-value(time[.]);
      break
        prescribe-normal-phototherapy#(; time)
      if   get-bilirubin(patient-data[.].tsb) ≠ phototherapy-normal
        ∨    get-bilirubin(patient-data[.].tsb) = phototherapy-normal
          ∧ 4 ≤ time[.] - phototherapy-normal-prescription-activated[.].value
          ∧ ¬ get-decrease(patient-data[.].tsb)
    end;

    prescribe-normal-phototherapy# (time)
    begin
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
    end
end enrich
```

---

```
Phototherapy-normal-recommendation =
enrich Asbru-basic with
  procedures
      phototherapy-normal-recommendation#   ()   : (patient-data, nat);
      prescribe-observation-manual#         ()   : nat;
      prescribe-normal-phototherapy-manual# ()   : nat;
  declaration

    phototherapy-normal-recommendation# (patient-data, time)
    begin
      await get-bilirubin(patient-data[.].tsb) = phototherapy-recommendation;
```

**break**
prescribe-observation-manual#(**;** time) **OR**
prescribe-normal-phototherapy-manual#(**;** time)
**if** get-bilirubin(patient-data[.].tsb) ≠ phototherapy-recommendation
**end**;

prescribe-observation-manual# (time)
**begin**
**var** $time_0$ = time[.], dt = ? **in await** $time_0$[.] + dt[.] ≤ time[.];
**skip**;
**var** $time_0$ = time[.], dt = ? **in await** $time_0$[.] + dt[.] ≤ time[.]
**end**;

prescribe-normal-phototherapy-manual# (time)
**begin**
**var** $time_0$ = time[.], dt = ? **in await** $time_0$[.] + dt[.] ≤ time[.];
**skip**;
**var** $time_0$ = time[.], dt = ? **in await** $time_0$[.] + dt[.] ≤ time[.]
**end**
**end enrich**

---

Asbru-basic = other-diseases + patient-data + unknown-nat-div + plan-state + time

---

other-diseases =
**enrich** bool, nat-div, posneg, bloodtype, colour **with**
**predicates**

| | | |
|---|---|---|
| possibility-of-other-diseases | : | bool × bool × bool × bool × bool × bool × bool × bool × |
| consider-holding-cord-blood | : | bloodtype × posneg × posneg; |
| possibility-of-isoimmune-hemolytic-disease | : | posneg; |
| possibility-of-hemolytic-disease | : | bool × bool × bool × bool; |
| possibility-of-inherited-disease | : | bool; |
| possibility-of-g6pd | : | nat; |
| possibility-of-cholestatic-disease | : | bool × colour × colour; |

**variables**
let, apn, tac, ins-temperature, beh-changes, hep, vom, fee-difficulty, exc-weight-loss, fam-history, eth-origin, geo-
blo-mother: bloodtype;
rhe-mother, ant-mother, dir-coombs-test: posneg;
age: nat;
col-stools, col-urine: colour;

**axioms**

let
∨ apn
∨ tac
∨ ins-temperature
∨ beh-changes
∨ hep
∨ vom
∨ fee-difficulty
∨ exc-weight-loss
↔ possibility-of-other-diseases(let, apn, tac, ins-temperature, beh-changes, hep, vom, fee-difficulty, exc-weight-loss);

blo-mother = o-type ∨ rhe-mother = pos ∧ ant-mother = neg
↔ consider-holding-cord-blood(blo-mother, rhe-mother, ant-mother);

dir-coombs-test = pos ↔ possibility-of-isoimmune-hemolytic-disease(dir-coombs-test);

fam-history ∨ eth-origin ∨ geo-origin ∨ ear-jaundice
↔ possibility-of-hemolytic-disease(fam-history, eth-origin, geo-origin, ear-jaundice);

eth-origin ↔ possibility-of-inherited-disease(eth-origin);

25 ≤ age ∧ age ≤ 48 ↔ possibility-of-g6pd(age);

phy-exam-ok ≠ true ∨ col-stools = light ∨ col-urine = dark
↔ possibility-of-cholestatic-disease(phy-exam-ok, col-stools, col-urine);

**end enrich**

---

patient-data =
**enrich** patient-data-basic **with**
**functions**

| | | | | | |
|---|---|---|---|---|---|
| set-age | : | patient-data × nat | → | patient-data | ; |
| set-term | : | patient-data × bool | → | patient-data | ; |
| set-tsb | : | patient-data × tsb | → | patient-data | ; |
| set-dir-serum-bilirubin | : | patient-data × nat | → | patient-data | ; |
| set-tot-urine-bilirubin | : | patient-data × nat | → | patient-data | ; |

**variables** a, $a_1$, $a_2$: nat;

**axioms**

set-age(pd, a)
= mk-patient-data(a, pd.term, pd.tsb, pd.dir-serum-bilirubin, pd.tot-urine-bilirubin,
pd.blo-mother, pd.rhe-mother, pd.ant-mother);

set-term(pd, boolvar)
= mk-patient-data(pd.age, boolvar, pd.tsb, pd.dir-serum-bilirubin,
pd.tot-urine-bilirubin, pd.blo-mother, pd.rhe-mother, pd.ant-mother);

set-tsb(pd, tsb)
= mk-patient-data(pd.age, pd.term, tsb, pd.dir-serum-bilirubin,
pd.tot-urine-bilirubin, pd.blo-mother, pd.rhe-mother, pd.ant-mother);

set-dir-serum-bilirubin(pd, a)
= mk-patient-data(pd.age, pd.term, pd.tsb, a, pd.tot-urine-bilirubin, pd.blo-mother,
pd.rhe-mother, pd.ant-mother);

set-tot-urine-bilirubin(pd, a)
= mk-patient-data(pd.age, pd.term, pd.tsb, pd.dir-serum-bilirubin, a, pd.blo-mother,
pd.rhe-mother, pd.ant-mother);

**end enrich**

---

plan-state =
**enrich** plan-state-basic **with**
**predicates**

| | | |
|---|---|---|
| . .failed | : | plan-state; |
| . .terminated | : | plan-state; |

**axioms**

ps.failed ↔ ps = rejected ∨ ps = aborted;

ps.terminated ↔ ps = rejected ∨ ps = aborted ∨ ps = completed;

**end enrich**

---

time =
**enrich** nat-div **with**
**variables**
time, $time_0$, dt: nat;
$\varphi$, $\varphi_1$, $\varphi_2$, $\varphi_3$: bool;

**end enrich**

---

unknown-nat-div =
**actualize** unknowndata **with** nat-div **by morphism**
unknowndata → unknown-nat; data → nat; ud → un; $ud_0$ → $un_0$
**end actualize**

---

colour =
**data specification**
colour = light
| dark
;
**variables** col, $col_1$, $col_2$: colour;
**end data specification**

---

patient-data-basic =
**data specification**
**using** bool, TSB, unknown-bloodtype, unknown-posneg
patient-data = mk-patient-data (. .age : nat ;
. .term : bool ;
. .tsb : tsb ;
. .dir-serum-bilirubin : nat ;
. .tot-urine-bilirubin : nat ;
. .blo-mother : unknown-bloodtype ;
. .rhe-mother : unknown-posneg ;
. .ant-mother : unknown-posneg ;
);
**variables** patient-data, pd, $pd_0$, $pd_1$, $pd_2$: patient-data;
**end data specification**

---

plan-state-basic =
**data specification**
plan-state = inactive
| considered
| rejected
| suspended
| completed
| aborted

;
**variables** ps, $ps_1$, $ps_2$, $ps_3$: plan-state;
**end data specification**

---

TSB =
**enrich** nat-div, bilirubin **with**
**sorts** tsb;
**functions**

| | | | | | |
|---|---|---|---|---|---|
| set-age-level | : | tsb × nat × nat | → | tsb | ; |
| get-bilirubin | : | tsb | → | bilirubin | ; |
| get-change | : | tsb | → | nat | ; |

**predicates** get-decrease : tsb;
**variables**
tsb, $tsb_1$, $tsb_2$, $tsb_3$: tsb;
tsb-value, $tsb\text{-}value_1$, $tsb\text{-}value_2$, $tsb\text{-}value_3$, a, $a_1$, $a_2$, $a_3$, l, $l_1$, $l_2$, $l_3$: nat;

