Formal Specification and Documention using Z:
A Case Study Approach

A Text Formatting Tool
(Case Study)

Jonathan Bowen
Department of Computer Science
University of Reading

Email: *J.P.Bowen@reading.ac.uk*

URL: *http://www.comlab.ox.ac.uk/oucl/users/jonathan.bowen/zbook.html*

Basic Concepts

UNIX text files consist of ASCII characters.

$$[\ CHAR\ ]$$

One of these characters is a blank space.

$$space : CHAR$$

Characters are organized as lines in a text file. A complete text file, or document, consists of a number of lines of characters.

$$LINE == \mathrm{seq}\, CHAR$$
$$DOC == \mathrm{seq}\, LINE$$

Repeating a sequence

$$[X]$$

$$\_\ \underline{rep}\ \_ : (\mathrm{seq}\, X) \times \mathrm{fl} \rightarrow \mathrm{seq}\, X$$

$$\forall\, s : \mathrm{seq}\, X;\ n : \mathrm{fl}_1 \bullet$$
$$s\ \underline{rep}\ 0 = \langle\rangle \wedge$$
$$s\ \underline{rep}\ n = s\ \emptyset\ (s\ \underline{rep}\ (n-1))$$

Some laws about *rep*:

$$\begin{array}{rcl}
s : \mathrm{seq}\, X & \vdash & s\ \underline{rep}\ 0 = \langle\rangle \\
s : \mathrm{seq}\, X & \vdash & s\ \underline{rep}\ 1 = s \\
n : \mathrm{fl} & \vdash & \langle\rangle\ \underline{rep}\ n = \langle\rangle \\
s : \mathrm{seq}\, X;\ n : \mathrm{fl} & \vdash & \#(s\ \underline{rep}\ n) = \#s * n \\
s : \mathrm{seq}\, X;\ n : \mathrm{fl}_1 & \vdash & \mathrm{ran}(s\ \underline{rep}\ n) = \mathrm{ran}\, s \\
s : \mathrm{seq}\, X;\ n : \mathrm{fl} & \vdash & \forall\, m : \mathrm{fl} \mid m < n \bullet \\
 & & \quad (s\ \underline{rep}\ n)\ \underline{after}\ (\#s * m)\ \underline{for}\ n = s
\end{array}$$

Processing the Input

Removing leading spaces from a line:

$$
\begin{array}{|l}
clipleft : LINE \rightarrow LINE \\
\hline
clipleft\langle\rangle = \langle\rangle \\
\forall\, l : LINE \mid l \neq \langle\rangle \bullet \\
\qquad head\ l \neq space \Rightarrow clipleft\ l = l \wedge \\
\qquad head\ l = space \Rightarrow clipleft\ l = clipleft\ (tail\ l)
\end{array}
$$

Removing trailing spaces:

$$
\begin{array}{|l}
clipright : LINE \rightarrow LINE \\
\hline
clipright = rev \mathbin{\emptyset} clipleft \mathbin{\emptyset} rev
\end{array}
$$

Removing leading and trailing spaces:

$$
\begin{array}{|l}
clip : LINE \rightarrow LINE \\
\hline
clip = clipleft \mathbin{\emptyset} clipright
\end{array}
$$

Spaces may be removed in either order:

$$\vdash\ clipleft \mathbin{\emptyset} clipright = clipright \mathbin{\emptyset} clipleft$$

Indenting to a given column position:

$$
\begin{array}{|l}
left_0 : \mathrm{fl} \rightarrow LINE \rightarrow LINE \\
\hline
\forall\, n : \mathrm{fl};\ l : LINE \bullet \\
\qquad left_0\ n\ l = (\langle space\rangle\ \underline{rep}\ n) \mathbin{\emptyset} l
\end{array}
$$

Centering a line:

$$
\begin{array}{|l}
centre_0 : \mathrm{fl} \rightarrow LINE \rightarrow LINE \\
\hline
\forall\, n : \mathrm{fl};\ l : LINE \bullet \\
\qquad \#l \leq 2 * n \Rightarrow centre_0\ n\ l = (\langle space\rangle\ \underline{rep}\ (n - (\#l \operatorname{div} 2))) \mathbin{\emptyset} l \wedge \\
\qquad \#l > 2 * n \Rightarrow centre_0\ n\ l = l
\end{array}
$$

