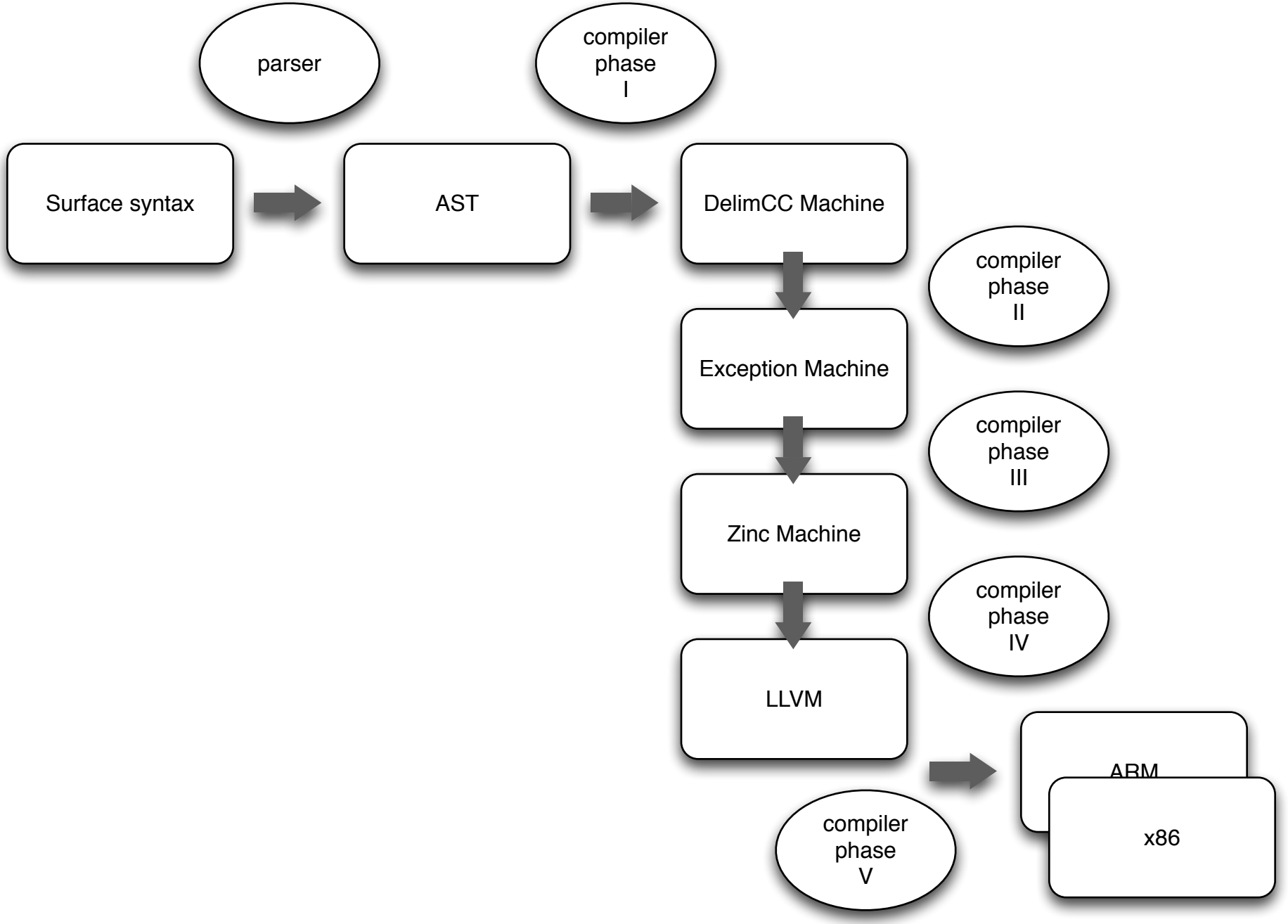