**axioms**

$25 \leq a \wedge a \leq 48 \wedge l < 12 \rightarrow$ get-bilirubin(set-age-level(tsb, a, l)) = observation;

$25 \leq a \wedge a \leq 48 \wedge 12 \leq l \wedge l < 15$
$\rightarrow$ get-bilirubin(set-age-level(tsb, a, l)) = phototherapy-normal;

$25 \leq a \wedge a \leq 48 \wedge 15 \leq l \wedge l < 20$
$\rightarrow$ get-bilirubin(set-age-level(tsb, a, l)) = phototherapy-recommendation;

$25 \leq a \wedge a \leq 48 \wedge 20 \leq l \wedge l < 25$
$\rightarrow$ get-bilirubin(set-age-level(tsb, a, l)) = phototherapy-intensive;

$25 \leq a \wedge a \leq 48 \wedge 25 \leq l \wedge l < 100 \rightarrow$ get-bilirubin(set-age-level(tsb, a, l)) = transfusion;

$25 \leq a \wedge a \leq 48 \wedge 100 \leq l \rightarrow$ get-bilirubin(set-age-level(tsb, a, l)) = toxic;

$l_1 < l_2 \vdash$ get-decrease(set-age-level(set-age-level(tsb, $a_1$, $l_1$), $a_2$, $l_2$));

$\neg\, l_1 < l_2 \rightarrow \neg$ get-decrease(set-age-level(set-age-level(tsb, $a_1$, $l_1$), $a_2$, $l_2$));

get-change(set-age-level(set-age-level(tsb, $a_1$, $l_1$), $a_2$, $l_2$))
= ($l_2 > l_1 \supset l_2$ - $l_1$; $l_1$ - $l_2$) $*$ 10 / ($a_2$ - $a_1$);

**end enrich**

---

unknown-bloodtype =
**actualize** unknowndata **with** bloodtype **by morphism**
unknowndata → unknown-bloodtype; data → bloodtype; d → bt; $d_0 \rightarrow bt_0$;
$d_1 \rightarrow bt_1$; $d_2 \rightarrow bt_2$; ud → ubt; $ud_0 \rightarrow ubt_0$
**end actualize**

---

unknown-posneg =
**actualize** unknowndata **with** posneg **by morphism**
unknowndata → unknown-posneg; data → posneg; d → pn; $d_0 \rightarrow pn_0$; $d_1 \rightarrow$
$pn_1$; $d_2 \rightarrow pn_2$; ud → upn; $ud_0 \rightarrow upn_0$
**end actualize**

---

bilirubin =
**data specification**
bilirubin = observation

| phototherapy-normal
| phototherapy-recommendation
| phototherapy-intensive
| transfusion
| toxic
;
**variables** bili, $bili_1$, $bili_2$, $bili_3$: bilirubin;
**end data specification**

---

bloodtype =
**data specification**
bloodtype = a-type
| b-type
| o-type
| ab-type
;
**variables** bt, $bt_1$, $bt_2$: bloodtype;
**end data specification**

---

posneg =
**data specification**
posneg = pos
| neg
;
**variables** pn, $pn_1$, $pn_2$: posneg;
**end data specification**

---

unknowndata =
**generic data specification**
**parameter** data
unknowndata = ⊥
| mk-value (. .value : data ;)
;
**variables** ud, $ud_0$: unknowndata;
**end generic data specification**

---

bool =
**data specification**
bool = true
| false
;
**variables** boolvar, $boolvar_0$: bool;
**end data specification**

---

nat-div =
**enrich** nat-mult **with**
**functions**
. / . : nat × nat → nat **prio** 11;
. % . : nat × nat → nat **prio** 11;

**axioms**

Divdef : n ≠ 0 → m / n ∗ n ≤ m ∧ m < (m / n)+1 ∗ n;

Moddef : n ≠ 0 → m = m / n ∗ n + m % n;

**end enrich**

---

data =
**specification**
**sorts** data;
**variables** d, $d_0$, $d_1$, $d_2$: data;
**end specification**

---

nat-mult =
**enrich** nat **with**
**functions** . ∗ . : nat × nat → nat **prio** 10;

**axioms**

m ∗ 0 = 0;

m ∗ n +1 = m ∗ n + m;

**end enrich**

---

nat =
**enrich** nat-basic2 **with**
**functions** . - . : nat × nat → nat **prio** 8 **left**;
**predicates**
. ≤ . : nat × nat;
. > . : nat × nat;
. ≥ . : nat × nat;

**axioms**

m - 0 = m;

m - n +1 = (m - n)-1;

m ≤ n ↔ ¬ n < m;

m > n ↔ n < m;

m ≥ n ↔ ¬ m < n;

**end enrich**

---

nat-basic2 =
**enrich** nat-basic1 **with**
**functions** . + . : nat × nat → nat **prio** 9;
**variables** m, $n_0$: nat;

**axioms**

$n + 0 = n$;

$m + n + 1 = (m + n) + 1$;

$n < n_0 \vee n = n_0 \vee n_0 < n$;

$1 = 0 + 1$;

$0 \neq 1$;

**end enrich**

---

nat-basic1 =
**data specification**
nat = 0
| . +1 (. -1 : nat ;)
;
**variables** n: nat;
**order predicates** . < . : nat × nat;
**end data specification**

---

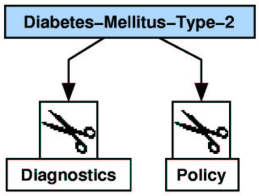


Figure 5: Top-level of Diabetes

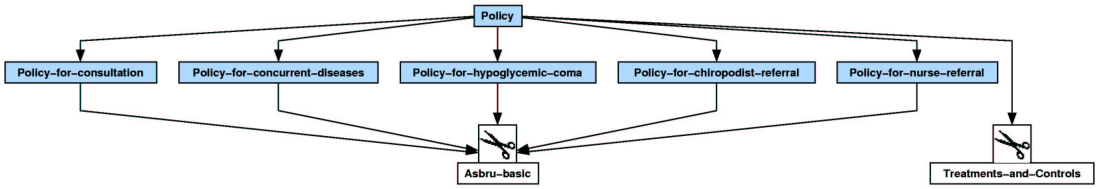


Figure 6: Policy part of Diabetes

## B.2 The Diabetes Protocol

### B.2.1 The Specification Graph

For better readability we have split the specification graph (Fig. 5) into several loosely coupled parts: a diagnostics part (Fig. 9), a policy part (Fig. 6) and a part for treatment (Fig. 10) and annual control (Fig. 7). Basic Asbru concepts, which are needed in almost all specifications, were put into a separate graph as well (Fig. 8). The splits are indicated by a scissor symbol in the specification graphs.

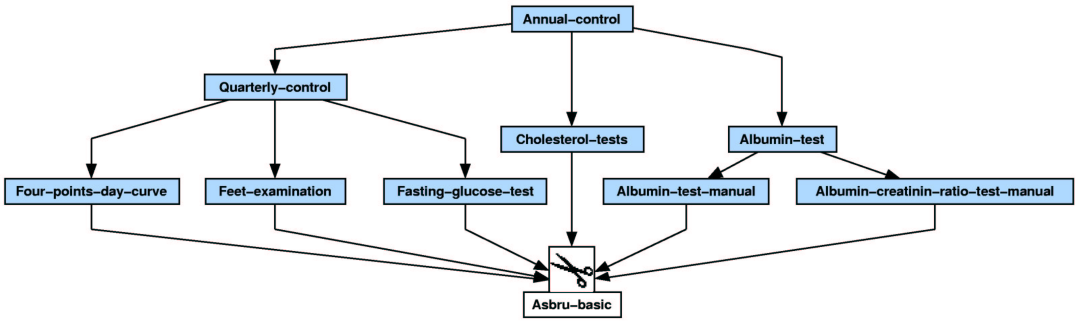


Figure 7: Annual Control part of Treatment

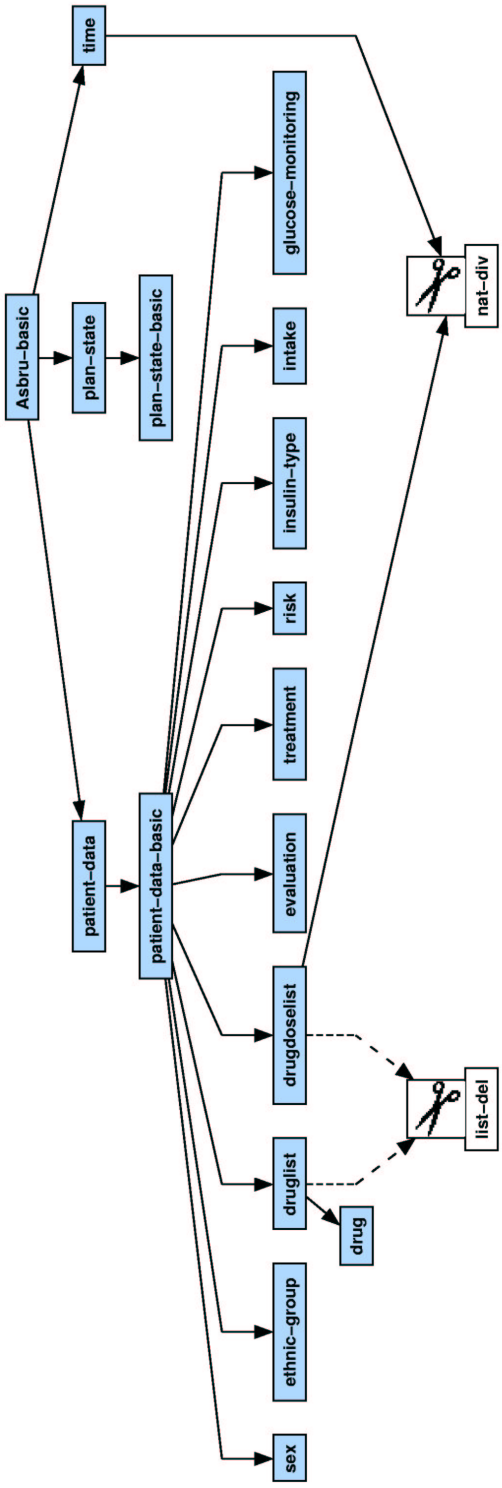


Figure 8: Basic Asbru parts of Diabetes

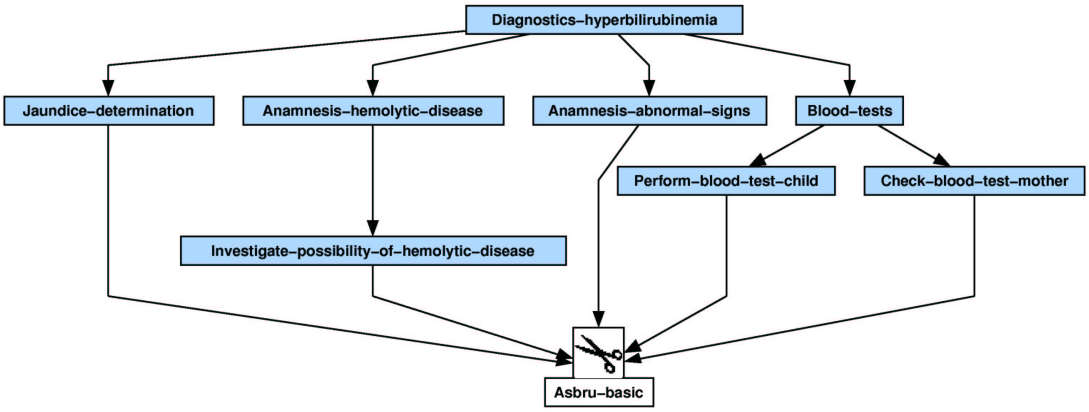


Figure 9: Diagnostics part of Diabetes

### B.2.2 The Specifications

— The following text was automatically generated from KIV. —

Diabetes-Mellitus-Type-2, 41
Diagnostics, 41
Policy, 41
Anamnesis, 42
Glucose-determination, 43
Policy-for-chiropodist-referral, 43
Policy-for-concurrent-diseases, 44
Policy-for-consultation, 45
Policy-for-hypoglycemic-coma, 46
Policy-for-nurse-referral, 47
Risk-inventory, 48
Treatments-and-Controls, 50
Anamnesis-olderthan-45, 51
Anamnesis-typical-signs, 52
Annual-control, 53
Fasting-glucose-test-manual, 55
Insulin-DMT2-treatments, 55
Non-fasting-glucose-test-manual, 56
Non-insulin-DMT2-treatments, 56
Treatment-of-CV-disease-risk-factors, 57
Albumin-test, 58
Cholesterol-tests, 59
Insulin-plus-antidiabetics-treatment, 59
Microalbuminuria-treatment, 60
Only-insulin-treatment, 60
Quarterly-control, 61
SU-derivative-plus-metformin-treatment, 62
SU-derivative-treatment, 63
Albumin-creatinin-ratio-test-manual, 63
Albumin-test-manual, 64
Change-insulin-to-2dd, 64
Check-for-antidiabetic-problems, 65

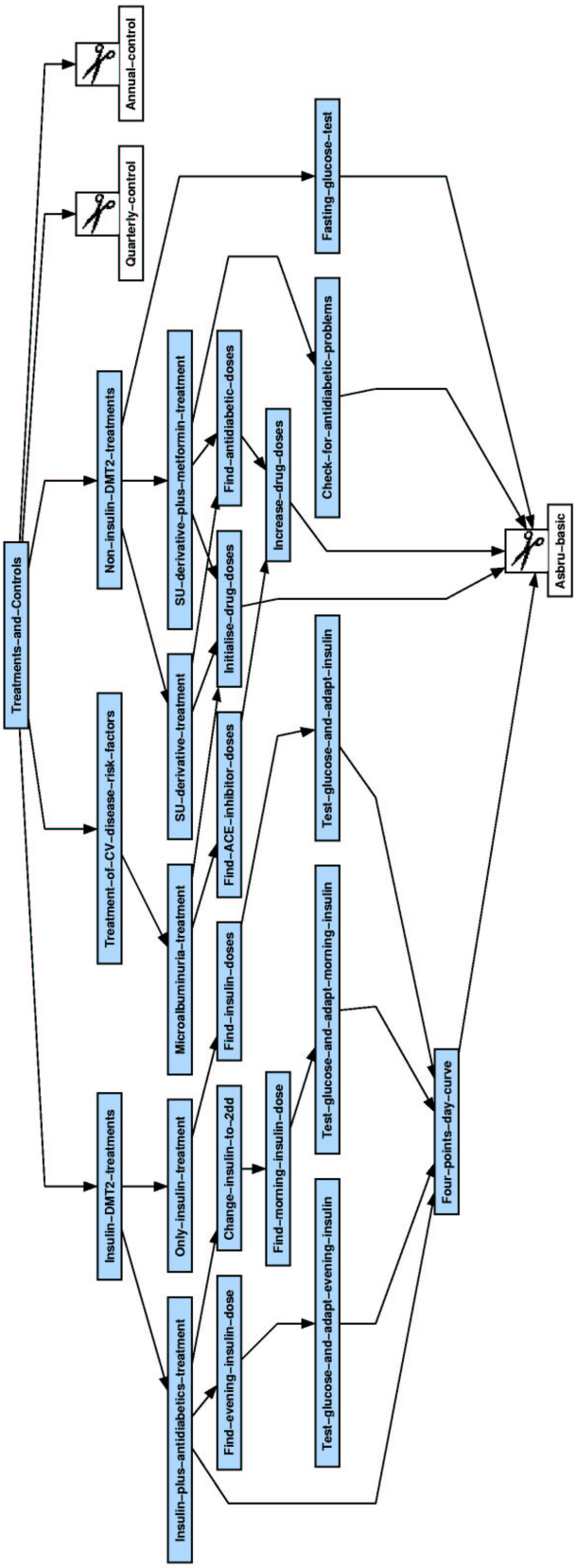


Figure 10: Treatment part of Diabetes

Fasting-glucose-test, 66
Feet-examination, 66
Find-ACE-inhibitor-doses, 66
Find-antidiabetic-doses, 67
Find-evening-insulin-dose, 67
Find-insulin-doses, 68
Initialise-drug-doses, 68
Find-morning-insulin-dose, 69
Increase-drug-doses, 69
Test-glucose-and-adapt-evening-insulin, 70
Test-glucose-and-adapt-insulin, 70
Test-glucose-and-adapt-morning-insulin, 71
Four-points-day-curve, 71
Asbru-basic, 72
patient-data, 72
plan-state, 30
time, 31
patient-data-basic, 74
plan-state-basic, 31
drugdoselist, 75
druglist, 75
ethnic-group, 76
evaluation, 76
glucose-monitoring, 76
insulin-type, 76
intake, 76
risk, 77
sex, 77
treatment, 77
drug, 77
bool, 33
list-del, 77
nat-div, 34
list-last, 78
nat-mult, 34
list-dup, 79
list, 79
list-data, 80
elem, 80
nat, 34
nat-basic2, 35
nat-basic1, 35

