Interpreting Pumping Lemma for Regular languages

Here is one way to interpret the Pumping Lemma for regular languages. Credit is due to Shai Halevi, who taught this in a recitation I was in at MIT in 1995.

Pumping Lemma

If L is regular	$\Longrightarrow$	then it has a DFA D recognizing it
then there exists $p \geq 0$	$\Longrightarrow$	where p is the number of states of D
so that for all $s \in L$	$\Longrightarrow$	which defines a path in D
with $\lvert s\rvert \geq p$	$\Longrightarrow$	so this path contains a cycle
there is a parse of $s = xyz$	$\Longrightarrow$	x is everything *before* the first cycle
with $\lvert y\rvert \geq 1$ and $\lvert xy\rvert \leq p$		y is everything *on* the first cycle
		z is everything *after* the first cycle
such that for any $i \geq 0$	$\Longrightarrow$	no matter how many times we repeat the cycle
$xy^iz \in L$	$\Longrightarrow$	we end up in a final state of D

Showing that L is not regular

For any integer $p \geq 0$	$\Longrightarrow$	show that L is not recognizable by any DFA with p states
		suppose toward a contradiction that there was a DFA D
		having p states and recognizing L
there exists a string $s \in L$ such that $\lvert s\rvert \geq p$	$\Longrightarrow$	so in processing s we must visit more than p states
	$\Longrightarrow$	so it defines a path with a cycle
so that for any parse $s = xyz$	$\Longrightarrow$	no matter where this cycle is
with $\lvert y\rvert \geq 1$ and $\lvert xy\rvert \leq p$	$\Longrightarrow$	
there exists some $i \geq 0$	$\Longrightarrow$	we can repeat the cycle i times
so that $xy^iz \notin L$	$\Longrightarrow$	and end up in a non-final state of D