Right-alignment of a line:

$$
\begin{array}{|l}
right_0 : \mathrm{fl} \rightarrow LINE \rightarrow LINE \\
\hline
\forall\, n : \mathrm{fl};\ l : LINE \bullet \\
\qquad \#l \leq n \Rightarrow right_0\ n\ l = (\langle space\rangle\ \underline{rep}\ (n - \#l)) \mathbin{\emptyset} l \wedge \\
\qquad \#l > n \Rightarrow right_0\ n\ l = l
\end{array}
$$

Clipping lines first:

$$
\begin{array}{|l}
left, \\
centre, \\
right : \mathrm{fl} \rightarrow LINE \rightarrow LINE \\
\hline
\forall\, n : \mathrm{fl};\ l : LINE \bullet \\
\quad left\ n\ l = left_0\ n\ (clip\ l) \wedge \\
\quad centre\ n\ l = centre_0\ n\ (clip\ l) \wedge \\
\quad right\ n\ l = right_0\ n\ (clip\ l)
\end{array}
$$

Documents

$$
\begin{array}{|l}
\hline
TEXT \\
\hline
text : DOC \\
\hline
\end{array}
$$

Changes in this document:

$$\Delta TEXT \ \widehat{=}\ TEXT \wedge TEXT'$$

Change selected using an option:

$$Option \ ::= \ LeftOption \mid CentreOption \mid RightOption$$

$$
\begin{array}{|l}
\hline
POS_0 \\
\hline
\Delta TEXT \\
option? : Option \\
column? : \mathrm{fl} \\
\hline
option? = LeftOption \Rightarrow text' = text \oplus (left\ \ column?) \\
option? = CentreOption \Rightarrow text' = text \oplus (centre\ \ column?) \\
option? = RightOption \Rightarrow text' = text \oplus (right\ \ column?) \\
\hline
\end{array}
$$

Implementation Details

Previous assumption: all printable characters of the same width.

Tabs

Horizontal *tab*:

$$
\begin{array}{|l}
tab : CHAR \\
\hline
tab \neq space
\end{array}
$$

'Tabulate' to the next tab column position.

$$
\begin{array}{|l}
tabsize : \mathrm{fl} \\
\hline
tabsize > 1
\end{array}
$$

Introduces additional complication into the specification.

Split a sequence into a series of segments.

$$
\begin{array}{|l}
\hline
[X] \\
\hline
\_\ \underline{split}\ \_ : (\mathrm{seq}\, X) \times \mathrm{fl}_1 \rightarrow \mathrm{seq}(\mathrm{seq}\, X) \\
\hline
\forall\, s : \mathrm{seq}_1\, X;\ n : \mathrm{fl}_1;\ ss : \mathrm{seq}_1(\mathrm{seq}_1\, X) \bullet \\
\qquad \langle\rangle\ \underline{split}\ n = \langle\rangle \wedge \\
\qquad s\ \underline{split}\ n = ss \Leftrightarrow \\
\qquad\qquad \emptyset / ss = s \wedge \\
\qquad\qquad (\forall\, t : \mathrm{seq}\, X \mid t \in \mathrm{ran}(front\ ss) \bullet \#t = n) \wedge \\
\qquad\qquad 0 < \#(last\ ss) \leq n \\
\hline
\end{array}
$$

Convert trailing spaces to a tab.

$$
\begin{array}{|l}
addtab : LINE \ \tilde{}\ LINE \\
\hline
\forall\, l : LINE \mid tab \notin \mathrm{ran}\, l \bullet \\
\qquad (\#l = tabsize \wedge \#(clipright\ l) < \#l - 1) \Rightarrow \\
\qquad\qquad addtab\ l = (clipright\ l) \emptyset \langle tab \rangle \wedge \\
\qquad (\#l \neq tabsize \vee \#(clipright\ l) \geq \#l - 1) \Rightarrow \\
\qquad\qquad addtab\ l = l
\end{array}
$$

Convert spaces in a line to tabs where appropriate:

$$
\begin{array}{|l}
unexpand : LINE \ \tilde{}\ LINE \\
\hline
\forall\, l : LINE \mid tab \notin \mathrm{ran}\, l \bullet \\
\qquad unexpand\ l = \emptyset / ((l\ \underline{split}\ tabsize) \oslash addtab)
\end{array}
$$

Only use on lines not containing tabs.