```
Diabetes-Mellitus-Type-2 =
enrich Diagnostics, Policy with
    procedures diabetes-mellitus-type-2#() : (patient-data, nat, plan-state);
    variables diabetes-mellitus-type-2-state, policy-state: plan-state;
    declaration

      diabetes-mellitus-type-2# (pd, time, diabetes-mellitus-type-2-state)
      var diagnostics-state = inactive, policy-state = inactive in
      begin
        diagnostics#(; pd, time, diagnostics-state);
        policy#(; pd, time, policy-state);
        if diagnostics-state[.] = completed ∨ policy-state[.] = completed then
          diabetes-mellitus-type-2-state := completed
        else
          diabetes-mellitus-type-2-state := aborted
      end
end enrich
```

---

```
Diagnostics =
enrich Anamnesis, Glucose-determination, Risk-inventory with
    procedures diagnostics#() : (patient-data, nat, plan-state);
    variables diagnostics-state: plan-state;
    declaration

      diagnostics# (pd, time, diagnostics-state)
      var anamnesis-state = inactive,
          glucose-determination-state = inactive,
          risk-inventory-state = inactive
      in
      begin
        anamnesis#(; pd, time, anamnesis-state);
        glucose-determination#(; pd, time, glucose-determination-state);
        risk-inventory#(; pd, time, risk-inventory-state);
        if anamnesis-state = completed then diagnostics-state := completed else
          diagnostics-state := aborted
      end
end enrich
```

---

```
Policy =
enrich Treatments-and-Controls, Policy-for-concurrent-diseases, Policy-for-hypoglycemic-coma, Policy-for-consultation,
    procedures
          policy#              ()   : (patient-data, nat, plan-state);
          education-dmt2#      ()   : (patient-data, nat, plan-state);
    variables policy-state, education-dmt2-state: plan-state;
    declaration

      policy# (pd, time, policy-state)
      var education-dmt2-state = inactive,
          treatments-and-controls-state = inactive,
          policy-for-concurrent-diseases-state = inactive,
          policy-for-hypoglycemic-coma-state = inactive,
          policy-for-consultation-state = inactive,
          policy-for-chiropodist-referral-state = inactive,
```

```
        policy-for-nurse-referral-state = inactive
    in
    begin
      await pd[.].glucose-evaluation = dmt2;
      education-dmt2#(; pd, time, education-dmt2-state);
        treatments-and-controls#(; pd,
                                   time,
                                   treatments-and-controls-state)
      ||s while policy-for-concurrent-diseases-state[.] ≠ completed do
            policy-for-concurrent-diseases#(; pd,
                                              time,
                                              policy-for-concurrent-diseases-state)
      ||s while policy-for-hypoglycemic-coma-state[.] ≠ completed do
            policy-for-hypoglycemic-coma#(; pd,
                                             time,
                                             policy-for-hypoglycemic-coma-state)
      ||s while policy-for-consultation-state[.] ≠ completed do
            policy-for-consultation#(; pd,
                                       time,
                                       policy-for-consultation-state)
      ||s while policy-for-chiropodist-referral-state[.] ≠ completed do
            policy-for-chiropodist-referral#(; pd,
                                               time,
                                               policy-for-chiropodist-referral-state)
      ||s while policy-for-nurse-referral-state ≠ completed do
            policy-for-nurse-referral#(; pd,
                                         time,
                                         policy-for-nurse-referral-state);
      if   education-dmt2-state[.] = completed
        ∧ treatments-and-controls-state[.] = completed
      then
        policy-state := completed
      else
        policy-state := aborted
    end;

    education-dmt2# (pd, time, education-dmt2-state)
    begin
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      education-dmt2-state := completed
    end
end enrich
```

---

```
Anamnesis =
enrich Anamnesis-typical-signs, Anamnesis-olderthan-45 with
    procedures anamnesis#() : (patient-data, nat, plan-state);
    variables anamnesis-state: plan-state;
    declaration

      anamnesis# (pd, time, anamnesis-state)
      var anamnesis-typical-signs-state = inactive, anamnesis-olderthan-45-state = inactive in
      begin
```

```
        anamnesis-typical-signs#(; pd, time, anamnesis-typical-signs-state);
        anamnesis-olderthan-45#(; pd, time, anamnesis-olderthan-45-state);
        pd := set-glucose-determination-needed(pd[.],
eval-glucose-determination-needed(pd[.]));
        if anamnesis-typical-signs-state[.] = completed then anamnesis-state := completed else
          anamnesis-state := aborted
      end
end enrich
```

---

```
Glucose-determination =
enrich Fasting-glucose-test-manual, Non-fasting-glucose-test-manual, Fasting-glucose-test with
    procedures glucose-determination#() : (patient-data, nat, plan-state);
    variables glucose-determination-state: plan-state;
    declaration

      glucose-determination# (pd, time, glucose-determination-state)
      var fasting-glucose-test-manual-state = inactive,
          non-fasting-glucose-test-manual-state = inactive,
          fasting-glucose-test-state = inactive
      in
      begin
        await pd[.].glucose-determination-needed;
          break
                fasting-glucose-test-manual#(; pd,
                                             time,
                                             fasting-glucose-test-manual-state)
            ||a non-fasting-glucose-test-manual#(; pd,
                                                   time,
                                                   non-fasting-glucose-test-manual-state)
          if   fasting-glucose-test-manual-state[.] = completed
             ∨ non-fasting-glucose-test-manual-state[.] = completed
        ||a if pd[.].glucose-evaluation = dmt2 then
              begin
                var time0 = time[.], dt = ? in
                if 2 ≤ dt[.] ∧ dt[.] ≤ 7 then
                  await dt[.] = time[.] - time0[.]
                else
                false;
                fasting-glucose-test#(; pd,
                                        time,
                                        fasting-glucose-test-state)
              end;
        if   (   fasting-glucose-test-manual-state[.] = completed
              ∨ non-fasting-glucose-test-manual-state[.] = completed)
           ∧ (fasting-glucose-test-state[.] = completed ∨ pd[.].glucose-evaluation ≠ dmt2)
        then
          glucose-determination-state := completed
        else
          glucose-determination-state := aborted
      end
end enrich
```

---

Policy-for-chiropodist-referral =

```
enrich Asbru-basic with
    procedures
        policy-for-chiropodist-referral#   ()   : (patient-data, nat, plan-state);
        chiropodist-referral#              ()   : nat;
    variables
        policy-for-chiropodist-referral-state: plan-state;
        chiropodist-referral-over: bool;
    declaration

      policy-for-chiropodist-referral# (pd, time, policy-for-chiropodist-referral-state)
      var chiropodist-referral-over = ? in
      begin
        await pd[.].pressure-points ∨ pd[.].bad-standing-habits;
        break
          chiropodist-referral#(; time);
          chiropodist-referral-over := ?;
          var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        if chiropodist-referral-over[.];
        if chiropodist-referral-over[.] then
          policy-for-chiropodist-referral-state := completed
        else
          policy-for-chiropodist-referral-state := aborted
      end;

      chiropodist-referral# (time)
      begin
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end
end enrich
```

---

```
Policy-for-concurrent-diseases =
enrich Asbru-basic with
    procedures
        policy-for-concurrent-diseases#      ()   : (patient-data, nat, plan-state);
        glucose-lowering-therapy-advice#     ()   : (patient-data, nat, plan-state);
        extra-fluid-intake-prescription#     ()   : (patient-data, nat, plan-state);
    variables
        policy-for-concurrent-diseases-state, glucose-lowering-therapy-advice-state, extra-fluid-intake-prescription-state:
        concurrent-diseases-over: bool;
    declaration

      policy-for-concurrent-diseases# (pd, time, policy-for-concurrent-diseases-state)
      var extra-fluid-intake-prescription-state = inactive,
          glucose-lowering-therapy-advice-state = inactive,
          concurrent-diseases-over = ?
      in
      begin
        await pd[.].fever ∨ pd[.].vomiting ∨ pd[.].diarrhea;
        break
          extra-fluid-intake-prescription#(; pd,
                                             time,
                                             extra-fluid-intake-prescription-state);
          glucose-lowering-therapy-advice#(; pd,
```

```
                                        time,
                                        glucose-lowering-therapy-advice-state);
        concurrent-diseases-over := ?;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      if concurrent-diseases-over[.]
    end;

    glucose-lowering-therapy-advice# (pd, time, glucose-lowering-therapy-advice-state)
    begin
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      glucose-lowering-therapy-advice-state := completed
    end;

    extra-fluid-intake-prescription# (pd, time, extra-fluid-intake-prescription-state)
    begin
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      extra-fluid-intake-prescription-state := completed
    end
end enrich
```

---

```
Policy-for-consultation =
enrich Asbru-basic with
  procedures
      policy-for-consultation#   ()   : (patient-data, nat, plan-state);
      consultation#              ()   : (patient-data, nat, plan-state);
  variables
    policy-for-consultation-state, consultation-state: plan-state;
    consultation-over: bool;
  declaration

    policy-for-consultation# (pd, time, policy-for-consultation-state)
    var consultation-over = ?, consultation-state = inactive in
    begin
      await   pd[.].insulin-adjustment-needed
            ∨ pd[.].postprandial-glucose-monitoring = bad ∧ pd[.].insulin-dd = 2
            ∨ pd[.].diabetes-ulcus-after-2-weeks
            ∨ pd[.].creatinin > 200
            ∨ 30 > pd.creatinin-clearance
            ∨ pd[.].lethargy
            ∨ pd[.].dehydration
            ∨ pd[.].vomiting
            ∨ pd[.].hypoglycemic-coma-after-30-min;
      break
        consultation#(; pd, time, consultation-state);
        consultation-over := ?;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      if consultation-over[.];
      policy-for-consultation-state := completed
    end;

    consultation# (pd, time, consultation-state)
```

```
    begin
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      consultation-state := completed
    end
end enrich
```

---

```
Policy-for-hypoglycemic-coma =
enrich Asbru-basic with
  procedures
      policy-for-hypoglycemic-coma#              ()  : (patient-data, nat, plan-state);
      glucose-solution-prescription-manual#      ()  : (patient-data, nat, plan-state);
      glucagon-prescription-manual#              ()  : (patient-data, nat, plan-state);
      carbohydrate-rich-feeding-prescription#    ()  : (patient-data, nat, plan-state);
      hypoglycemia-cause-investigation#          ()  : (patient-data, nat, plan-state);
  variables
    policy-for-hypoglycemic-coma-state, glucose-solution-prescription-manual-state, glucagon-prescription-manual-st
    hypoglycemic-coma-over: bool;
  declaration

    policy-for-hypoglycemic-coma# (pd, time, policy-for-hypoglycemic-coma-state)
    var policy-for-hypoglycemic-coma-state = inactive,
        glucose-solution-prescription-manual-state = inactive,
        glucagon-prescription-manual-state = inactive,
        carbohydrate-rich-feeding-prescription-state = inactive,
        hypoglycemia-cause-investigation-state = inactive,
        hypoglycemic-coma-over = ?
    in
    begin
      await pd[.].hypoglycemic-coma;
      break
        break
            glucose-solution-prescription-manual#(; pd,
                                                    time,
                                                    glucose-solution-prescription-manual-
state)
          ||a glucagon-prescription-manual#(; pd,
                                             time,
                                             glucagon-prescription-manual-state)
        if   glucose-solution-prescription-manual-state[.] = completed
          ∨ glucagon-prescription-manual-state[.] = completed;
        carbohydrate-rich-feeding-prescription#(; pd,
                                                  time,
                                                  carbohydrate-rich-feeding-prescription-
state);
        hypoglycemia-cause-investigation#(; pd,
                                          time,
                                          hypoglycemia-cause-investigation-state);
        hypoglycemic-coma-over := ?;
        pd := set-hypoglycemic-coma-over(pd, hypoglycemic-coma-over);
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      if pd[.].hypoglycemic-coma-over;
      if pd[.].hypoglycemic-coma-over then
```

```
            policy-for-hypoglycemic-coma-state := aborted
        else
            policy-for-hypoglycemic-coma-state := completed
    end;

    glucose-solution-prescription-manual# (pd, time,
glucose-solution-prescription-manual-state)
    begin
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        glucose-solution-prescription-manual-state := completed
    end;

    glucagon-prescription-manual# (pd, time, glucagon-prescription-manual-state)
    begin
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        glucagon-prescription-manual-state := completed
    end;

    carbohydrate-rich-feeding-prescription# (pd, time,
carbohydrate-rich-feeding-prescription-state)
    begin
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        carbohydrate-rich-feeding-prescription-state := completed
    end;

    hypoglycemia-cause-investigation# (pd, time, hypoglycemia-cause-investigation-state)
    begin
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        hypoglycemia-cause-investigation-state := completed
    end
end enrich
```

---

```
Policy-for-nurse-referral =
enrich Asbru-basic with
  procedures
      policy-for-nurse-referral#   ()   : (patient-data, nat, plan-state);
      nurse-referral#              ()   : nat;
  variables
    policy-for-nurse-referral-state: plan-state;
    nurse-referral-over: bool;
  declaration

    policy-for-nurse-referral# (pd, time, policy-for-nurse-referral-state)
    var nurse-referral-over = ? in
    begin
        await pd[.].self-monitoring-training-needed ∨ pd[.].insulin-treatment-start;
```

```
        break
          nurse-referral#(; time);
          nurse-referral-over := ?;
          var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        if nurse-referral-over[.];
        if nurse-referral-over[.] then policy-for-nurse-referral-state := completed else
          policy-for-nurse-referral-state := aborted
      end;

      nurse-referral# (time)
      begin
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end
end enrich
```

---

