De CXTEX distributie

Abstract
Het einddoel van het CXTEX project is om een hele texexec aanroep van begin tot eind te kunnen uitvoeren binnen één enkel, zo efficiënt mogelijk, systeemproces.
De eerste onderdelen van deze distributie worden in dit artikel gepresenteerd: nog op zichzelf staande, maar al wel naar C vertaalde versies van texexec, texutil en pdfetex.

Intro

In het dagelijks taalgebruik hebben we het over 'TEX draaien' of 'texexec runnen'. Dat wekt de indruk dat er sprake is van één enkele executable die wordt uitgevoerd om een PDF-bestand te genereren. De werkelijkheid is echter aanzienlijk complexer, omdat een bestand vaak herhaaldelijk gecompileerd moet worden. Dat kan zijn vanwege kruisverwijzingen, maar het kan ook voorkomen dat er externe programma's aangeroepen moeten worden voor bijvoorbeeld het aanmaken van indexen en de bibliografie.

Een concreet voorbeeld: ConTEXt maakt gebruik van twee verschillende hulpprogramma's:

Texutil

Tijdens het verwerken van een TEX-bestand bewaart ConTEXt allerlei informatie (onder andere voor de inhoudsopgave, lijsten van tabellen en figuren, registergegevens en kruisreferenties) in een bestand met als extensie `.tui`.

Dit bestand wordt na de TEX aanroep verwerkt door het programma texutil, om gebruikt te worden als invoer voor een eventuele volgende TEX aanroep.

Texexec

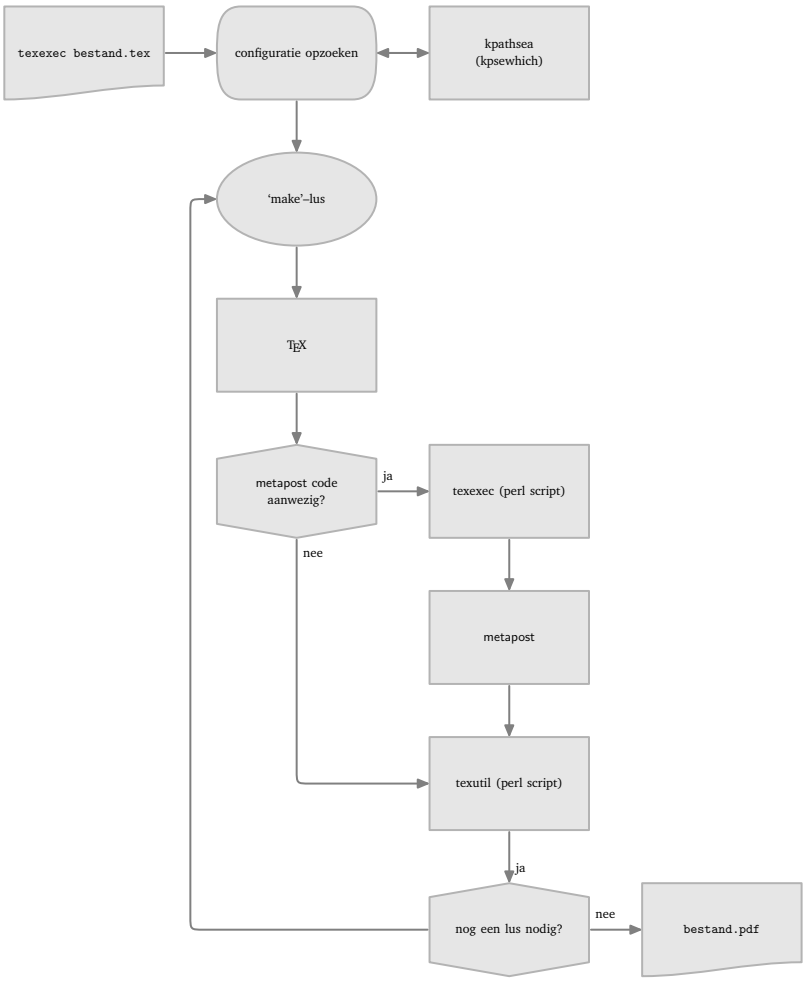
texutil en TEX zelf worden aangestuurd door het programma texexec. De belangrijkste taak van dit programma is om 'make' te emuleren. Dat wil zeggen: ervoor te zorgen dat TEX en texutil vaak genoeg worden aangeroepen om alle referenties op te lossen.

