

# Automatisk bestemmelse af formatunderstøttelse

## 1. grammatik

151082 | Søren Vrist (seet@diku.dk)  
260576 | Sune Juel Jensen (di050504@diku.dk)

9. marts 2006

En typebeskrivelse, f.eks. pdf, ps, xp, word97, dvs. alle de forskellige ting der kan stå inden i byggeklodserne.

$\langle \text{formatId} \rangle ::= \text{formatid}$

Mushroom typen hvor formatId beskriver "formatet" af den pågældende dokument.

$\langle \text{App piece} \rangle ::= \text{'App('} \langle \text{formatId} \rangle \text{'}'$

Beskrivelse af en "converter" hvor de tre formatId beskriver fra venstre til højre hhv. venstre, bund, og højre side af converteren.

$\langle \text{Conv piece} \rangle ::= \text{'Conv('} \langle \text{formatId} \rangle \text{' ; '} \langle \text{formatId} \rangle \text{' ; '} \langle \text{formatId} \rangle \text{'}'$

Beskrivelse af en fortolker. De to formatId beskriver hhv top og bunden af fortolkeren.

$\langle \text{Intr piece} \rangle ::= \text{'Intr('} \langle \text{formatId} \rangle \text{' ; '} \langle \text{formatId} \rangle \text{'}'$

En maskine der kan forstå en "visning". Type descr beskriver formatet af maskinen. Det klassiske eksempel er xp, mips, x86 osv.

$\langle \text{Mach piece} \rangle ::= \text{'Mach('} \langle \text{formatId} \rangle \text{'}'$

Instansiering af de forskellige "byggeklodser". Bemærk at der er separate navnerum til de forskellige klodser.

$\langle \text{App decl} \rangle ::= \langle \text{App piece} \rangle \text{ App-id}$

$\langle \text{Conv decl} \rangle ::= \langle \text{Conv piece} \rangle \text{ Conv-id}$

## T-diagram grammatik

---

$\langle \text{Intr decl} \rangle ::= \langle \text{Intr piece} \rangle \text{Intr-id}$

$\langle \text{Mach decl} \rangle ::= \langle \text{Mach piece} \rangle \text{Mach-id}$

Man kan lave en `App` enten ved en direkte reference, eller ved at konverte en `App` med en `Conv` og en række af nul eller flere fortolkerer som er rodfæstet. Dvs. første argument til `conv` er venstre paddehat og `Anchor` sættes på bunden af `T`'et og `Conv` er beskrivelsen af den konverter der faciliterer konverteringen. Ascriptionen navngiver "outputtet" fra konverteringen som en ny paddehat.

$\langle \text{App} \rangle ::= \text{App-id}$   
| 'app-convert ' $\langle \text{App} \rangle$ ' with ' $\langle \text{Conv} \rangle$ ' rooted by ' $\langle \text{Anchor} \rangle$ ' as ' $\text{App-id}$ '

En `Conv` er enten en direkte reference til en konverter, eller en konversion af en `Conv` ved hjælp af en rodfæstning og en `Conv`. Dvs. første argument en input til tredje argument, hvor tredje argument er rodfæstet med andet argument. Ascriptionen navngiver den resulterende `Conv`.

$\langle \text{Conv} \rangle ::= \text{Conv-id}$   
| 'conv-convert ' $\langle \text{Conv} \rangle$ ' with ' $\langle \text{Conv} \rangle$ ' rooted by ' $\langle \text{Anchor} \rangle$ ' as ' $\text{App-id}$ '

En `Intr` er enten en direkte reference til en fortolker eller en fortolkning af en fortolker. Første argument bliver sat på bunden af andet argument. Dvs. at en `Intr` har hverken top eller bund.

$\langle \text{Intr} \rangle ::= \text{Intr-id}$   
| 'combine ' $\text{Intr-id}$ ' with ' $\langle \text{Intr} \rangle$ '

Et `Anchor` er en rodfæstet række af nul eller flere fortolkerer. Dvs. enten en maskine eller en maskine på bunden af en række fortolkerer.

$\langle \text{Anchor} \rangle ::= \text{'root' } \langle \text{Intr} \rangle \text{' with ' } \langle \text{Anchor} \rangle \text{' as 'Mach-id}$   
| Mach-id

Et program består af en række deklARATIONER, nul eller flere pre-diagramdele<sup>1</sup>, og tilsidst det endelige diagram. DeklARATIONERNE laves som en sekvens af deklARATIONER adskilt af semikolon og deklARATIONERNE adskilles fra selve programmet med `--`.

$\langle \text{Pgm} \rangle ::= \langle \text{Decls} \rangle \text{'--' } \langle \text{Pres} \rangle \text{run}(\langle \text{Body} \rangle)$

$\langle \text{Decls} \rangle ::= \langle \text{Decl} \rangle$   
|  $\langle \text{Decl} \rangle ; \langle \text{Decls} \rangle$

---

<sup>1</sup>Kan evt. ses som makroer?