```
Risk-inventory =
enrich Albumin-test, Cholesterol-tests, Feet-examination with
    procedures
          risk-inventory#                ()   : (patient-data, nat, plan-state);
          retinopathy-investigation#     ()   : (patient-data, nat, plan-state);
    variables
        risk-inventory-state, retinopathy-investigation-state: plan-state;
        ch-diseases, ch-disease-in-youngerthan-60-1st-grade-relatives, feeding-habits, smoking, alcohol-consuming, physic
        quetelet-index, higher-blood-pressure, lower-blood-pressure, ghb: nat;
    declaration

      risk-inventory# (pd, time, risk-inventory-state)
      var ch-diseases = ?,
          ch-disease-in-youngerthan-60-1st-grade-relatives = ?,
          feeding-habits = ?,
          smoking = ?,
          alcohol-consuming = ?,
          physical-exercise = ?,
          quetelet-index = ?,
          higher-blood-pressure = ?,
          lower-blood-pressure = ?,
          ghb = ?,
          creatinin = ?,
          albumin-test-state = inactive,
          cholesterol-tests-state = inactive,
          feet-examination-state = inactive,
          retinopathy-investigation-state = inactive
      in
      begin
        await pd[.].glucose-evaluation = dmt2;
        break
              begin
                ch-diseases := ?;
                pd := set-ch-diseases(pd[.], ch-diseases[.]);
                var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
              end
          ||a begin
```

```
            ch-disease-in-youngerthan-60-1st-grade-relatives := ?;
            pd := set-ch-disease-in-youngerthan-60-1st-grade-relatives(pd[.],
ch-disease-in-youngerthan-60-1st-grade-relatives[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        end
    ||a begin
            feeding-habits := ?;
            pd := set-feeding-habits(pd[.], feeding-habits[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        end
    ||a begin
            smoking := ?;
            pd := set-smoking(pd[.], smoking[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        end
    ||a begin
            alcohol-consuming := ?;
            pd := set-alcohol-consuming(pd[.], alcohol-consuming[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        end
    ||a begin
            physical-exercise := ?;
            pd := set-physical-exercise(pd[.], physical-exercise[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        end
    ||a begin
            quetelet-index := ?;
            pd := set-quetelet-index(pd[.], quetelet-index[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        end
    ||a begin
            higher-blood-pressure := ?;
            pd := set-higher-blood-pressure(pd[.], higher-blood-pressure[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        end
    ||a begin
            lower-blood-pressure := ?;
            pd := set-lower-blood-pressure(pd[.], lower-blood-pressure[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        end
    ||a if pd[.].age < 50 then albumin-test#(; pd,
                time,
                albumin-test-state)
    ||a begin
            ghb := ?;
            pd := set-ghb(pd[.], ghb[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        end
    ||a begin
            creatinin := ?;
            pd := set-creatinin(pd[.], creatinin[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        end
    ||a cholesterol-tests#(; pd,
```

```
                                    time,
                                    cholesterol-tests-state)
          ||_a feet-examination#(; pd,
                                    time,
                                    feet-examination-state)
          ||_a retinopathy-investigation#(; pd,
                                            time,
                                            retinopathy-investigation-state)
        if   albumin-test-state[.] = aborted
          ∨ cholesterol-tests-state[.] = aborted
          ∨ feet-examination-state[.] = aborted
          ∨ retinopathy-investigation-state[.] = aborted;
        if   (pd[.].age ≥ 50 ∨ albumin-test-state[.] = completed)
          ∧ cholesterol-tests-state[.] = completed
          ∧ feet-examination-state[.] = completed
          ∧ retinopathy-investigation-state[.] = completed
        then
          risk-inventory-state := completed
        else
          risk-inventory-state := aborted
      end;

      retinopathy-investigation# (pd, time, retinopathy-investigation-state)
      begin
        skip;
        var time_0 = time[.], dt = ? in await time_0[.] + dt[.] ≤ time[.];
        retinopathy-investigation-state := completed
      end
end enrich
```

---

```
Treatments-and-Controls =
enrich Non-insulin-DMT2-treatments, Insulin-DMT2-treatments, Treatment-of-CV-disease-risk-factors, Quarterly-contr
    procedures treatments-and-controls#() : (patient-data, nat, plan-state);
    variables treatments-and-controls-state: plan-state;
    declaration

      treatments-and-controls# (pd, time, treatments-and-controls-state)
      var non-insulin-dmt2-treatments-state = inactive,
          insulin-dmt2-treatments-state = inactive,
          treatment-of-cv-disease-risk-factors-state = inactive,
          quarterly-control-state = inactive,
          annual-control-state = inactive
      in
      begin
        break
          non-insulin-dmt2-treatments#(; pd, time, non-insulin-dmt2-treatments-state);
            if non-insulin-dmt2-treatments-state = aborted then
              insulin-dmt2-treatments#(; pd,
                                         time,
                                         insulin-dmt2-treatments-state)
          ||_s treatment-of-cv-disease-risk-factors#(; pd,
                                                       time,
                                                       treatment-of-cv-disease-risk-factors-state)
```

```
        ||s while true do
              begin
                quarterly-control#(; pd,
                                     time,
                                     quarterly-control-state);
                var time0 = time[.], dt = ? in
                if 25 ≤ dt[.] ∧ dt[.] ≤ 35 then
                  await time0[.] + dt[.] = time[.]
                else
                  false
              end
        ||s while true do
              begin
                annual-control#(; pd,
                                  time,
                                  annual-control-state);
                var time0 = time[.], dt = ? in
                if 115 ≤ dt[.] ∧ dt[.] ≤ 125 then
                  await time0[.] + dt[.] = time[.]
                else
                  false
              end
      if ¬ pd[.].alive;
      if ¬ pd[.].alive then treatments-and-controls-state := completed else
        treatments-and-controls-state := aborted
    end
end enrich
```

---

```
Anamnesis-olderthan-45 =
enrich Asbru-basic with
  procedures anamnesis-olderthan-45#() : (patient-data, nat, plan-state);
  variables
    anamnesis-olderthan-45-state: plan-state;
    dmt2-in-1sr-grade-relatives, hypertension, cv-diseases, fat-metabolism-problems, dm-in-past, newborns-biggertha
    quetelet-index: nat;
    ethnic-group: ethnic-group;
  declaration

    anamnesis-olderthan-45# (pd, time, anamnesis-olderthan-45-state)
    var dmt2-in-1sr-grade-relatives = ?,
        hypertension = ?,
        cv-diseases = ?,
        fat-metabolism-problems = ?,
        quetelet-index = ?,
        ethnic-group = ?,
        dm-in-past = ?,
        newborns-biggerthan-4kg = ?
    in
    begin
      await 45 ≤ pd[.].age ∧ pd[.].triennial-visit;
        begin
          dmt2-in-1sr-grade-relatives := ?;
          pd := set-dmt2-in-1sr-grade-relatives(pd[.], dmt2-in-1sr-grade-relatives[.]);
```

```
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end
        ||a begin
            hypertension := ?;
            pd := set-hypertension(pd[.], hypertension[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end
        ||a begin
            cv-diseases := ?;
            pd := set-cv-diseases(pd[.], cv-diseases[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end
        ||a begin
            fat-metabolism-problems := ?;
            pd := set-fat-metabolism-problems(pd[.], fat-metabolism-problems[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end
        ||a begin
            quetelet-index := ?;
            pd := set-quetelet-index(pd[.], quetelet-index[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end
        ||a begin
            ethnic-group := ?;
            pd := set-ethnic-group(pd[.], ethnic-group[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end;
        if pd[.].sex = female then
          begin
            dm-in-past := ?;
            pd := set-dm-in-past(pd[.], dm-in-past[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
            newborns-biggerthan-4kg := ?;
            pd := set-newborns-biggerthan-4kg(pd[.], newborns-biggerthan-4kg[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end;
        pd := set-risk-factors(pd[.], eval-risk-factors(pd[.]));
        anamnesis-olderthan-45-state := completed
      end
end enrich
```

---

```
Anamnesis-typical-signs =
enrich Asbru-basic with
  procedures anamnesis-typical-signs#() : (patient-data, nat, plan-state);
  variables
    anamnesis-typical-signs-state: plan-state;
    thirst, polyuria, weight-loss, pruritis-vulvae: bool;
  declaration

    anamnesis-typical-signs# (pd, time, anamnesis-typical-signs-state)
    var thirst = ?, polyuria = ?, weight-loss = ?, pruritis-vulvae = ? in
    begin
        begin
```

```
            thirst := ?;
            pd := set-thirst(pd[.], thirst[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end
        ||a begin
            polyuria := ?;
            pd := set-polyuria(pd[.], polyuria[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end
        ||a begin
            weight-loss := ?;
            pd := set-weight-loss(pd[.], weight-loss[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end;
        if pd[.].age > 60 ∧ pd[.].sex = female then
          begin
            pruritis-vulvae := ?;
            pd := set-pruritis-vulvae(pd[.], pruritis-vulvae[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end;
        pd := set-typical-signs(pd[.], eval-typical-signs(pd[.]));
        anamnesis-typical-signs-state := completed
    end
end enrich
```

---

```
Annual-control =
enrich Quarterly-control, Cholesterol-tests, Albumin-test with
  procedures
      annual-control#                 ()  : (patient-data, nat, plan-state);
      injection-points-investigation# ()  : (patient-data, nat, plan-state);
      funduscopy#                     ()  : (patient-data, nat, plan-state);
  variables
    annual-control-state, injection-points-investigation-state, funduscopy-state: plan-state;
    visus-problems, cv-problems, neuropathy, sexual-problems, creatinin: bool;
    higher-blood-pressure, lower-blood-pressure, ghb: nat;
  declaration

    annual-control# (pd, time, annual-control-state)
    var quarterly-control-state = inactive,
        feet-examination-state = inactive,
        injection-points-investigation-state = inactive,
        cholesterol-tests-state = inactive,
        albumin-test-state = inactive,
        funduscopy-state = inactive,
        visus-problems = ?,
        cv-problems = ?,
        neuropathy = ?,
        sexual-problems = ?,
        higher-blood-pressure = ?,
        lower-blood-pressure = ?,
        ghb = ?,
        creatinin = ?
    in
```

```
    begin
      quarterly-control#(; pd, time, quarterly-control-state);
      pd := set-visus-problems(pd[.], visus-problems[.]);
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      pd := set-cv-problems(pd[.], cv-problems[.]);
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      pd := set-neuropathy(pd[.], neuropathy[.]);
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      pd := set-sexual-problems(pd[.], sexual-problems[.]);
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      pd := set-higher-blood-pressure(pd[.], higher-blood-pressure[.]);
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      pd := set-lower-blood-pressure(pd[.], lower-blood-pressure[.]);
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      feet-examination#(; pd, time, feet-examination-state);
      if   pd[.].dmt2-treatment = insulin
        ∨ pd[.].dmt2-treatment = insulin-plus-antidiabetics
      then
        injection-points-investigation#(; pd, time, injection-points-investigation-state);
      pd := set-ghb(pd[.], ghb[.]);
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      pd := set-creatinin(pd[.], creatinin[.]);
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      if ¬ pd[.].cholesterol-treatment then
        cholesterol-tests#(; pd, time, cholesterol-tests-state);
      if pd[.].age < 50 then albumin-test#(; pd, time, albumin-test-state);
      funduscopy#(; pd, time, funduscopy-state);
      if   quarterly-control-state[.] = completed
        ∧ feet-examination-state[.] = completed
        ∧ injection-points-investigation-state[.] = completed
        ∧ cholesterol-tests-state[.] = completed
        ∧ albumin-test-state[.] = completed
        ∧ funduscopy-state[.] = completed
      then
        annual-control-state := completed
      else
        annual-control-state := aborted
    end;

    injection-points-investigation# (pd, time, injection-points-investigation-state)
    begin
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      injection-points-investigation-state := completed
    end;

    funduscopy# (pd, time, funduscopy-state)
    begin
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      funduscopy-state := completed
    end
end enrich
```

```
Fasting-glucose-test-manual =
enrich Asbru-basic with
    procedures fasting-glucose-test-manual#() : (patient-data, nat, plan-state);
    variables
        fasting-glucose-test-manual-state: plan-state;
        glucose-measurement-type: bool;
        fasting-glucose: nat;
    declaration

      fasting-glucose-test-manual# (pd, time, fasting-glucose-test-manual-state)
      var glucose-measurement-type = ?, fasting-glucose = ? in
      begin
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
          begin
            glucose-measurement-type := ?;
            pd := set-glucose-measurement-type(pd[.], glucose-measurement-type[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end
        ||a begin
            fasting-glucose := ?;
            pd := set-fasting-glucose(pd[.], fasting-glucose[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end;
        fasting-glucose-test-manual-state := completed
      end
end enrich
```

---