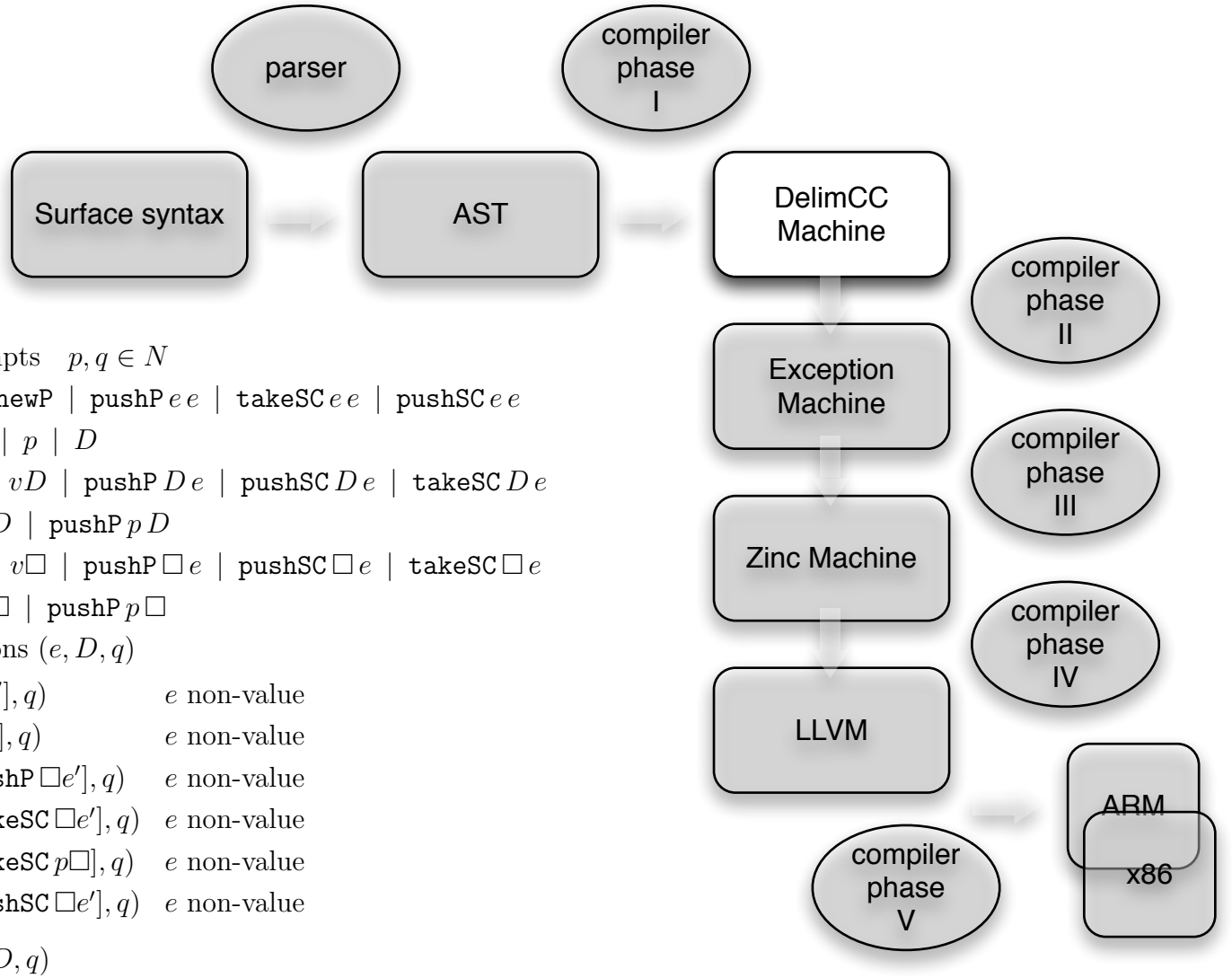

