Text normalization rules for Swedish TTS

Erik Sterneberg

February 16, 2011

Contents

1 Introduction

The text normalization rules in `tn_rules.xml` can be used to expand non-alpha-strings such as numbers and dates to words, but is also very useful for expanding number expressions containing abbreviations such as *5 km, 68 kg, 200 lb* and *3,5 dl*. The files `tn_rules.xml`, `tn_rules_regex.txt` `Preprocess.java` `TNParser.java` and `TNNormalize.java` together function as a finite state transducer, taking string input and generating string output. Starting with the rule named `start`, different rules are triggered handling different parts of the input. Child nodes (`<ref name="..."/>`) whose names are written in upper case have regular expressions pertaining to them in the file `tn_rules_regex`. The first rule matching the input string will be added to the action stack – the non-matching rules and the rest of the untried rules will be discarded. Child nodes that have names written in lower case have already been matched to the input string by their parent and will be added to the action stack without trial. When a rule containing child nodes called `in` and `out` appear on top of the stack, the program will look for the contents of the `in`-node in the beginning of the inputstring, provided that the regular expression was matched in case of the rule being upper case. As an example, see what happens with the input string `998,40`:

1. The child rule `CARDINAL` to the rule `start` will be matched and put on top of stack.
2. Rule `CARDINAL` gives no output; redirects to `CARDINAL_INTEGER_DECIMAL`.
3. Rule `CARDINAL_INTEGER_DECIMAL` gives no output; puts the rules `cardinal_integer, comma`, and `digit_by_digit` on top of stack.
4. The child rule `CARDINAL_100_999` (matching the range 100 to 999) to the rule `cardinal_integer` will be matched and put on top of stack.
5. Rule `CARDINAL_100_999` gives no output; puts the rules `cardinal_single_dig_neutr, hundra` and `cardinal_00_99` on top of the stack.
6. The child rule `CARDINAL_2_9` to the rule `cardinal_single_dig_neutr` will be matched and put on top of the stack.
7. The child rule `NINE` to the rule `CARDINAL_2_9` will be matched and put on top of the stack.
8. The rule `NINE` takes a `9` as input (removing it from the inputstring/buffer), and gives `nio` as output.
9. The rule `hundra` takes no input and gives `hundra` as output.
10. The child rule `CARDINAL_10_99` to the rule `cardinal_00_99` will be matched and put on top of the stack.
11. The child rule `CARDINAL_20_99` to the rule `CARDINAL_10_99` will be matched and put on top of the stack.
12. The child rule `CARDINAL_NINETIES` to the rule `CARDINAL_20_99` will be matched and put on top of the stack.
13. Rule `CARDINAL_NINETIES` gives no output; puts the rules `ninety` and `cardinal_single_dig_silent_zero` on top of the stack.
14. The rule `ninety` takes a `9` from the inputbuffer and gives the output `nittio`.
15. The child rule `EIGHT` to the rule `cardinal_single_dig_silent_zero` will be matched and put on top of the stack.
16. The rule `EIGHT` takes `8` as input and gives `åtta` as output.
17. The rule `comma` takes no input (spaces, commas and dots are automatically removed) and gives `komma` as output.
18. The rule `digit_by_digit` gives no output; puts the rules `cardinal_single_dig` and `digit_by_digit` on top of the stack.

19. The child rule `CARDINAL_2_9` to the rule `cardinal_single_dig` will be matched and put on top of the stack.

20. The child rule `FOUR` to the rule `CARDINAL_2_9` will be matched and put on top of the stack.

21. The rule `FOUR` will take `4` as input, giving `fyra` as output.

22. The remaining zero in the inputbuffer will be processed in the same was as the preceeding digit, ultimately leaving the inputbuffer empty and the action stack with one rule. Whenever the action stack or the inputbuffer is empty, the processing will halt.

Currently, only tn-rules for cardinal numbers have been written and tested properly. When adding new categories, start by listing cases where the rules should be triggered.

2 Numbers

2.1 Cardinals

Cases:

Example	Expansion
1 234	ett tusen två hundra trettiofyra
1234	ett tusen två hundra trettiofyra
1.234	ett tusen två hundra trettiofyra
1 234,56	ett tusen två hundra trettiofyra komma femtiosex
1234,56	ett tusen två hundra trettiofyra komma femtiosex
123456789	ett hundra tjugotre miljoner fyra hundra femtiosex tusen sju hundra åttionio
1.234.567,8	en miljon två hundra trettiofyra tusen fem hundra sextiosju komma åtta
1234567,8	en miljon två hundra trettiofyra tusen fem hundra sextiosju komma åtta
1 234 567,8	en miljon två hundra trettiofyra tusen fem hundra sextiosju komma åtta
1 miljon	en miljon
1 tusen	ett tusen

Extra information:

- Cardinals must not begin with a zero

2.2 Future work

2.2.1 Intervals

Cases:

Example	Expansion
0-70	noll till sjuttio
20-24	tjugo till tjugofyra

2.2.2 Ordinals

Cases:

Example	Expansion
15:e	femtonde
6:e	sjätte
5e	femte
1:a	första
1:e	förste
100:e	hundrade
1 000:e	tusende

2.2.3 Currency

2.2.4 Phone numbers

3 Dates & time

3.1 Future work

3.1.1 Dates

3.1.2 Time

4 Digit by digit

These are the 'fallback'-rules which are to be used if no other rules have been triggered.
Cases:

Example	Expansion
09876	noll nio åtta sju sex