```
Insulin-DMT2-treatments =
enrich Insulin-plus-antidiabetics-treatment, Only-insulin-treatment with
    procedures
        insulin-dmt2-treatments#   ()   : (patient-data, nat, plan-state);
        education-insulin#         ()   : (patient-data, nat, plan-state);
    variables insulin-dmt2-treatments-state, education-insulin-state: plan-state;
    declaration

      insulin-dmt2-treatments# (pd, time, insulin-dmt2-treatments-state)
      var education-insulin-state = inactive,
          insulin-plus-antidiabetics-treatment-state = inactive,
          only-insulin-treatment-state = inactive
      in
      begin
        await pd[.].glucose-monitoring = bad;
        education-insulin#(; pd, time, education-insulin-state);
        break
              insulin-plus-antidiabetics-treatment#(; pd,
                                                    time,
                                                    insulin-plus-antidiabetics-treatment-state)
          ||a only-insulin-treatment#(; pd,
                                        time,
                                        only-insulin-treatment-state)
        if   insulin-plus-antidiabetics-treatment-state[.] = completed
          ∨ only-insulin-treatment-state[.] = completed;
        if   insulin-plus-antidiabetics-treatment-state[.] = completed
          ∨ only-insulin-treatment-state[.] = completed
```

```
        then
          insulin-dmt2-treatments-state := completed
        else
          insulin-dmt2-treatments-state := aborted
      end;

      education-insulin# (pd, time, education-insulin-state)
      begin
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        education-insulin-state := completed
      end
end enrich
```

---

```
Non-fasting-glucose-test-manual =
enrich Asbru-basic with
    procedures non-fasting-glucose-test-manual#() : (patient-data, nat, plan-state);
    variables
        non-fasting-glucose-test-manual-state: plan-state;
        glucose-measurement-type: bool;
        non-fasting-glucose: nat;
    declaration

      non-fasting-glucose-test-manual# (pd, time, non-fasting-glucose-test-manual-state)
      var glucose-measurement-type = ?, non-fasting-glucose = ? in
      begin
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
          begin
            glucose-measurement-type := ?;
            pd := set-glucose-measurement-type(pd[.], glucose-measurement-type[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end
        ||a begin
            non-fasting-glucose := ?;
            pd := set-non-fasting-glucose(pd[.], non-fasting-glucose[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
          end;
        non-fasting-glucose-test-manual-state := completed
      end
end enrich
```

---

```
Non-insulin-DMT2-treatments =
enrich Fasting-glucose-test, SU-derivative-treatment, SU-derivative-plus-metformin-treatment with
    procedures
        non-insulin-dmt2-treatments#   ()   : (patient-data, nat, plan-state);
        diet-specialist-referral#      ()   : (patient-data, nat, plan-state);
    variables non-insulin-dmt2-treatments-state, diet-specialist-referral-state: plan-state;
    declaration

      non-insulin-dmt2-treatments# (pd, time, non-insulin-dmt2-treatments-state)
      var diet-specialist-referral-state = inactive,
          fasting-glucose-test-state = inactive,
          su-derivative-treatment-state = inactive,
```

```
        su-derivative-plus-metformin-treatment-state = inactive
    in
    begin
      break
        diet-specialist-referral#(; pd, time, diet-specialist-referral-state);
        var time0 = time[.], dt = ? in
        if 11 ≤ dt[.] ∧ dt[.] ≤ 13 then await time0[.] + dt[.] = time[.] else false;
        fasting-glucose-test#(; pd, time, fasting-glucose-test-state);
        su-derivative-treatment#(; pd, time, su-derivative-treatment-state);
        su-derivative-plus-metformin-treatment#(; pd,
                                                  time,
                                                  su-derivative-plus-metformin-treatment-
state)
      if   pd[.].glucose-monitoring = bad
        ∧ su-derivative-plus-metformin-treatment-state[.] = completed;
      if diet-specialist-referral-state[.] = completed then
        non-insulin-dmt2-treatments-state := completed
      else
        non-insulin-dmt2-treatments-state := aborted
    end;

    diet-specialist-referral# (pd, time, diet-specialist-referral-state)
    begin
      skip;
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      diet-specialist-referral-state := completed
    end
end enrich
```

---

```
Treatment-of-CV-disease-risk-factors =
enrich Microalbuminuria-treatment with
  procedures
      treatment-of-cv-disease-risk-factors#   ()   : (patient-data, nat, plan-state);
      smoking-advice#                         ()   : (patient-data, nat, plan-state);
      hypertension-treatment#                 ()   : (patient-data, nat, plan-state);
      cholesterol-treatment#                  ()   : (patient-data, nat, plan-state);
  variables treatment-of-cv-disease-risk-factors-state, smoking-advice-state, hypertension-treatment-state, cholesterol-
  declaration

    treatment-of-cv-disease-risk-factors# (pd, time, treatment-of-cv-disease-risk-factors-state)
    var smoking-advice-state = inactive,
        hypertension-treatment-state = inactive,
        cholesterol-treatment-state = inactive,
        microalbuminuria-treatment-state = inactive
    in
    begin
       smoking-advice#(; pd,
                         time,
                         smoking-advice-state)
      ||s hypertension-treatment#(; pd,
                                    time,
                                    hypertension-treatment-state)
      ||s cholesterol-treatment#(; pd,
```

```
                                    time,
                                    cholesterol-treatment-state)
      ||_s microalbuminuria-treatment#(; pd,
                                    time,
                                    microalbuminuria-treatment-state);
      treatment-of-cv-disease-risk-factors-state := completed
    end;

    smoking-advice# (pd, time, smoking-advice-state)
    begin
      await pd[.].smoking;
      skip;
      var time_0 = time[.], dt = ? in await time_0[.] + dt[.] ≤ time[.];
      smoking-advice-state := completed
    end;

    hypertension-treatment# (pd, time, hypertension-treatment-state)
    begin
      await pd[.].higher-blood-pressure > 50 ∨ pd[.].lower-blood-pressure > 85;
      skip;
      var time_0 = time[.], dt = ? in await time_0[.] + dt[.] ≤ time[.];
      hypertension-treatment-state := completed
    end;

    cholesterol-treatment# (pd, time, cholesterol-treatment-state)
    begin
      await   pd[.].fat-metabolism-problems
            ∧ pd[.].life-expectancy > 5
            ∧ pd[.].ch-disease-risk = significant;
      skip;
      var time_0 = time[.], dt = ? in await time_0[.] + dt[.] ≤ time[.];
      cholesterol-treatment-state := completed
    end
end enrich
```

---

```
Albumin-test =
enrich Albumin-test-manual, Albumin-creatinin-ratio-test-manual with
  procedures albumin-test#() : (patient-data, nat, plan-state);
  variables albumin-test-state: plan-state;
  declaration

    albumin-test# (pd, time, albumin-test-state)
    var albumin-test-manual-state = inactive,
        albumin-creatinin-ratio-test-manual-state = inactive
    in
    begin
      break
        albumin-test-manual#(; pd, time, albumin-test-manual-state);
        albumin-creatinin-ratio-test-manual#(; pd,
                                    time,
                                    albumin-creatinin-ratio-test-manual-state)
      if   albumin-test-manual-state[.] = completed
        ∨ albumin-creatinin-ratio-test-manual-state[.] = completed;
```

```
        if   albumin-test-manual-state[.] = completed
          ∨ albumin-creatinin-ratio-test-manual-state[.] = completed
        then
          albumin-test-state := completed
        else
          albumin-test-state := aborted
      end
end enrich
```

---

```
Cholesterol-tests =
enrich Asbru-basic with
    procedures cholesterol-tests#() : (patient-data, nat, plan-state);
    variables
        cholesterol-tests-state: plan-state;
        total-cholesterol, hdl-cholesterol: nat;
        triglycerids: bool;
    declaration

      cholesterol-tests# (pd, time, cholesterol-tests-state)
      var total-cholesterol = ?, hdl-cholesterol = ?, triglycerids = ? in
      begin
        total-cholesterol := ?;
        pd := set-total-cholesterol(pd[.], total-cholesterol[.]);
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        hdl-cholesterol := ?;
        pd := set-hdl-cholesterol(pd[.], hdl-cholesterol[.]);
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        triglycerids := ?;
        pd := set-triglycerids(pd[.], triglycerids[.]);
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        cholesterol-tests-state := completed
      end
end enrich
```

---

```
Insulin-plus-antidiabetics-treatment =
enrich Four-points-day-curve, Find-evening-insulin-dose, Change-insulin-to-2dd with
    procedures insulin-plus-antidiabetics-treatment#() : (patient-data, nat, plan-state);
    variables
        insulin-plus-antidiabetics-treatment-state: plan-state;
        par-evening-insulin-dose: nat;
    declaration

      insulin-plus-antidiabetics-treatment# (pd, time,
insulin-plus-antidiabetics-treatment-state)
      var par-evening-insulin-dose = ?,
          four-points-day-curve-state = inactive,
          find-evening-insulin-dose-state = inactive,
          change-insulin-to-2dd-state = inactive
      in
      begin
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        pd := set-dmt2-treatment(pd[.], insulin-plus-antidiabetics);
        four-points-day-curve#(; pd, time, four-points-day-curve-state);
```

```
        pd := set-insulin-type(pd[.], mid-term);
        pd := set-insulin-dd(pd[.], 1);
        var par-evening-insulin-dose = ? in
        begin
          par-evening-insulin-dose := ?;
          pd := set-evening-insulin-dose(pd[.], par-evening-insulin-dose[.]);
          var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
        end;
        pd := set-evening-insulin-intake(pd[.], after-meal);
        var time0 = time[.], dt = ? in
        if 2 ≤ dt[.] ∧ dt[.] ≤ 4 then await time0[.] + dt[.] = time[.] else false;
        find-evening-insulin-dose#(; pd, time, find-evening-insulin-dose-state);
        change-insulin-to-2dd#(; pd, time, change-insulin-to-2dd-state);
        if four-points-day-curve-state[.] = completed then
          insulin-plus-antidiabetics-treatment-state := completed
        else
          insulin-plus-antidiabetics-treatment-state := aborted
      end
end enrich
```

---

```
Microalbuminuria-treatment =
enrich Initialise-drug-doses, Find-ACE-inhibitor-doses with
    procedures microalbuminuria-treatment#() : (patient-data, nat, plan-state);
    variables
        microalbuminuria-treatment-state: plan-state;
        drug-name: drug;
    declaration

      microalbuminuria-treatment# (pd, time, microalbuminuria-treatment-state)
      var drug-name = ?,
          initialise-drug-doses-state = inactive,
          find-ace-inhibitor-doses-state = inactive
      in
      begin
        await 50 > pd[.].age ∧ pd[.].microalbuminuria;
        drug-name := ?;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        pd := set-drugs(pd[.], pd[.].drugs + drug-name[.]);
        initialise-drug-doses#(; pd, time, initialise-drug-doses-state);
        find-ace-inhibitor-doses#(; pd, time, find-ace-inhibitor-doses-state);
        microalbuminuria-treatment-state := completed
      end
end enrich
```

---

```
Only-insulin-treatment =
enrich Find-insulin-doses with
    procedures only-insulin-treatment#() : (patient-data, nat, plan-state);
    variables
        only-insulin-treatment-state: plan-state;
        par-insulin-type: insulin-type;
    declaration

      only-insulin-treatment# (pd, time, only-insulin-treatment-state)
```

```
    var par-insulin-type = ?, find-insulin-doses-state = inactive in
    begin
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      pd := set-dmt2-treatment(pd[.], insulin);
      par-insulin-type := ?;
      pd := set-insulin-type(pd[.], par-insulin-type[.]);
      var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
      pd := set-insulin-dd(pd[.], 2);
      pd := set-morning-insulin-dose(pd[.], 12);
      pd := set-morning-insulin-intake(pd[.], before-meal);
      pd := set-evening-insulin-dose(pd[.], 6);
      pd := set-evening-insulin-intake(pd[.], before-meal);
      find-insulin-doses#(; pd, time, find-insulin-doses-state);
      if find-insulin-doses-state[.] = completed then
        only-insulin-treatment-state := completed
      else
        only-insulin-treatment-state := aborted
    end
end enrich
```

---

```
Quarterly-control =
enrich Four-points-day-curve, Fasting-glucose-test, Feet-examination with
  procedures quarterly-control#() : (patient-data, nat, plan-state);
  variables
    quarterly-control-state: plan-state;
    wellbeing, hypoglycemia, feeding-problems, treatment-problems, feet-problems: bool;
    weight: nat;
  declaration

    quarterly-control# (pd, time, quarterly-control-state)
    var wellbeing = ?,
        hypoglycemia = ?,
        feeding-problems = ?,
        treatment-problems = ?,
        weight = ?,
        feet-problems = ?,
        four-points-day-curve-state = inactive,
        fasting-glucose-test-state = inactive,
        feet-examination-state = inactive
    in
      begin
        pd := set-wellbeing(pd[.], wellbeing[.]);
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end
    ||a begin
        pd := set-hypoglycemia(pd[.], hypoglycemia[.]);
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end
    ||a begin
        pd := set-feeding-problems(pd[.], feeding-problems[.]);
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
      end
    ||a begin
```