parser
compiler phase I
Surface syntax
AST
DelimCC Machine
compiler phase II
Exception Machine
compiler phase III
Zinc Machine
compiler phase IV
LLVM
compiler phase V
ARM
x86



Variables $x, y, \ldots$ Prompts $p, q \in N$

Expressions $e ::= v \mid e\,e \mid \texttt{newP} \mid \texttt{pushP}\,e\,e \mid \texttt{takeSC}\,e\,e \mid \texttt{pushSC}\,e\,e$

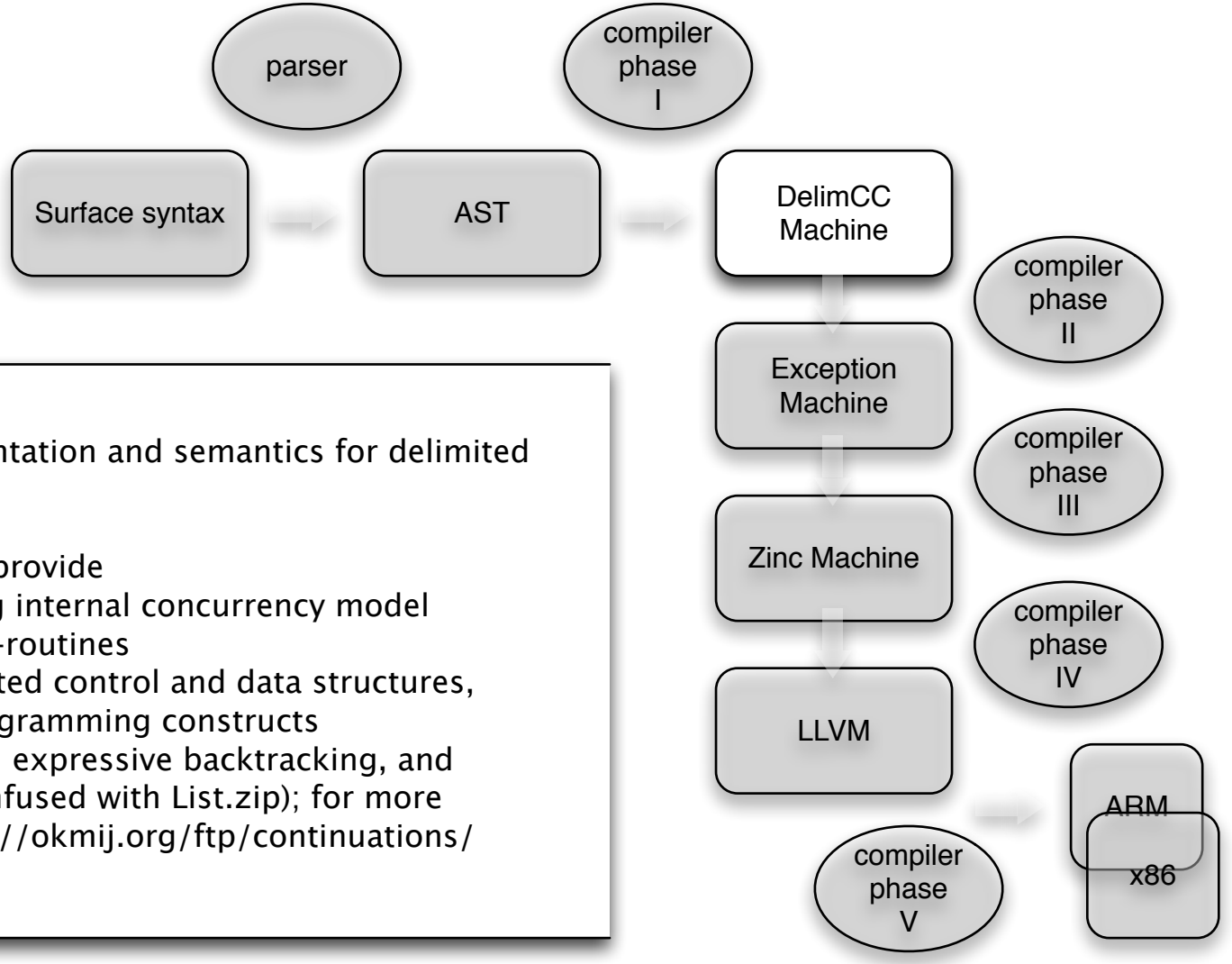
Values $v ::= x \mid \lambda x.\,e \mid p \mid D$