## T-diagram grammatik

---

$$\begin{aligned} \langle Decl \rangle ::= & \langle App\ decl \rangle \\ & | \langle Conv\ decl \rangle \\ & | \langle Intr\ decl \rangle \\ & | \langle Mach\ decl \rangle \end{aligned}$$

Kroppen består, enten af et deldiagram eller en forankret App.

$$\begin{aligned} \langle Body \rangle ::= & \langle Pre \rangle \\ & | \langle App \rangle \text{ with } \langle Anchor \rangle \end{aligned}$$

Et deldiagram er de forskellige byggeklodser som kan laves med App, Conv, Anchor, Intr. Dvs. både enkelte byggeklodser, eller lovlige sammensætninger af byggeklodser. Flere deldiagrammer adskilles af semikolon.

$$\begin{aligned} \langle Pre \rangle ::= & \langle App \rangle \\ & | \langle Conv \rangle \\ & | \langle Anchor \rangle \\ & | \langle Intr \rangle \end{aligned}$$
$$\begin{aligned} \langle Pres \rangle ::= & \langle Pre \rangle ; \\ & | \langle Pre \rangle ; \langle Pres \rangle \\ & | \end{aligned}$$

## A Diagrammer

### A.1 d1.b

```
# $Id: d1.b,v 1.2 2006/03/19 22:15:29 seet Exp $
#
# |-----|
# \ p.g /
# |gif|
# -----
# |gif|
# |ie6|
# |xp |
# -----
# \xp /
# \ /

# Deklarationer
App(gif) photo.gif;
Intr(gif,xp) ie6;
Mach(xp) windows;
--
# <Pres>
combine ie6 with windows as ie6xp; # <Anchor>, as in pair internet explorer with IE6
# to create an anchor

# <Body>
run(
  photo.gif with ie6xp # Put anchor on the bottom of photo.gif to 'show'
# the photo.gif image
)
```



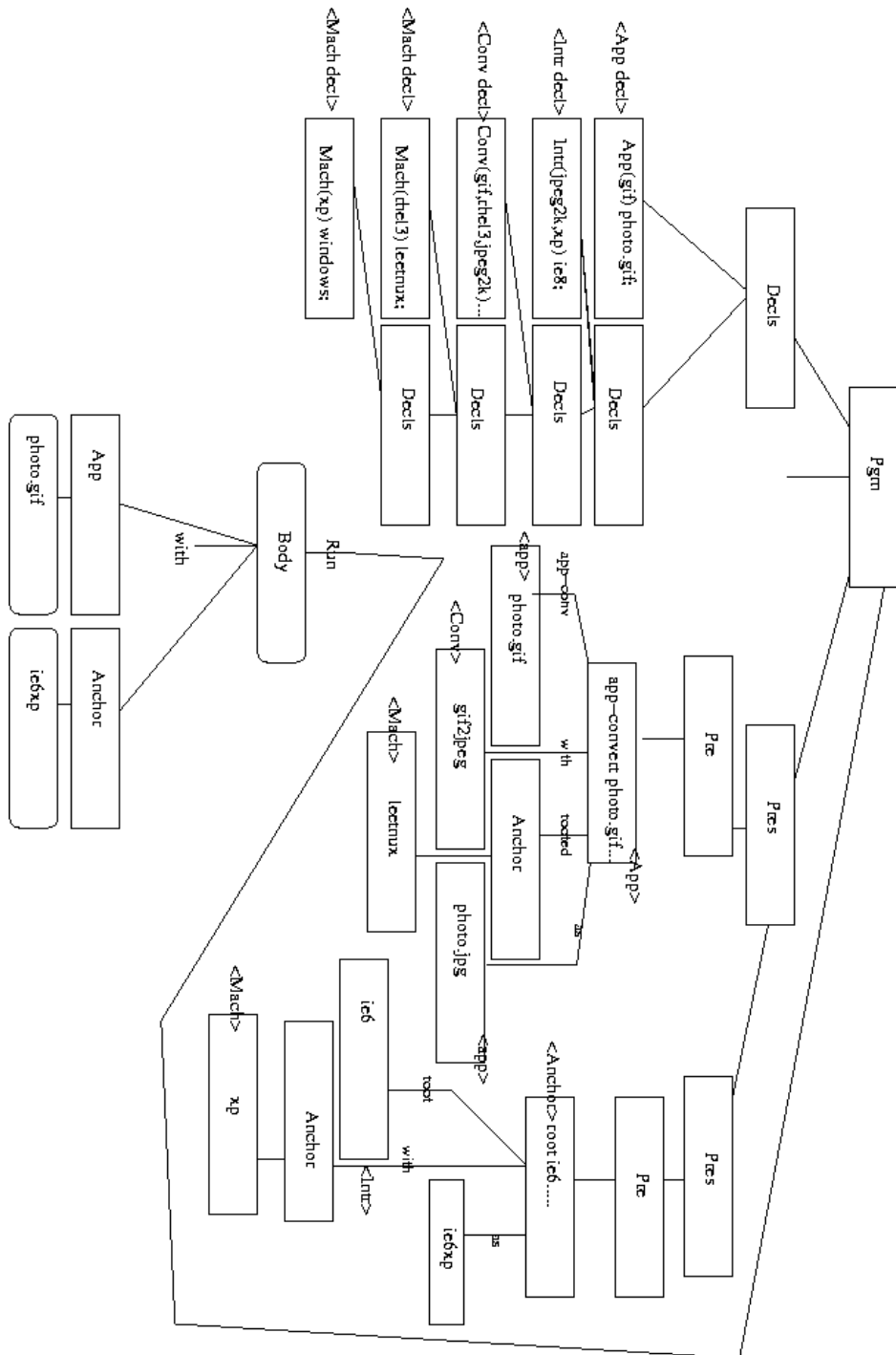
**A.2 d2.b**

```
# $Id: d2.b,v 1.3 2006/03/27 22:33:36 seet Exp $
#
# No Ascii art!
#
# Convert photo.gif on a redhat to photo.jpg with gif2jpg and show it in IE8
# on XP.

# Deklarationer
App(gif) photo.gif;
Intr(jpeg2k, xp) ie8;
Conv(gif, rhel3, jpeg2k) gif2jpg;
Mach(xp) windows;
Mach(rhel3) leetnux;
--
# <Pres>

app-convert photo.gif with gif2jpg rooted by leetnux as photo.jpg;
root ie8 with xp as ie8xp;

# <Body>
run(
  photo.jpg with ie8xp;
)
```