```
            pd := set-treatment-problems(pd, treatment-problems[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
         end
      ||a begin
            pd := set-weight(pd[.], weight[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
         end
      ||a if   pd[.].dmt2-treatment = insulin
            ∨ pd[.].dmt2-treatment = insulin-plus-antidiabetics
          then
            four-points-day-curve#(; pd,
                                     time,
                                     four-points-day-curve-state)
          else
            fasting-glucose-test#(; pd,
                                    time,
                                    fasting-glucose-test-state)
      ||a begin
            pd := set-feet-problems(pd[.], feet-problems[.]);
            var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
            if pd[.].feet-problems then
               feet-examination#(; pd,
                                   time,
                                   feet-examination-state)
         end
end enrich
```

---

```
SU-derivative-plus-metformin-treatment =
enrich Initialise-drug-doses, Find-antidiabetic-doses, Check-for-antidiabetic-problems with
   procedures su-derivative-plus-metformin-treatment#() : (patient-data, nat, plan-state);
   variables
      su-derivative-plus-metformin-treatment-state: plan-state;
      drug-name: drug;
   declaration

      su-derivative-plus-metformin-treatment# (pd, time,
su-derivative-plus-metformin-treatment-state)
      var drug-name = ?,
          initialise-drug-doses-state = inactive,
          find-antidiabetic-doses-state = inactive,
          check-for-antidiabetic-problems-state = inactive
      in
      begin
         await pd[.].glucose-monitoring = bad;
         drug-name := ?;
         var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
         pd := set-drugs(pd[.], pd[.].drugs + drug-name);
         pd := set-drugs(pd[.], pd[.].drugs + metformin);
         initialise-drug-doses#(; pd, time, initialise-drug-doses-state);
         break
              find-antidiabetic-doses#(; pd,
                                         time,
                                         find-antidiabetic-doses-state)
```

```
        ||_s check-for-antidiabetic-problems#(; pd,
                                               time,
                                               check-for-antidiabetic-problems-state)
      if find-antidiabetic-doses-state[.] = completed;
      if   find-antidiabetic-doses-state[.] = completed
        ∧ initialise-drug-doses-state[.] = completed
      then
        su-derivative-plus-metformin-treatment-state := completed
      else
        su-derivative-plus-metformin-treatment-state := aborted
    end
end enrich
```

---

```
SU-derivative-treatment =
enrich Initialise-drug-doses, Find-antidiabetic-doses with
  procedures su-derivative-treatment#() : (patient-data, nat, plan-state);
  variables
    su-derivative-treatment-state: plan-state;
    drug-name: drug;
  declaration

    su-derivative-treatment# (pd, time, su-derivative-treatment-state)
    var drug-name = ?,
        initialise-drug-doses-state = inactive,
        find-antidiabetic-doses-state = inactive
    in
    begin
      await pd[.].glucose-monitoring = bad;
      pd := set-dmt2-treatment(pd[.], antidiabetics);
      if pd[.].quetelet-index ≤ 27 then
        begin
          drug-name := ?;
          var time_0 = time[.], dt = ? in await time_0[.] + dt[.] ≤ time[.];
          pd := set-drugs(pd[.], pd[.].drugs + drug-name)
        end
      else
        pd := set-drugs(pd[.], pd[.].drugs + metformin);
      initialise-drug-doses#(; pd, time, initialise-drug-doses-state);
      find-antidiabetic-doses#(; pd, time, find-antidiabetic-doses-state);
      if   initialise-drug-doses-state[.] = completed
        ∧ find-antidiabetic-doses-state[.] = completed
      then
        su-derivative-treatment-state := completed
      else
        su-derivative-treatment-state := aborted
    end
end enrich
```

---

```
Albumin-creatinin-ratio-test-manual =
enrich Asbru-basic with
  procedures albumin-creatinin-ratio-test-manual#() : (patient-data, nat, plan-state);
  variables
```

albumin-creatinin-ratio-test-manual-state: plan-state;
albumin-creatinin-ratio-in-urine: bool;
**declaration**

albumin-creatinin-ratio-test-manual# (pd, time,
albumin-creatinin-ratio-test-manual-state)
**var** albumin-creatinin-ratio-in-urine = ? **in**
**begin**
**var** $\text{time}_0$ = time[.], dt = ? **in await** $\text{time}_0$[.] + dt[.] $\leq$ time[.];
albumin-creatinin-ratio-in-urine := ?;
pd := set-albumin-creatinin-ratio-in-urine(pd[.], albumin-creatinin-ratio-in-urine[.]);
**var** $\text{time}_0$ = time[.], dt = ? **in await** $\text{time}_0$[.] + dt $\leq$ time[.];
albumin-creatinin-ratio-test-manual-state := completed
**end**
**end enrich**

---

Albumin-test-manual =
**enrich** Asbru-basic **with**
**procedures** albumin-test-manual#() : (patient-data, nat, plan-state);
**variables**
albumin-test-manual-state: plan-state;
albumin-in-urine: bool;
**declaration**

albumin-test-manual# (pd, time, albumin-test-manual-state)
**var** albumin-in-urine = ? **in**
**begin**
**var** $\text{time}_0$ = time[.], dt = ? **in await** $\text{time}_0$[.] + dt[.] $\leq$ time[.];
albumin-in-urine := ?;
pd := set-albumin-in-urine(pd[.], albumin-in-urine[.]);
**var** $\text{time}_0$ = time[.], dt = ? **in await** $\text{time}_0$[.] + dt[.] $\leq$ time[.];
albumin-test-manual-state := completed
**end**
**end enrich**

---

Change-insulin-to-2dd =
**enrich** Find-morning-insulin-dose **with**
**procedures**
change-insulin-to-2dd# () : (patient-data, nat, plan-state);
diabetes-specialist-referral# () : (patient-data, nat, plan-state);
**variables**
change-insulin-to-2dd-state, diabetes-specialist-referral-state: plan-state;
par-morning-insulin-dose: nat;
**declaration**

change-insulin-to-2dd# (pd, time, change-insulin-to-2dd-state)
**var** par-morning-insulin-dose = ?,
find-morning-insulin-dose-state = inactive,
diabetes-specialist-referral-state = inactive
**in**
**begin**
**await** pd[.].evening-dose > 40 $\wedge$ pd[.].fasting-glucose-monitoring $\neq$ bad
$\vee$ pd[.].postprandial-glucose-monitoring = bad;

```
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        pd := set-dmt2-treatment(pd[.], insulin);
        pd := set-dd(pd[.], 2);
        pd := set-evening-intake(pd[.], before-meal);
        par-morning-insulin-dose := ?;
        pd := set-morning-dose(pd[.], par-morning-insulin-dose[.]);
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        pd := set-morning-intake(pd[.], before-meal);
        find-morning-insulin-dose#(; pd, time, find-morning-insulin-dose-state);
        diabetes-specialist-referral#(; pd, time, diabetes-specialist-referral-state);
        if find-morning-insulin-dose-state[.] = completed then
          change-insulin-to-2dd-state := completed
        else
          change-insulin-to-2dd-state := aborted
      end;

      diabetes-specialist-referral# (pd, time, diabetes-specialist-referral-state)
      begin
        await pd[.].postprandial-glucose-monitoring = bad;
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        diabetes-specialist-referral-state := completed
      end
end enrich
```

---

```
Check-for-antidiabetic-problems =
enrich Asbru-basic with
    procedures check-for-antidiabetic-problems#() : (patient-data, nat, plan-state);
    variables
        check-for-antidiabetic-problems-state: plan-state;
        problematic-antidiabetic: drug;
        drug-dose, iterator: nat;
    declaration

      check-for-antidiabetic-problems# (pd, time, check-for-antidiabetic-problems-state)
      var problematic-antidiabetic = ?, drug-dose = ? in
      begin
        await pd[.].contraindications ∨ pd[.].side-effects;
        problematic-antidiabetic := ?;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        var iterator = 0 in
        begin
          while   pd[.].drugs[iterator[.]] ≠ problematic-antidiabetic[.]
                ∧ iterator[.] < # pd[.].drugs do
              iterator := iterator[.] + 1;
          pd := set-drugs(pd[.], (pd[.].drugs -1l iterator[.]) + acarbose);
          drug-dose := ?;
          var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
          pd := set-drugdoses(pd[.], (pd[.].drugdoses -1l iterator[.]) + drug-dose);
          check-for-antidiabetic-problems-state := completed
        end
      end
end enrich
```

---

```
Fasting-glucose-test =
enrich Asbru-basic with
    procedures fasting-glucose-test#() : (patient-data, nat, plan-state);
    variables
        fasting-glucose-test-state: plan-state;
        glucose-measurement-type: bool;
        fasting-glucose: nat;
    declaration

      fasting-glucose-test# (pd, time, fasting-glucose-test-state)
      var glucose-measurement-type = ?, fasting-glucose = ? in
      begin
            begin
              glucose-measurement-type := ?;
              pd := set-glucose-measurement-type(pd[.], glucose-measurement-type[.]);
              var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
            end
        ||a begin
              fasting-glucose := ?;
              pd := set-fasting-glucose(pd[.], fasting-glucose[.]);
              var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.]
            end;
          fasting-glucose-test-state := completed
      end
end enrich
```

---

```
Feet-examination =
enrich Asbru-basic with
    procedures feet-examination#() : (patient-data, nat, plan-state);
    variables feet-examination-state: plan-state;
    declaration

      feet-examination# (pd, time, feet-examination-state)
      begin
        skip;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        feet-examination-state := completed
      end
end enrich
```

---

```
Find-ACE-inhibitor-doses =
enrich Increase-drug-doses with
    procedures find-ace-inhibitor-doses#() : (patient-data, nat, plan-state);
    variables find-ace-inhibitor-doses-state: plan-state;
    declaration

      find-ace-inhibitor-doses# (pd, time, find-ace-inhibitor-doses-state)
      var increase-drug-doses-state = inactive in
      begin
        break
          while true do
              begin
                increase-drug-doses#(; pd, time, increase-drug-doses-state);
```

```
                var time0 = time[.], dt = ? in
                if 1 ≤ dt[.] ∧ dt[.] ≤ 3 then await time0[.] + dt[.] = time[.] else
                  false
              end
        if    pd[.].higher-blood-pressure ≤ 140 ∧ pd[.].lower-blood-pressure ≤ 80
          ∨ pd[.].ace-inhibitor-maximal-doses;
        find-ace-inhibitor-doses-state := completed
      end
end enrich
```

---

```
Find-antidiabetic-doses =
enrich Increase-drug-doses with
    procedures find-antidiabetic-doses#() : (patient-data, nat, plan-state);
    variables find-antidiabetic-doses-state: plan-state;
    declaration

      find-antidiabetic-doses# (pd, time, find-antidiabetic-doses-state)
      var increase-drug-doses-state = inactive in
      begin
        break
          while true do
              begin
                increase-drug-doses#(; pd, time, increase-drug-doses-state);
                var time0 = time[.], dt = ? in
                if 2 ≤ dt[.] ∧ dt[.] ≤ 4 then await time0[.] + dt[.] = time[.] else
                  false
              end
        if    pd[.].glucose-monitoring ≠ bad
          ∨ pd[.].antidiabetic-maximal-doses ∧ pd[.].glucose-monitoring = bad;
        if pd[.].glucose-monitoring ≠ bad then
          find-antidiabetic-doses-state := completed
        else
          find-antidiabetic-doses-state := aborted
      end
end enrich
```

---

```
Find-evening-insulin-dose =
enrich Test-glucose-and-adapt-evening-insulin with
    procedures find-evening-insulin-dose#() : (patient-data, nat, plan-state);
    variables find-evening-insulin-dose-state: plan-state;
    declaration

      find-evening-insulin-dose# (pd, time, find-evening-insulin-dose-state)
      var test-glucose-and-adapt-evening-insulin-state = inactive in
      begin
        break
          while true do
              begin
                test-glucose-and-adapt-evening-insulin#(; pd,
                                                         time,
                                                         test-glucose-and-adapt-evening-
insulin-state);
                var time0 = time[.], dt = ? in
```

```
                if 7 days ≤ dt[.] ∧ dt[.] ≤ 10 days then
                  await time0[.] + dt[.] = time[.]
                else
                  false
              end
        if pd[.].fasting-glucose-monitoring = good ∨ pd[.].evening-dose ≥ 40;
        if   test-glucose-and-adapt-evening-insulin-state[.] = completed
          ∧ pd[.].fasting-glucose-monitoring = good
        then
          find-evening-insulin-dose-state := completed
        else
          find-evening-insulin-dose-state := aborted
      end
end enrich
```

---

```
Find-insulin-doses =
enrich Test-glucose-and-adapt-insulin with
    procedures find-insulin-doses#() : (patient-data, nat, plan-state);
    variables find-insulin-doses-state: plan-state;
    declaration

      find-insulin-doses# (pd, time, find-insulin-doses-state)
      var test-glucose-and-adapt-insulin-state = inactive in
      break
        while true do
          begin
            test-glucose-and-adapt-insulin#(; pd,
                                              time,
                                              test-glucose-and-adapt-insulin-state);
            var time0 = time[.], dt = ? in
            if 2 days ≤ dt[.] ∧ dt[.] ≤ 3 days then
              await time0[.] + dt[.] = time[.]
            else
              false
          end;
        find-insulin-doses-state := completed
      if pd[.].glucose-monitoring ≠ bad
end enrich
```

---