Contexts $D ::= \Box \mid De \mid vD \mid \texttt{pushP}\,D\,e \mid \texttt{pushSC}\,D\,e \mid \texttt{takeSC}\,D\,e$
$\mid \texttt{takeSC}\,p\,D \mid \texttt{pushP}\,p\,D$

Single Frame $::= \Box e \mid v\Box \mid \texttt{pushP}\,\Box\,e \mid \texttt{pushSC}\,\Box\,e \mid \texttt{takeSC}\,\Box\,e$
$\mid \texttt{takeSC}\,p\,\Box \mid \texttt{pushP}\,p\,\Box$

Transitions between configurations (e, D, q)

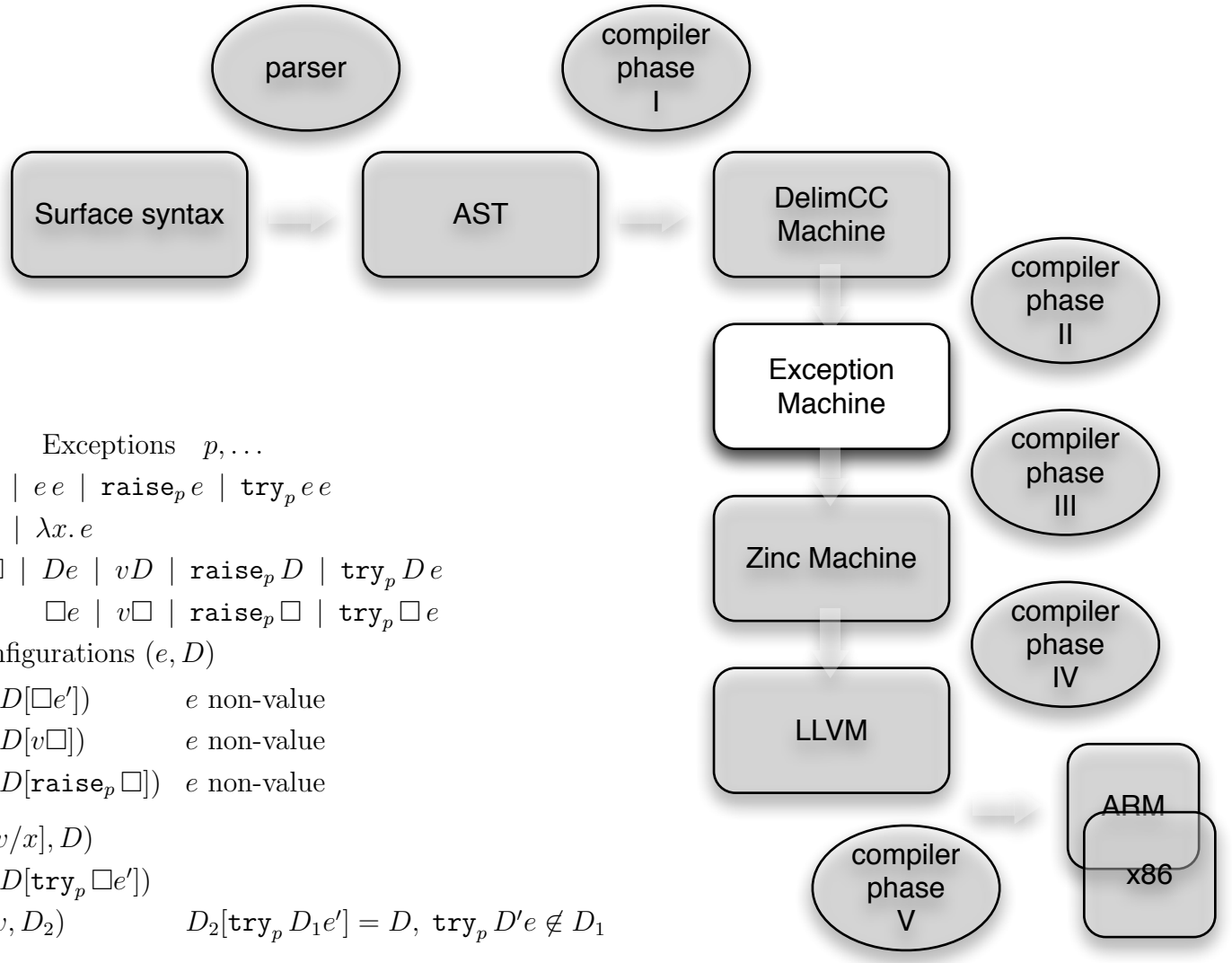
$$
\begin{array}{rcll}
(ee', D, q) & \mapsto & (e, D[\Box e'], q) & e \text{ non-value} \\
(ve, D, q) & \mapsto & (e, D[v\Box], q) & e \text{ non-value} \\
(\texttt{pushP}\,ee', D, q) & \mapsto & (e, D[\texttt{pushP}\,\Box e'], q) & e \text{ non-value} \\
(\texttt{takeSC}\,ee', D, q) & \mapsto & (e, D[\texttt{takeSC}\,\Box e'], q) & e \text{ non-value} \\
(\texttt{takeSC}\,pe, D, q) & \mapsto & (e, D[\texttt{takeSC}\,p\Box], q) & e \text{ non-value} \\
(\texttt{pushSC}\,ee', D, q) & \mapsto & (e, D[\texttt{pushSC}\,\Box e'], q) & e \text{ non-value} \\
\\
((\lambda x.\,e)v, D, q) & \mapsto & (e[v/x], D, q) & \\
(\texttt{newP}, D, q) & \mapsto & (q, D, q+1) & \\
(\texttt{pushP}\,pe, D, q) & \mapsto & (e, D[\texttt{pushP}\,p\Box], q) & \\
(\texttt{takeSC}\,pv, D, q) & \mapsto & (vD_1, D_2, q) & D_2[\texttt{pushP}\,pD_1] = D,\ \texttt{pushP}\,pD' \notin D_1 \\
(\texttt{pushSC}\,D'e, D, q) & \mapsto & (e, D[D'], q) & \\
\\
(v, D[D_1], q) & \mapsto & (D_1[v], D, q) & D_1 \text{ single frame} \\
(\texttt{pushP}\,pv, D, q) & \mapsto & (v, D, q) &
\end{array}
$$