Can only reduce the length of the line:

$$\vdash\ \forall\, l : LINE \mid tab \notin \mathrm{ran}\, l \bullet \#(unexpand\ l) \leq \#l$$

Cut a sequence into segments:

$[X]$

$$\_ \underline{cut} \_ : (\mathrm{seq}\, X) \times (\mathrm{seq}_1\, X) \mathbin{\tilde{}} \mathrm{seq}(\mathrm{seq}\, X)$$

$$\begin{aligned}
&\forall\, p : \mathrm{seq}_1\, X;\ s, t : \mathrm{seq}\, X \mid \neg\, (p \subseteq \mathit{front}(s \mathbin{\emptyset} p)) \bullet \\
&\qquad \langle\rangle \mathbin{\underline{cut}} p = \langle\rangle \wedge \\
&\qquad s \neq \langle\rangle \Rightarrow s \mathbin{\underline{cut}} p = \langle s \rangle \wedge \\
&\qquad (s \mathbin{\emptyset} p \mathbin{\emptyset} t) \mathbin{\underline{cut}} p = \langle s \mathbin{\emptyset} p \rangle \mathbin{\emptyset} (t \mathbin{\underline{cut}} p)
\end{aligned}$$

Recombining the segments gives the original sequence:

$$a, b : \mathrm{seq}\, X \ \vdash\ \emptyset/(a \mathbin{\underline{cut}} b) = a$$

Trailing spaces can be substituted with a tab.

$$\mathit{addspace} : \mathit{LINE} \rightarrow \mathit{LINE}$$

$$\begin{aligned}
&\mathit{addspace}\langle\rangle = \langle\rangle \\
&\forall\, l : \mathit{LINE} \mid l \neq \langle\rangle \bullet \\
&\qquad \mathit{last}\ l = \mathit{tab} \Rightarrow \mathit{addspace}\ l = \\
&\qquad\qquad (\mathit{front}\ l) \mathbin{\emptyset} (\langle \mathit{space} \rangle \mathbin{\underline{rep}} (\mathit{tabsize} + 1 - (\#l \bmod \mathit{tabsize}))) \wedge \\
&\qquad \mathit{last}\ l \neq \mathit{tab} \Rightarrow \mathit{addspace}\ l = l
\end{aligned}$$

Convert tabs to spaces:

$$\mathit{expand} : \mathit{LINE} \rightarrow \mathit{LINE}$$

$$\begin{aligned}
&\forall\, l : \mathit{LINE} \bullet \\
&\qquad \mathit{expand}\ l = \emptyset/((l \mathbin{\underline{cut}} \langle \mathit{tab} \rangle) \mathbin{\emptyset} \mathit{addspace})
\end{aligned}$$

Can only increase the length:

$$l : \mathit{LINE} \ \vdash\ \#(\mathit{expand}\ l) \geq \#l$$

Converting spaces to tabs and back gives the same line:

$$\vdash\ \forall\, l : \mathit{LINE} \mid \mathit{tab} \notin \mathrm{ran}\, l \bullet \mathit{expand}\ (\mathit{unexpand}\ l) = l$$

But converting tabs to spaces and back may not:

$$\vdash\ \exists\, l : \mathit{LINE} \bullet \mathit{unexpand}\ (\mathit{expand}\ l) \neq l$$

Column width of a line:

$$\mathit{width} : \mathit{LINE} \rightarrow \mathrm{fl}$$

$$\begin{aligned}
&\forall\, l : \mathit{LINE} \bullet \\
&\qquad \mathit{width}\ l = \#(\mathit{expand}\ l)
\end{aligned}$$

Adding tabs does not affect the width of a line:

$$l : \mathit{LINE} \ \vdash\ \mathit{width}\ (\mathit{unexpand}\ l) = \mathit{width}\ l$$

Lines

A non-empty unique *newline* character sequence may be used to terminate lines:

$$
\begin{array}{|l}
nl : \mathrm{seq}_1\ CHAR \\
\hline
\mathrm{disjoint}\ \langle \{space, tab\}, \mathrm{ran}\ nl \rangle
\end{array}
$$

Under UNIX this sequence is the ASCII *line-feed* 'control' character.