```
Initialise-drug-doses =
enrich Asbru-basic with
    procedures initialise-drug-doses#() : (patient-data, nat, plan-state);
    variables
        initialise-drug-doses-state: plan-state;
        drug-dose, iterator: nat;
        drugdoses: drugdoselist;
    declaration

      initialise-drug-doses# (pd, time, initialise-drug-doses-state)
      begin
        var iterator = 0, drugdoses = @, drug-dose = ? in
        begin
          while iterator < # pd[.].drugs do
```

```
                begin
                  drug-dose := ?;
                  var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
                  drugdoses := drugdoses[.] + drug-dose[.]
                end;
            pd := set-drugdoses(pd[.], drugdoses[.])
          end;
          initialise-drug-doses-state := completed
      end
end enrich
```

---

```
Find-morning-insulin-dose =
enrich Test-glucose-and-adapt-morning-insulin with
    procedures find-morning-insulin-dose#() : (patient-data, nat, plan-state);
    variables find-morning-insulin-dose-state: plan-state;
    declaration

      find-morning-insulin-dose# (pd, time, find-morning-insulin-dose-state)
      var test-glucose-and-adapt-morning-insulin-state = inactive in
      begin
        break
          while true do
              begin
                test-glucose-and-adapt-morning-insulin#(; pd,
                                                         time,
                                                         test-glucose-and-adapt-morning-
insulin-state);
                var time0 = time[.], dt = ? in
                if 7 days ≤ dt[.] ∧ dt[.] ≤ 10 days then
                  await time0[.] + dt[.] = time[.]
                else
                  false
              end
        if pd[.].postprandial-glucose-monitoring = good;
        if test-glucose-and-adapt-morning-insulin-state[.] = completed then
          find-morning-insulin-dose-state := completed
        else
          find-morning-insulin-dose-state := aborted
      end
end enrich
```

---

```
Increase-drug-doses =
enrich Asbru-basic with
    procedures increase-drug-doses#() : (patient-data, nat, plan-state);
    variables
        increase-drug-doses-state: plan-state;
        maximal-drug-doses, drug-dose-increase, iterator: nat;
        aux-doses: drugdoselist;
    declaration

      increase-drug-doses# (pd, time, increase-drug-doses-state)
      var maximal-drug-doses = ?, drug-dose-increase = ? in
      begin
```

```
        maximal-drug-doses := ?;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        var iterator = 0, aux-doses = @ in
        begin
          while iterator < # pd[.].drugs do
              begin
                drug-dose-increase := ?;
                var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
                aux-doses := aux-doses[.] + pd[.].drugdoses[iterator] + drug-dose-increase[.]
              end;
          pd := set-drugdoses(pd[.], aux-doses[.])
        end;
        increase-drug-doses-state := completed
    end
end enrich
```

---

```
Test-glucose-and-adapt-evening-insulin =
enrich Four-points-day-curve with
    procedures test-glucose-and-adapt-evening-insulin#() : (patient-data, nat, plan-state);
    variables
        test-glucose-and-adapt-evening-insulin-state: plan-state;
        evening-adaptation: nat;
    declaration

      test-glucose-and-adapt-evening-insulin# (pd, time,
test-glucose-and-adapt-evening-insulin-state)
      var evening-adaptation = ?, four-points-day-curve-state = inactive in
      begin
        evening-adaptation := ?;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        pd := set-evening-dose(pd[.], evening-adaptation[.] + pd[.].evening-dose);
        four-points-day-curve#(; pd, time, four-points-day-curve-state);
        if four-points-day-curve-state[.] = completed then
          test-glucose-and-adapt-evening-insulin-state := completed
        else
          test-glucose-and-adapt-evening-insulin-state := aborted
      end
end enrich
```

---

```
Test-glucose-and-adapt-insulin =
enrich Four-points-day-curve with
    procedures test-glucose-and-adapt-insulin#() : (patient-data, nat, plan-state);
    variables
        test-glucose-and-adapt-insulin-state: plan-state;
        evening-adaptation, morning-adaptation: nat;
    declaration

      test-glucose-and-adapt-insulin# (pd, time, test-glucose-and-adapt-insulin-state)
      var evening-adaptation = ?,
          morning-adaptation = ?,
          four-points-day-curve-state = inactive
      in
      begin
```

```
        four-points-day-curve#(; pd, time, four-points-day-curve-state);
        evening-adaptation := ?;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        pd := set-evening-dose(pd[.], evening-adaptation[.] + pd[.].evening-dose);
        morning-adaptation := ?;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        pd := set-morning-dose(pd[.], morning-adaptation[.] + pd[.].morning-dose);
        if four-points-day-curve-state[.] = completed then
          test-glucose-and-adapt-insulin-state := completed
        else
          test-glucose-and-adapt-insulin-state := aborted
      end
end enrich
```

---

```
Test-glucose-and-adapt-morning-insulin =
enrich Four-points-day-curve with
    procedures test-glucose-and-adapt-morning-insulin#() : (patient-data, nat, plan-state);
    variables
        test-glucose-and-adapt-morning-insulin-state: plan-state;
        morning-adaptation: nat;
    declaration

      test-glucose-and-adapt-morning-insulin# (pd, time,
test-glucose-and-adapt-morning-insulin-state)
      var morning-adaptation = ?, four-points-day-curve-state = inactive in
      begin
        morning-adaptation := ?;
        var time0 = time[.], dt = ? in await time0[.] + dt[.] ≤ time[.];
        pd := set-morning-dose(pd[.], morning-adaptation[.] + pd[.].morning-dose);
        four-points-day-curve#(; pd, time, four-points-day-curve-state);
        if four-points-day-curve-state[.] = completed then
          test-glucose-and-adapt-morning-insulin-state := completed
        else
          test-glucose-and-adapt-morning-insulin-state := aborted
      end
end enrich
```

---

```
Four-points-day-curve =
enrich Asbru-basic with
    procedures four-points-day-curve#() : (patient-data, nat, plan-state);
    variables
        four-points-day-curve-state: plan-state;
        fasting-glucose, after-breakfast-glucose, after-lunch-glucose, after-dinner-glucose: nat;
    declaration