Bovendien verzorgt texexec het (eventueel) uitvoeren van metapost voor het aanmaken van gewenste plaatjes.

Het stroomdiagram van texexec is te zien boven aan de volgende bladzijde.

Omwille van het overzicht is ook nog het een en ander verwijderd uit het diagram:

- De twee stappen 'texexec (perl script)' en texutil (perl script)' voeren eenzelfde 'find configuration <=> kpathsea (kpsewhich)' actie uit als die je ziet weergegeven op de eerste regel van het diagram.
- En 'texutil' en 'texexec' bevatten zelfs nog een subprocedure om de nodige perl scripts *zelf* te vinden: de executables zijn namelijk kleine programma's die eerst `texutil.pl` dan wel `texexec.pl` opzoeken op de harde schijf, en daarna perl aanroepen.
- En dan zijn er nog diverse impliciete en expliciete programma-aanroepen in de scripts: commando's die uitgevoerd worden om bijvoorbeeld tijdelijke bestanden te maken, kopiëren of verwijderen.



Het gevolg van dit alles is dat tijdens het uitvoeren van het commando

```
$ texexec paper
```

er pakweg 60 verschillende processen worden opgestart voordat texexec klaar is met het aanmaken van het PDF-bestand voor het artikel dat je nu zit te lezen.

Het opstarten en afsluiten van al die processen (die grotendeels met elkaar communiceren via bestanden op de harde schijf) beslaat een flink gedeelte van de tijd die gemoeid is met het 'texexec runnen'.

Doelstelling

Ons bedrijf (Elvenkind) is een product aan het ontwikkelen (op basis van TEX uiteraard) dat XML Formatting Objects converteert naar PDF-bestanden. Omdat we dit programma willen en moeten leveren aan klanten die nog nooit van TEX, metapost, ConTEXt of perl gehoord hebben, willen we uitdrukkelijk een zo simpel mogelijke distributie. Ons doel is daarom dan ook:

- ☐ Eén enkel uitvoerbaar programma voor het produceren van de PDF.
- ☐ Met alle benodigde 'make' functionaliteit ingebakken, zodat een éénmalige aanroep volstaat.

- Configuratie door middel van een `.ini` bestand. Het voordeel van het `.ini` bestandsformaat is dat het veel makkelijker te begrijpen is dan de huidige (van de Unix shell afgeleide) syntax van `texmf.cnf`.
- We hebben graag een minimaal aantal instellingen in dit bestand. Instellingen die alleen voor experts begrijpelijk zijn willen we zoveel mogelijk verwijderen.
- We willen slechts een handjevol data-archieven verspreiden in plaats van een uitgebreide TDS boomstructuur met (tien)duizenden losse bestandjes.
- Het eindresultaat moet zo snel mogelijk zijn, voor gebruik in batch- en CGI-achtige omgevingen.

Uitvoering in stappen

Het belangrijkste eerste doel is het geschikt maken van de verschillende onderdelen van het toekomstige programma voor gebruik buiten de Web2C/TEX-live omgeving.

Losweken uit Web2c

Met name `kpathsea` is nauw verbonden met de Web2C compilatie-omgeving. Er waren wat trucs nodig om te zorgen dat de kpathsea-bibliotheek wilde compileren buiten de TEX-live boom. Het is gelukt, maar de huidige situatie is verre van elegant.

Daarnaast waren Siep Kroonenberg's patches voor de MinGW GNU C-Compiler nodig om een versie voor Microsoft Windows te kunnen compileren.

Conversie

Het aanmaken van één enkele executable is uiteraard een stuk eenvoudiger als de diverse programma's gebruik maken van dezelfde programmeertaal.

Gezien onze expertise en onze wensen was er feitelijk maar één optie beschikbaar: de programmeertaal C. Alle andere mogelijkheden zouden ons te veel moeite kosten om te leren óf zouden een veel te inefficiënt eindresultaat opleveren.

Een groot gedeelte van de gewenste 'make' en commando-regel functionaliteit was reeds beschikbaar in het perl script texexec, en we wisten al dat we ConTEXt als format zouden gaan gebruiken dus texutil was ook nodig. pdfetex was verreweg de beste TEX voor ons doel, en metapost zal worden gebruikt voor het aanmaken van plaatjes.