The characters should not occur *within* any line.

Combine a sequence of segments using another sequence as a terminator:

$$
\begin{array}{|l}
\hline
[X] \\
\hline
\_\ \underline{comb}\ \_ : (\mathrm{seq}(\mathrm{seq}\ X)) \times (\mathrm{seq}\ X) \rightarrow \mathrm{seq}\ X \\
\hline
\forall\ s, p : \mathrm{seq}\ X;\ ss : \mathrm{seq}(\mathrm{seq}\ X) \bullet \\
\qquad \langle\rangle\ \underline{comb}\ p = \langle\rangle\ \wedge \\
\qquad \langle s \rangle\ \underline{comb}\ p = s \mathbin{\emptyset} p\ \wedge \\
\qquad (\langle s \rangle \mathbin{\emptyset} ss)\ \underline{comb}\ p = s \mathbin{\emptyset} p \mathbin{\emptyset} (ss\ \underline{comb}\ p) \\
\hline
\end{array}
$$

With a *terminator* rather than a *separator*, an empty document and a single empty line can be distinguished:

$$p : \mathrm{seq}\ X\ \vdash\ \langle\rangle\ \underline{comb}\ p = \langle\rangle$$

$$p : \mathrm{seq}\ X\ \vdash\ \langle\langle\rangle\rangle\ \underline{comb}\ p = p$$

Separate a sequence into a series of segments using a non-empty pattern.

$$
\begin{array}{|l}
\hline
[X] \\
\hline
\_\ \underline{sep}\ \_ : (\mathrm{seq}\ X) \times (\mathrm{seq}_1\ X) \rightarrow \mathrm{seq}(\mathrm{seq}\ X) \\
\hline
\forall\ s : \mathrm{seq}\ X;\ p : \mathrm{seq}_1\ X;\ ss : \mathrm{seq}(\mathrm{seq}\ X) \bullet \\
\qquad s\ \underline{sep}\ p = ss \Leftrightarrow \\
\qquad\qquad ss\ \underline{comb}\ p = s\ \wedge \\
\qquad\qquad (\ \forall\ t : \mathrm{seq}\ X \mid t \in \mathrm{ran}\ ss \bullet \neg\ (p \subseteq t)\ ) \\
\hline
\end{array}
$$

N.B.: the sequence to be separated must be terminated with the pattern or be empty to be valid.

Files

A UNIX file is implemented as a sequence of characters:

$$FILE == \mathrm{seq}\, CHAR$$

On input, several files can be combined with tabs converted to spaces:

$$\begin{array}{|l}
\hline
\textit{PrePOS}_0 \\
TEXT \\
input? : \mathrm{seq}\, FILE \\
\hline
text = ((\emptyset / input?)\ \underline{sep}\ nl) \mathbin{;} expand \\
\hline
\end{array}$$

Output is a single document.

Optionally, spaces may be be converted to tabs where possible.

$$Blanks ::= Yes \mid No$$

$$\begin{array}{|l}
\hline
\textit{PostPOS}_0 \\
blanks? : Blanks \\
output! : FILE \\
TEXT' \\
\hline
blanks? = Yes \Rightarrow output! = text'\ \underline{comb}\ nl \\
blanks? = No \Rightarrow output! = (text' \mathbin{;} unexpand)\ \underline{comb}\ nl \\
\hline
\end{array}$$

Combined new specification for the behaviour of the tool:

$$POS \mathrel{\widehat{=}} PrePOS_0 \wedge POS_0 \wedge PostPOS_0$$