      four-points-day-curve# (pd, time, four-points-day-curve-state)
      var fasting-glucose = ?,
          after-breakfast-glucose = ?,
          after-lunch-glucose = ?,
          after-dinner-glucose = ?
      in
      begin
        fasting-glucose := ?;
```

pd := set-fasting-glucose(pd[.], fasting-glucose[.]);
**var** $time_0$ = time[.], dt = ? **in await** $time_0$[.] + dt[.] ≤ time[.];
after-breakfast-glucose := ?;
pd := set-after-breakfast-glucose(pd[.], after-breakfast-glucose[.]);
**var** $time_0$ = time[.], dt = ? **in await** $time_0$[.] + dt[.] ≤ time[.];
after-lunch-glucose := ?;
pd := set-after-lunch-glucose(pd[.], after-lunch-glucose[.]);
**var** $time_0$ = time[.], dt = ? **in await** $time_0$[.] + dt[.] ≤ time[.];
after-dinner-glucose := ?;
pd := set-after-dinner-glucose(pd[.], after-dinner-glucose[.]);
**var** $time_0$ = time[.], dt = ? **in await** $time_0$[.] + dt[.] ≤ time[.];
four-points-day-curve-state := completed

**end**

**end enrich**

---

Asbru-basic = plan-state + patient-data + time

---

patient-data =
**enrich** patient-data-basic **with**
**functions**

set-glucose-measurement-type : patient-data × bool → patient-data ;
set-fasting-glucose : patient-data × nat → patient-data ;
set-non-fasting-glucose : patient-data × nat → patient-data ;
set-after-breakfast-glucose : patient-data × nat → patient-data ;
set-after-lunch-glucose : patient-data × nat → patient-data ;
set-after-dinner-glucose : patient-data × nat → patient-data ;
set-albumin-creatinin-ratio-in-urine : patient-data × bool → patient-data ;
set-albumin-in-urine : patient-data × bool → patient-data ;
set-dmt2-in-1sr-grade-relatives : patient-data × bool → patient-data ;
set-hypertension : patient-data × bool → patient-data ;
set-cv-diseases : patient-data × bool → patient-data ;
set-fat-metabolism-problems : patient-data × bool → patient-data ;
set-quetelet-index : patient-data × nat → patient-data ;
set-ethnic-group : patient-data × ethnic-group → patient-data ;
set-dm-in-past : patient-data × bool → patient-data ;
set-newborns-biggerthan-4kg : patient-data × bool → patient-data ;
set-risk-factors : patient-data × bool → patient-data ;
set-thirst : patient-data × bool → patient-data ;
set-polyuria : patient-data × bool → patient-data ;
set-weight-loss : patient-data × bool → patient-data ;
set-pruritis-vulvae : patient-data × bool → patient-data ;
set-typical-signs : patient-data × bool → patient-data ;
set-glucose-determination-needed : patient-data × bool → patient-data ;
set-wellbeing : patient-data × bool → patient-data ;
set-hypoglycemia : patient-data × bool → patient-data ;
set-hypoglycemic-coma-over : patient-data × bool → patient-data ;
set-feeding-problems : patient-data × bool → patient-data ;
set-treatment-problems : patient-data × bool → patient-data ;
set-weight : patient-data × nat → patient-data ;
set-feet-problems : patient-data × bool → patient-data ;
set-total-cholesterol : patient-data × nat → patient-data ;
set-hdl-cholesterol : patient-data × nat → patient-data ;
set-triglycerids : patient-data × bool → patient-data ;
set-visus-problems : patient-data × bool → patient-data ;
set-cv-problems : patient-data × bool → patient-data ;
set-neuropathy : patient-data × bool → patient-data ;
set-sexual-problems : patient-data × bool → patient-data ;
set-higher-blood-pressure : patient-data × nat → patient-data ;
set-lower-blood-pressure : patient-data × nat → patient-data ;
set-ghb : patient-data × nat → patient-data ;
set-creatinin : patient-data × bool → patient-data ;
set-morning-dose : patient-data × nat → patient-data ;
set-evening-dose : patient-data × nat → patient-data ;
set-dmt2-treatment : patient-data × treatment → patient-data ;
set-insulin-type : patient-data × insulin-type → patient-data ;
set-insulin-dd : patient-data × nat → patient-data ;
set-dd : patient-data × nat → patient-data ;
set-morning-insulin-dose : patient-data × nat → patient-data ;
set-morning-insulin-intake : patient-data × intake → patient-data ;
set-morning-intake : patient-data × intake → patient-data ;
set-evening-insulin-dose : patient-data × nat → patient-data ;
set-evening-insulin-intake : patient-data × intake → patient-data ;
set-evening-intake : patient-data × intake → patient-data ;
set-drugs : patient-data × druglist → patient-data ;
set-drugdoses : patient-data × drugdoselist → patient-data ;
set-ch-diseases : patient-data × bool → patient-data ;
set-ch-disease-in-youngerthan-60-1st-grade-relatives : patient-data × bool → patient-data ;
set-feeding-habits : patient-data × bool → patient-data ;
set-smoking : patient-data × bool → patient-data ;
set-alcohol-consuming : patient-data × bool → patient-data ;
set-physical-exercise : patient-data × bool → patient-data ;

**predicates**

| | | |
|---|---|---|
| eval-risk-factors | : | patient-data; |
| eval-typical-signs | : | patient-data; |
| eval-glucose-determination-needed | : | patient-data; |

**axioms**

eval-risk-factors :
eval-risk-factors(pd)
↔ pd.dmt2-in-1st-grade-relatives
∨ pd.hypertension
∨ pd.cv-diseases
∨ pd.fat-metabolism-problems
∨ pd.quetelet-index ≥ 27
∨ pd.ethnic-group = hindustanian
∨ pd.sex = female ∧ (pd.dm-in-past ∨ pd.newborns-biggerthan-4kg);

eval-typical-signs :
eval-typical-signs(pd)
↔ pd.thirst
∨ pd.polyuria
∨ pd.weight-loss
∨ pd.age > 60 ∧ pd.sex = female ∧ pd.pruritis-vulvae;

eval-glucose-determination-needed :
eval-glucose-determination-needed(pd) ↔ pd.risk-factors ∨ pd.typical-signs;

**end enrich**

---

patient-data-basic =
**data specification**
**using** bool, treatment, sex, ethnic-group, glucose-monitoring, intake, druglist, drugdoselist, evaluation, insulin-type,
patient-data = mk-patient-data (. .alive : bool ;
. .age : nat ;
. .sex : sex ;
. .ethnic-group : ethnic-group ;
. .dm-in-past : bool ;
. .dmt2-treatment : treatment ;
. .dmt2-in-1st-grade-relatives : bool ;
. .cholesterol-treatment : bool ;
. .newborns-biggerthan-4kg : bool ;
. .glucose-measurement-type : bool ;
. .fasting-glucose : nat ;
. .after-breakfast-glucose : nat ;
. .after-lunch-glucose : nat ;
. .after-dinner-glucose : nat ;
. .self-monitoring-training-needed : bool ;
. .insulin-treatment-start : bool ;
. .insulin-adjustment-needed : bool ;
. .glucose-monitoring : glucose-monitoring ;
. .postprandial-glucose-monitoring : glucose-monitoring ;
. .fasting-glucose-monitoring : glucose-monitoring ;
. .insulin-dd : nat ;
. .diabetes-ulcus-after-2-weeks : bool ;
. .creatinin : nat ;

. .creatinin-clearance : nat ;
. .lethargy : bool ;
. .dehydration : bool ;
. .vomiting : bool ;
. .hypoglycemic-coma-after-30-min : bool ;
. .hypoglycemic-coma : bool ;
. .hypoglycemic-coma-over : bool ;
. .pressure-points : bool ;
. .bad-standing-habits : bool ;
. .triennial-visit : bool ;
. .hypertension : bool ;
. .cv-diseases : bool ;
. .fat-metabolism-problems : bool ;
. .quetelet-index : nat ;
. .risk-factors : bool ;
. .thirst : bool ;
. .polyuria : bool ;
. .weight-loss : bool ;
. .pruritis-vulvae : bool ;
. .typical-signs : bool ;
. .feet-problems : bool ;
. .morning-dose : nat ;
. .evening-dose : nat ;
. .drugs : druglist ;
. .drugdoses : drugdoselist ;
. .contraindications : bool ;
. .side-effects : bool ;
. .glucose-determination-needed : bool ;
. .glucose-evaluation : evaluation ;
. .higher-blood-pressure : nat ;
. .lower-blood-pressure : nat ;
. .ace-inhibitor-maximal-doses : bool ;
. .antidiabetic-maximal-doses : bool ;
. .microalbuminuria : bool ;
. .fever : bool ;
. .diarrhea : bool ;
. .life-expectancy : nat ;
. .ch-disease-risk : risk ;
. .smoking : bool ;
);

**variables** pd, $pd_1$, $pd_2$, $pd_3$: patient-data;

**end data specification**

---

drugdoselist =

**actualize** list-del **with** nat-div **by morphism**

list → drugdoselist; elem → nat; -$1_l$ → -$1_p$; + →. ++ . **prio** 9 **left**; x → drgds; y → $drgds_1$; z → $drgds_2$; $x_0$ → $drgds_3$; $y_0$ → $drgds_4$; $z_0$ → $drgds_5$; $x_1$ → $drgds_6$; $y_1$ → $drgds_7$; $z_1$ → $drgds_8$; $x_2$ → $drgds_9$; $y_2$ → $drgds_{10}$; $z_2$ → $drgds_{11}$; a → drgd; $a_0$ → $drgd_0$; b → $drgd_1$; c → $drgd_2$

**end actualize**

---

druglist =

**actualize** list-del **with** drug **by morphism**

list → druglist; elem → drug; x → drgs; y → $drgs_1$; z → $drgs_2$; $x_0$ → $drgs_3$; $y_0$ → $drgs_4$; $z_0$ → $drgs_5$; $x_1$ → $drgs_6$; $y_1$ → $drgs_7$; $z_1$ → $drgs_8$; $x_2$ → $drgs_9$; $y_2$ → $drgs_{10}$; $z_2$ → $drgs_{11}$; a → drg; $a_0$ → $drg_0$; b → $drg_1$; c → $drg_2$
**end actualize**

---

ethnic-group =
**data specification**
ethnic-group = hindustanian
| other
;
**variables** eth, $eth_1$, $eth_2$, $eth_3$: ethnic-group;
**end data specification**

---

evaluation =
**data specification**
evaluation = $dmt_2$
| other
;
**variables** eval, $eval_1$, $eval_2$, $eval_3$: evaluation;
**end data specification**

---

glucose-monitoring =
**data specification**
glucose-monitoring = good
| bad
| other
;
**variables** glm, $glm_1$, $glm_2$, $glm_3$: glucose-monitoring;
**end data specification**

---

insulin-type =
**data specification**
insulin-type = mid-term
| other
;
**variables** inst, $inst_1$, $inst_2$, $inst_3$: insulin-type;
**end data specification**

---

intake =
**data specification**
intake = before-meal
| after-meal
;
**variables** intk, $intk_1$, $intk_2$, $intk_3$: intake;
**end data specification**

---

risk =
**data specification**
    risk = significant
         | other
         ;
    **variables** risk, $risk_1$, $risk_2$, $risk_3$: risk;
**end data specification**

---

sex =
**data specification**
    sex = male
        | female
        ;
    **variables** sx, $sx_1$, $sx_2$, $sx_3$: sex;
**end data specification**

---

treatment =
**data specification**
    treatment = insulin
              | insulin-plus-antidiabetics
              | antidiabetics
              ;
    **variables** treat, $treat_1$, $treat_2$, $treat_3$: treatment;
**end data specification**

---

drug =
**specification**
    **sorts** drug;
    **constants**
        acarbose : drug;
        metformin : drug;

    **variables** drg, $drg_1$, $drg_2$, $drg_3$: drug;
**end specification**

---

list-del =
**enrich** list-last **with**
    **functions**

| | | | | | |
|---|---|---|---|---|---|
| . -$_l$ . | : | list × elem | → | list | **prio** 9; |
| . -1$_l$ . | : | list × elem | → | list | **prio** 9; |
| . -1$_l$ . | : | list × nat | → | list | **prio** 9; |
| . [ . ] | : | list × nat | → | elem | **prio** 2; |
| pos | : | elem × list | → | nat | ; |
| . [ . ] | : | list × nat × elem | → | list | ; |
| sublist | : | nat × nat × list | → | list | ; |
| firstn | : | nat × list | → | list | ; |
| restn | : | nat × list | → | list | ; |
| lastn | : | nat × list | → | list | ; |
| frome | : | list × elem | → | list | ; |

**axioms**

del-e : @ -$_l$ a = @;
del-y : (a ' + x) -$_l$ a = x -$_l$ a;
del-n : a ≠ b → (b ' + x) -$_l$ a = b ' + x -$_l$ a;
del1-e : @ -1$_l$ a = @;
del1-y : (a ' + x) -1$_l$ a = x;
del1-n : a ≠ b → (b ' + x) -1$_l$ a = b ' + x -1$_l$ a;
delpos-empty : @ -1$_l$ n = @;
delpos-base : (a ' + x) -1$_l$ 0 = x;
delpos-rec : (a ' + x) -1$_l$ n + 1 = a + x -1$_l$ n;
get-zero : a ' + x[0] = a;
get-succ : a ' + x[n + 1] = x[n];
pos-e : pos(a, @) = 0;
pos-y : pos(a, a ' + x) = 0;
pos-n : a ≠ b → pos(a, b ' + x) = pos(a, x) + 1;
put-zero : b ' + x[0, a] = a + x;
put-succ : b ' + x[n + 1, a] = b + x[n, a];
sublist-good : sublist(# x, # y, x + y + z) = y;
firstN-zero : firstn(0, x) = @;
firstN-rec : firstn(n + 1, x) = x.first + firstn(n, x.rest);
restN-zero : restn(0, x) = x;
restN-rec : restn(n + 1, x) = restn(n, x.rest);
lastN-zero : lastn(0, x) = @;
lastN-rec : lastn(n + 1, x) = lastn(n, x.butlast) + x.last;
fromE-empty : frome(@, a) = @;
fromE-yes : frome(a ' + x, a) = a ' + x;
fromE-no : a ≠ b → frome(a ' + x, b) = frome(x, b);

**end enrich**

---

list-last =
**enrich** list-dup **with**
**functions**

| | | | | | |
|---|---|---|---|---|---|
| . .last | : | list | → | elem | ; |
| . .butlast | : | list | → | list | ; |
| butlastn | : | nat × list | → | list | ; |
| rev | : | list | → | list | ; |
| mklist | : | elem × nat | → | list | ; |
| fillfirst | : | nat × elem × list | → | list | ; |

**predicates**

. ⊑ . : list × list;
. ⊒ . : list × list;

**axioms**

last : $x \neq @ \rightarrow x.butlast + x.last = x$;

rev-e : $rev(@) = @$;

rev-r : $rev(a\ ' + x) = rev(x) + a$;

mk-len : $\#\ mklist(a, n) = n$;

mk-elem : $a \in mklist(b, n) \rightarrow a = b$;

butlastN-base : $butlastn(0, x) = x$;

butlastN-rec : $butlastn(n + 1, x) = butlastn(n, x.butlast)$;

fillfirst-longer : $n \leq \#\ x \rightarrow fillfirst(n, a, x) = x$;

fillfirst-fill : $\#\ x < n \rightarrow fillfirst(n, a, x) = mklist(a, n - \#\ x) + x$;

prefix : $x \sqsubseteq y \leftrightarrow (\exists\ z.\ x + z = y)$;

postfix : $x \sqsupseteq y \leftrightarrow (\exists\ z.\ z + x = y)$;

**end enrich**

---

list-dup =
**enrich** list **with**
**functions**

| | | | | | |
|---|---|---|---|---|---|
| . ++ . | : | list × elem | → | list | **prio** 9 **left**; |
| rmdup | : | list | → | list | ; |

**predicates**

| | | |
|---|---|---|
| dups | : | list; |
| disj | : | list × list; |

**axioms**

rmdup-e : $rmdup(@) = @$;

rmdup-y : $a \in x \rightarrow rmdup(a\ ' + x) = rmdup(x)$;

rmdup-n : $\neg\ a \in x \rightarrow rmdup(a\ ' + x) = a + rmdup(x)$;

adjoin-in : $a \in x \rightarrow x +\!+ a = x$;

adjoin-notin : $\neg\ a \in x \rightarrow x +\!+ a = a + x$;

dups : $dups(x) \leftrightarrow (\exists\ a, x_0, y, z.\ x = x_0 + a + y + a + z)$;

disjoint : $disj(x, y) \leftrightarrow (\forall\ a.\ \neg\ (a \in x \wedge a \in y))$;

**end enrich**

---

list =
**enrich** list-data **with**
**functions**

| | | | | | |
|---|---|---|---|---|---|
| . ' | : | elem | → | list | ; |
| . + . | : | list × list | → | list | **prio** 9; |
| . + . | : | list × elem | → | list | **prio** 9; |
| . + . | : | elem × elem | → | list | **prio** 9; |

**predicates** . ∈ . : elem × list;

**axioms**

Nil : @ + x = x;

Cons : (a + x) + y = a + x + y;

One : a ' = a + @;

Last : x + a = x + a ';

Two : a + b = a ' + b ';

in : a ∈ x ↔ (∃ y, z. x = y + a + z);

**end enrich**

---

list-data =
**generic data specification**
**parameter** elem **using** nat
list = @
| . + . **prio** 9 (. .first : elem ; . .rest : list ;) **prio** 9
;
**variables** x, y, z, $x_0$, $y_0$, $z_0$, $x_1$, $y_1$, $z_1$, $x_2$, $y_2$, $z_2$: list;
**size functions** # . : list → nat ;
**order predicates** . < . : list × list;
**end generic data specification**

---

elem =
**specification**
**sorts** elem;
**variables** a, b, c: elem;
**end specification**

---

**DELIVERABLES TABLE**

| | |
|---|---|
| **Project Number:** | ***IST–2001-33049*** |
| **Project Acronym:** | **PROTOCURE** |
| **Title:** | ***Improving medical protocols by formal methods*** |

| Del. No. | Revision | Title | Type[1] | Classification[2] | Due Date | Issue Date |
|---|---|---|---|---|---|---|
| **D0** | **1** | **Project presentation** | **O*** | **Pub.** | **28/2/2002** | **26/2/2002** |
| **D1** | **1** | **Reference protocols and their Asbru models** | **R** | **Pub.** | **28/2/2002** | **28/2/2002** |
| **D2** | **1** | **Formal semantics of main Asbru elements** | **R** | **Pub.** | **31/3/2002** | **31/3/2002** |
| **D3** | **1** | **Desirable/required properties of main Asbru elements** | **R** | **Pub.** | **31/3/2002** | **31/3/2002** |
| **D3'** | **1** | **KIV formalisation** | **D** | **Pub.** | **31/5/2002** | **31/5/2002** |
| **D4** | | **Verification of properties on the reference protocols** | **R** | **Pub.** | **30/9/2002** | |
| **D5** | | **Evaluation of results** | **R** | **Pub.** | **30/11/2002** | |

[1] *R: Report; D: Demonstrator; S: Software; W: Workshop; O: Other – Specify in footnote*

[2] *Int.: Internal circulation within project (and Commission Project Officer + reviewers if requested)*
*Rest.: Restricted circulation list (specify in footnote) and Commission SO + reviewers only*
*IST: Circulation within IST Programme participants*
*FP5: Circulation within Framework Programme participants*
*Pub.: Public document*

* Report, webpage.