Provides explicit representation and semantics for delimited continuations.

Delimited continuations provide

- ✯ Target for interpreting internal concurrency model
- ✯ Green threads and co-routines
- ✯ Many other sophisticated control and data structures, as well as generic programming constructs most notably: fair and expressive backtracking, and Zippers (not to be confused with List.zip); for more information see: http://okmij.org/ftp/continuations/



Variables $x, y, \ldots$ Exceptions $p, \ldots$

Expressions $e ::= v \mid e\,e \mid \mathtt{raise}_p\, e \mid \mathtt{try}_p\, e\, e$

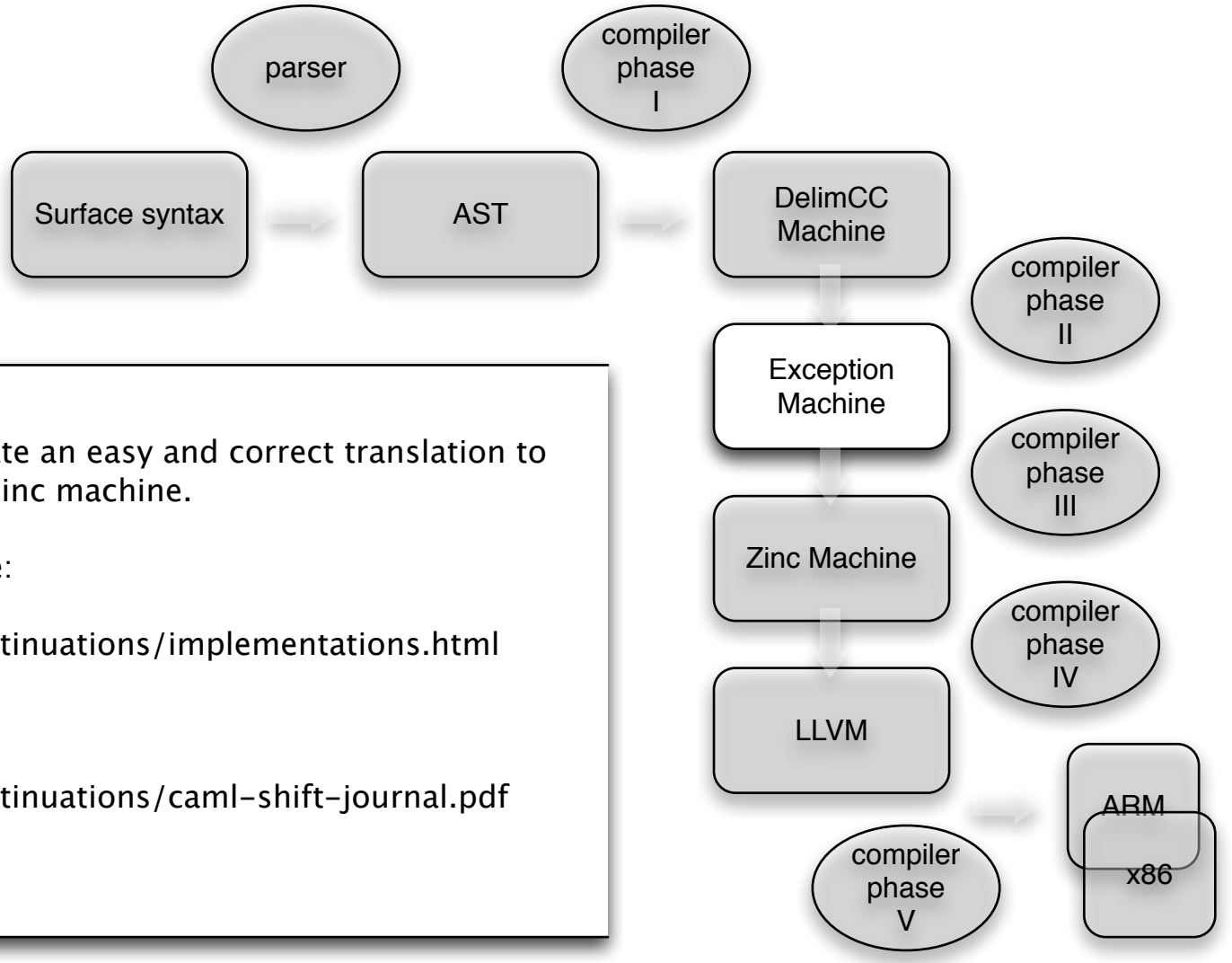
Values $v ::= x \mid \lambda x.\, e$

Contexts $D ::= \Box \mid De \mid vD \mid \mathtt{raise}_p\, D \mid \mathtt{try}_p\, D\, e$

Single Frame $::= \Box e \mid v\Box \mid \mathtt{raise}_p\, \Box \mid \mathtt{try}_p\, \Box\, e$

Transitions between configurations (e, D)

$$
\begin{array}{rcll}
(ee', D) & \mapsto & (e, D[\Box e']) & e \text{ non-value} \\
(ve, D) & \mapsto & (e, D[v\Box]) & e \text{ non-value} \\
(\mathtt{raise}_p\, e, D) & \mapsto & (e, D[\mathtt{raise}_p\, \Box]) & e \text{ non-value} \\
((\lambda x.\, e)v, D) & \mapsto & (e[v/x], D) & \\
(\mathtt{try}_p\, ee', D) & \mapsto & (e, D[\mathtt{try}_p\, \Box e']) & \\
(\mathtt{raise}_p\, v, D) & \mapsto & (e'v, D_2) & D_2[\mathtt{try}_p\, D_1 e'] = D,\ \mathtt{try}_p\, D' e \notin D_1 \\
(v, D[D_1]) & \mapsto & (D_1[v], D) & D_1 \text{ single frame} \\
(\mathtt{try}_p\, ve', D) & \mapsto & (v, D) &
\end{array}
$$