### A.3 d3.b

```
# $Id: d3.b,v 1.3 2006/03/27 22:38:42 seet Exp $
#
# No Ascii art!
#
# Convert photo.gif on a redhat to photo.jpg with gif2jpg and
# convert photo.jpg to photo.fif on osxx with jpg2fif and
# show it in IE100 on osXX.

# Deklarationer
App(gif) photo.gif;

Conv(gif,rhel3,jpeg2k) gif2jpg;
Conv(jpeg2k,rhel3,fif) jpg2fif;

Mach(xp) windows;
Mach(rhel3) leetnux;
Mach(osxx) mac;

Intr(fif,osxx) iel00;

--
# <Pres>
app-convert photo.gif with gif2jpg rooted by leetnux as photo.jpg;
app-convert photo.jpg with jpg2fif rooted by mac as photo.fif;

root iel00 with osxx as iel00osxx;

# <Body>
run(
  photo.fif with iel00osxx
)
```

### A.4 d4-1.b

```
# $Id: d4-1.b,v 1.3 2006/03/27 22:33:36 seet Exp $
#
# No Ascii art!
#
# Show doc.pdf with with a pdfviewer written i uvc and run uvc on java with
# a uvc2java interpreter.
#
# Deklarationer

App(pdf) doc.pdf;

Intr(pdf, uvc) pdfview.uvc;
Intr(uvc, java) uvcmach.java;

Mach(java) jvm;
--
# <Pres>
combine pdfview.uvc with uvcmach.java as pdfview.java;
root pdfview.java with jvm as uvcmach;

# <Body>
run(
  doc.pdf with pdfview
)
```

**A.5 d4-2.b**

```
# $Id: d4-2.b,v 1.3 2006/03/27 22:38:42 seet Exp $
#
# No Ascii art!
#
# Show doc.pdf with with a pdfviewer written i uvc and run uvc on borneo with
# a uvc2borneo interpreter.

# Deklarationer

App(pdf) doc.pdf;

Intr(pdf, uvc) pdfview.uvc;
Intr(uvc,borneo) uvcmach.borneo;

Mach(borneo) bvm;
--
# <Pres>
combine pdfview.uvc with uvcmach.borneo as pdfview.borneo;
root pdfview.borneo with bvm as pdfview;

# <Body>
run(
  doc.pdf with pdfview;
)
```

**A.6 d5-1.b**

```
# $Id: d5-1.b,v 1.3 2006/03/27 22:38:42 seet Exp $
#
# No Ascii art!
#
# Convert doc.pdf to doc.tm with a converter written in java and run doc.tm on jvm
# with a tm implementation in java.

# Deklarationer

App(pdf) doc.pdf;

Intr(tm, java) turing.java;

Conv(pdf, java,tm) pdf2tm;

Mach(java) jvm;
--
# <Pres>
convert doc.pdf with pdf2tm rooted by jvm as doc.tm;

root turing.java with jvm as turing;

# <Body>
run(
  doc.tm with turing
)
```

**A.7 d5-2.b**

```
# $Id: d5-2.b,v 1.3 2006/03/27 22:38:42 seet Exp $
#
# No Ascii art!
#
```

```
# Display the oooooold doc.tm with a turing-interpreter written in borneo on
# bvm

# Deklarationer

App(tm) doc.tm;

Intr(tm, borneo) turing.borneo;

Mach(borneo) bvm;
--
# <Pres>
root turing.borneo with bvm as turing;

# <Body>
run(
  doc.tm with turing
)
```