Omwille van het vereenvoudigen van ons productie-proces hebben we besloten eerst losstaande versies (in de programmeertaal C) te maken van de gewenste programma's. Een bonus daarvan is dat er een tussenresultaat beschikbaar is.

Texutil

Texutil wordt een vrij op zichzelf staand onderdeel van het uiteindelijke programma, en daarom is er vrij veel tijd besteed aan de conversie. De functionaliteit is identiek aan TeXUtil 8.2, maar ten opzichte van de perl implementatie zijn er een aantal flinke interne wijzigingen doorgevoerd.

- Er is een 'functie-bibliotheek' afgesplitst. Die is er vooral voor Win32/Unix unificatie (bewerkingen op bestanden en mappen, geheugen-allocatie).
- Er is langzamerhand ook een 'perl emulatie-bibliotheek' ontstaan. Het was voor een aantal perl functies vrij eenvoudig om generieke C code te schrijven die de feitelijke conversie simpeler maakte (zoals voor stringoperaties en het werken met hashes).
- Verwijdering van globale variabelen. Alle subroutines hebben als eerste argument een object dat een instance is van '`struct texutilstruct*`'. Die enkele struct bevat alle oorspronkelijke globale variabelen.
- Afsplitsing van een `main()`-functie die slechts zorg draagt voor de commandoregel-opties, zodat de hoofdmoot van het programma op triviale wijze beschikbaar is als een 'texutil-bibliotheek'.

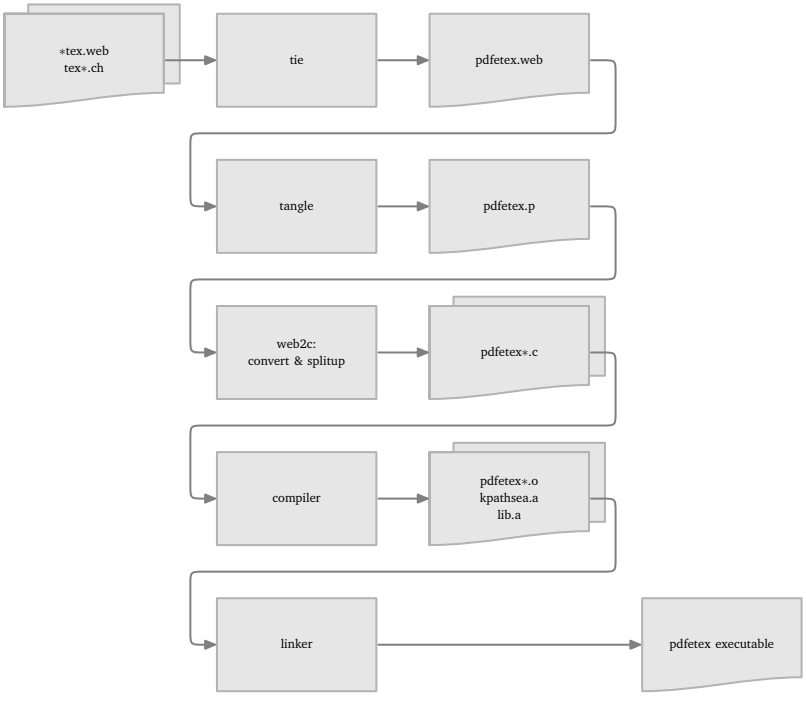
Texexec

Conversie van texexec naar C was redelijk eenvoudig. Omdat er tijdens deze conversie de geconverteerde versie van texutil al beschikbaar was, is er voor gekozen om rechtstreeks te linken tegen de 'texutil-bibliotheek', maar verder is het resultaat een getrouwe kopie van de functionaliteit van TeXExec 4.0.

Een interessant verschijnsel was dat juist de conversie van het help-systeem vrij ingewikkeld bleek. Dat is opmerkelijk omdat de help juist gebruik maakt van een perl module (Struct.pm) die uitdrukkelijk een C-constructie emuleert.

Pdfetex

Nu komen we bij de hoofdmoot van de werkzaamheden tot-nu-toe: de conversie van pdfetex. Hieronder zie je ter introductie het stroomdiagram van de compilatie van de Web2C-versie van TEX.



Voor ons doel was de handigste plaats om in te grijpen net na de 'tie' stap. Om uit te leggen waarom dit makkelijker is dan na de 'web2c' stap, moeten we een stukje voorbeeldcode laten zien. Als voorbeeld volgt nu de definitie van een van TEX's "Basic printing procedures".