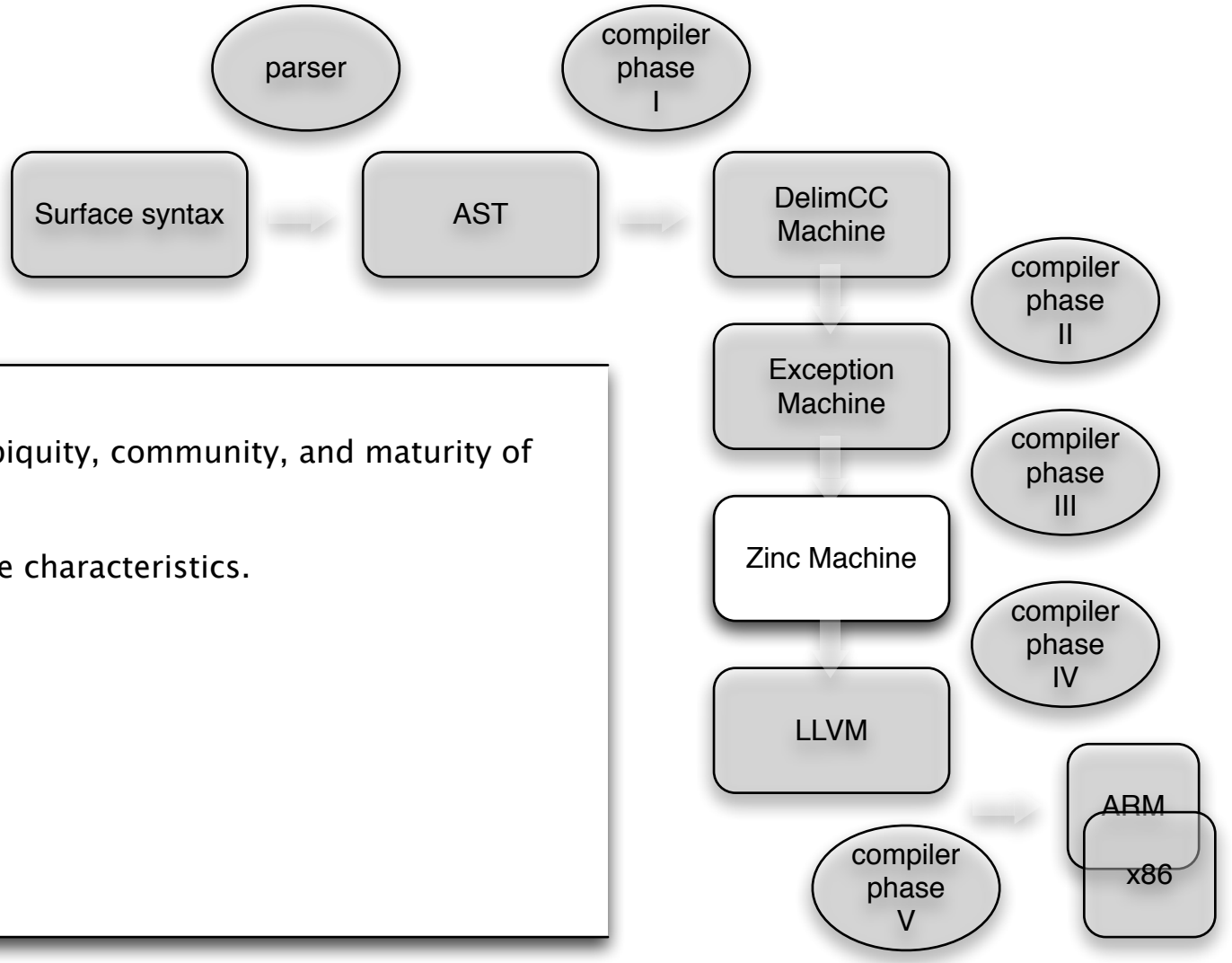
Principally here to facilitate an easy and correct translation to the target machine, the Zinc machine.

