

1 Notation

The grammar for the Lando System Specification Sublanguage is written in the EBNF notation. The main elements of the notation that we utilize are:

- Terminals are represented with double or single quotes; e.g. "explanation".
- Optional bits are represented with squared brackets; e.g. ["explanation" paragraph].
- Repetition is represented with curly braces; e.g. {identifier}
- We use a slightly enhanced notation - $\{a\}^+$ to indicate *non-zero* repetitions. This is simply equivalent to: $a\{a\}$.
- For defining terminals, we would like to use the EBNF special form to declare an extended regular expression - for example: $?\wedge+/?$. However since this is rather verbose, we will simply use $\wedge+/$ for convenience.
- Blocks in the language are typically delimited by keywords indicating the start of *another* block. In defining the grammar this translates to lookaheads: (i.e. peeking at incoming tokens without consuming them). We use the perl regular expression format to indicate this. E.g. $/\?= (eol "system")/$.
- Many constructs in the grammar include significant whitespaces (e.g. name-phrase). However note that we only consider *internal* whitespaces to be significant. For example, a index entry "Email Address : foo <foo@example.com> " is interpreted as the mapping - "Email Address" \mapsto "foo <foo@example.com>" and not "Email Address " \mapsto "foo <foo@example.com> "

2 Grammar

lando-source	::=	{ spec-element }	<i>Lando source</i>
spec-element	::=	system subsystem component event scenario requirement relation	<i>Specification Elements</i>
system	::=	\mathcal{LC} "system" name-phrase-rel [rel-keyword name-phrase] \mathcal{C} eol explanation eol [eol "indexing" indexing] { eol subsystem } $/(\?= nl\text{-}sys\text{-}keyword eof)/$ block-end	<i>System</i>
subsystem	::=	\mathcal{LC} "subsystem" name-phrase-rel [rel-keyword name-phrase] \mathcal{C} eol explanation eol [eol "indexing" indexing] { eol component } $/(\?= nl\text{-}sysys\text{-}keyword eof)/$ block-end	<i>Subsystem</i>

component	::=	\mathcal{LC} "component" name-phrase-rel [rel-keyword name-phrase] \mathcal{C} { eol component-part } /(?= nl-keyword eof) / block-end	Class
component-part	::=	constraint command query	Component Parts
constraint	::=	/[^\?!\+?\ \.]/m	Constraint
query	::=	/[^\?!\+?\ \?]/m	Query
command	::=	/[^\?!\+?\ !]/m	Command
event	::=	\mathcal{LC} "events" name-phrase \mathcal{C} { eol event-entry } /(?= nl-keyword eof) / block-end	Events
event-entry	::=	name-phrase eol sentence	Event Entry
scenario	::=	\mathcal{LC} "scenarios" name-phrase \mathcal{C} { eol scenario-entry } /(?= nl-keyword eof) / block-end	Scenario
scenario-entry	::=	name-phrase eol sentence	Scenario Entry
requirement	::=	\mathcal{LC} "requirements" name-phrase \mathcal{C} { eol req-entry } /(?= nl-keyword eof) / block-end	Requirements
req-entry	::=	name-phrase eol sentence	Requirements Entry
relation	::=	\mathcal{LC} "relation" name-phrase-rel rel-keyword name-phrase \mathcal{C} / (?= nl-keyword eof) / block-end	Relation Declaration
indexing	::=	{ eol index-entry }	Index List
index-entry	::=	index-key ':' index-val-list	Index List
index-key	::=	/[^\:]\+ (?= :)/	Index Key
index-val-list	::=	index-val { eol index-val } * / (?= eof nl-keyword eol index-key) /	Index Value List
index-val	::=	/[^\:]\+ /	Index Value
name-phrase-rel	::=	/\w.* (?= rel-keyword eol) /	Name-Phrase
sentence	::=	/[^\?!\+?\ \.?!]/m	Sentence
paragraph	::=	sentence ⁺ / (?= (eol keyword eof)) /	Paragraph
explanation	::=	paragraph	Explanation
keyword	::=	"system" "subsystem" "component" "events" "scenarios" "requirements" "relation"	All Keywords
nl-keyword	::=	eol keyword	Keyword on new line
nl-sys-keyword	::=	eol "system"	
nl-subsys-keyword	::=	eol ("subsystem" "system")	
rel-keyword	::=	"inherit" "client" "contains"	Relation keywords
block-end	::=	eol eof	Block End
eol	::=	/(\r?\n) \r/	New Line
\mathcal{LC}	::=	{ \mathcal{C} eol }	Opt Line Comments
\mathcal{C}	::=	["//" /. * (?= (eol eof)) /]	Opt Comment