De originele web programmacode voor de functie `print_cs`, die commando's en macro-namen toont bevat een paar web-commando's voor het aanmaken van de index (`@.XXX@>`) en de indentatie is niet echt fantastisch, maar verder is de pascal code redelijk makkelijk te volgen:

```
procedure print_cs(@!p:integer); {prints a purported control sequence}
begin if p<hash_base then {single character}
  if p>=single_base then
    if p=null_cs then
```

```
      begin print_esc("csname"); print_esc("endcsname");
      end
    else  begin print_esc(p-single_base);
      if cat_code(p-single_base)=letter then print_char(" ");
      end
  else if p<active_base then print_esc("IMPOSSIBLE.")
@.IMPOSSIBLE@>
  else print(p-active_base)
else if p>=undefined_control_sequence then print_esc("IMPOSSIBLE.")
else if (text(p)<0)or(text(p)>=str_ptr) then print_esc("NONEXISTENT.")
@.NONEXISTENT@>
else  begin print_esc(text(p));
  print_char(" ");
  end;
end;
```

De pascal code die wordt gegenereerd door tangle ziet er als volgt uit:

```
{:65}{262:}procedure printcs(p:integer);
begin if p<514 then if p>=257 then if p=513 then begin printesc(537);
printesc(538);end else begin printesc(p-257);
if eqtb[13636+p-257].hh.rh=11 then printchar(32);
end else if p<1 then printesc(539)else print(p-1)else if((p>=12526)and(p
<=16050))or(p>eqtbtop)then printesc(539)else if(hash[p].rh>=strptr)then
printesc(540)else begin printesc(hash[p].rh);printchar(32);end;end;
```

En dat is helemaal niet meer makkelijk te lezen, laat staan makkelijk te editen. Er zijn een aantal problemen met de leesbaarheid van deze pascal code.

1. Alle strings zijn vervangen door nummers (die nummers verwijzen naar het pool-bestand).
2. Alle 'symbolische constanten' (zoals `hash_base`) zijn vervangen door hun numerieke waarde.
3. Tangle's eindresultaat bevat net genoeg witruimte om de code eenduidig te maken, niets extra's.
4. De underscores uit het originele programmafragment zijn allemaal verdwenen.

Gelukkig voegt het Web2C programma convert weer wat witruimte toe, maar dat is niet meer goed genoeg om de situatie te redden. Het eindresultaat in C code van de 'web2c' stap ziet er zo uit:

```
void
#ifdef HAVE_PROTOTYPES
zprintcs ( integer p )
#else
zprintcs ( p )
  integer p ;
#endif
{
  printcs_regmem
  if ( p < 514 )
  if ( p >= 257 )
  if ( p == 513 )
  {
    printesc ( 537 ) ;
    printesc ( 538 ) ;
  }
```

```
  else {
    printesc ( p - 257 ) ;
    if ( eqtb [13636 + p - 257 ].hh .v.RH == 11 )
    printchar ( 32 ) ;
  }
  else if ( p < 1 )
  printesc ( 539 ) ;
  else print ( p - 1 ) ;
  else if ( ( ( p >= 12526 ) && ( p <= 16050 ) ) - ( p > eqtbtop ) )
  printesc ( 539 ) ;
  else if ( ( hash [p ].v.RH >= strptr ) )
  printesc ( 540 ) ;
  else {
    printesc ( hash [p ].v.RH ) ;
    printchar ( 32 ) ;
  }
}
```

Vanwege de slechte leesbaarheid van de door tangle/web2c gegenereerde code is er voor gekozen om de hele conversie van Web code naar C-code handmatig te doen. Dat was een heel karwei, maar het eindresultaat ziet er nu zo uit:

```
void print_cs (int p) { /* prints a purported control sequence */
  if (p < hash_base) { /* single character */
    if (p >= single_base) {
      if (p == null_cs) {
        print_esc_string ("csname");
        print_esc_string ("endcsname");
      } else {
        print_esc (p - single_base);
        if (cat_code (p - single_base) == letter)
          print_char (' ');
      }
    } else {
      if (p < active_base) { print_esc_string ("IMPOSSIBLE."); }
      else                 { zprint (p - active_base);         }
    }
  } else {
    if ((p >= undefined_control_sequence) && (p <= eqtb_size)) {
      print_esc_string ("IMPOSSIBLE.");
    } else {
      if (text (p) >= str_ptr){
        print_esc_string ("NONEXISTENT.");
      } else {
        print_esc (text (p)); print_char (' ');
      }
    }
  }
}
```

En dat vond ik zelf er een stuk prettiger uitzien. De 'nieuwe' procedures die op `_string` eindigen zijn ontstaan omdat dankzij die functies het `pool` bestand opgeheven kon worden. Een extra voordeel van de conversie 'met de hand' is dat de Web-commentaren zijn blijven bestaan.