For more information see:

http://okmij.org/ftp/continuations/implementations.html

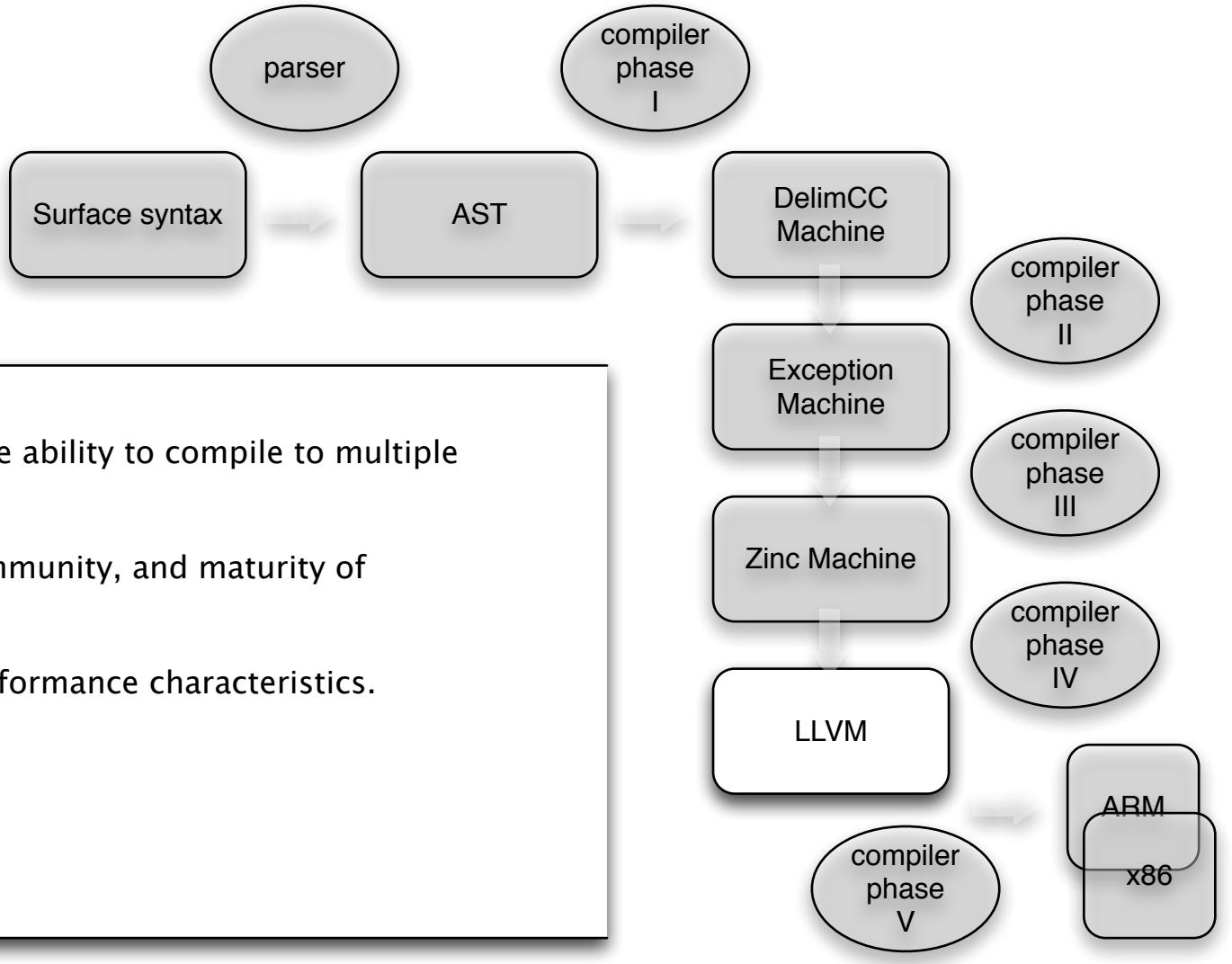
and especially

http://okmij.org/ftp/continuations/caml-shift-journal.pdf



Principally here due to ubiquity, community, and maturity of the technology.

It has known performance characteristics.



Principally here to provide ability to compile to multiple hardware targets.

Also enjoys ubiquity, community, and maturity of the technology.

Toolchain has known performance characteristics.