De huidige C code bestaat uit ongeveer 50.000 regels geconverteerde C en commentaren, in zo'n 70 bronbestanden met elk een eigen headerbestand. De pdfetex

Web-code waar we mee begonnen zijn is gebaseerd op pdfetex versie 1.11b, die gebruik maakte van web2c 7.5.2.

Het grootste deel van de extra functionaliteit die werd aangeboden door het Web2C-systeem is niet beschikbaar in de CXTEX-versie. Een paar onderdelen daarvan zijn heel doelbewust niet geconverteerd, zoals de volgende opties:

- `-translate-file`
  CXTEX gebruikt momenteel altijd 8-bit invoer. Uiteindelijk zal encTEX beschikbaar komen ter vervanging van deze optie.
- `-src-specials`
  Het gebruik van source specials is specifiek verbonden met DVI-uitvoer. De CXTEX zal echter geen DVI previewer bevatten waardoor deze optie zinloos wordt.
- `-ipc`
  Deze optie was alleen beschikbaar als hij expliciet was aangezet tijdens het compileren van de Web2c-versie, en werkte slechts in combinatie met DVI-uitvoer.
- `[-no]-mktex=FMT`
  De mktexXX-scripts worden nooit aangeroepen door CXTEX, dit is alvast in voorbereiding op het verdwijnen van de kpathsea-bibliotheek.

En een paar andere Web2C-opties zijn gewoonweg niet vertaald, maar zonder een uitdrukkelijke reden anders dan gebrek aan (werk)tijd. De onderstaande opties komen daarom wellicht later weer terug:

`-parse-first-line`
`-file-line-error-style`
`-recorder`
`-shell-escape` (staat momenteel altijd aan)
`-output-comment`

Er zijn nog wat andere verschillen met Web2C:

- De hash is bevroren op $(2^{20} - 2^{12} = 1.044.480)$ mogelijke commando's.
- Er wordt dynamische her-allocatie gebruikt voor een aantal van TEX's interne structuren. Op dit moment zijn dat:
  - `buf_size`
  - `nest_size`
  - `save_size`
  - `pool_size`
  - `max_strings`

  en vrijwel alle andere structuren zullen nog gedaan worden.
- Natuurlijk is de banner aangepast (Een CXTEX versienummer is toegevoegd aan de toch al lange rij versienummers).
- De TFM-lees-code is geoptimaliseerd.
- Er is geen pool file.
- Format files zijn niet uitwisselbaar met de Web2c-versies. Om toch toe te staan dat beide executables bestaan binnen één `texmf`-bestandsstructuur, hebben de CXTEX formats als extensie `efm`. Dit is een tijdelijke oplossing, het probleem zal uiteindelijk vanzelf verdwijnen dankzij een toekomstige aanpassing in de TEX Directory Structuur (TDS).

Voor de toekomst

- □ Conversie van metapost naar C en het geschikt maken van de metapost broncode om gebruikt te kunnen worden als bibliotheek-subroutine. Werk hieraan is inmiddels begonnen.
- □ Vervangen van de huidige kpathsea door een privé-bibliotheek of door de nieuwe generatie kpathsea waar Olaf Weber momenteel aan werkt.
- □ Generalisatie van de commando-regel.
- □ Creëren van de feitelijke CXTEX-executable.
- □ Maken van een grafische font installatie- en configuratietool

Wie wat waar?

De Homepage is http://tex.aanhet.net/cxtex.

Er staan op de website archieven van de bronbestanden en de Win32 executables van de huidige versie (0.5.1). Zelf compileren onder diverse unices hoort vlekkeloos te verlopen, de enige vereiste is een relatief nieuwe versie van gcc en g++ (versie 2.96 of hoger). Aan ondersteuning voor MacOS X wordt gewerkt.

Taco Hoekwater