Optimizing The Indexer

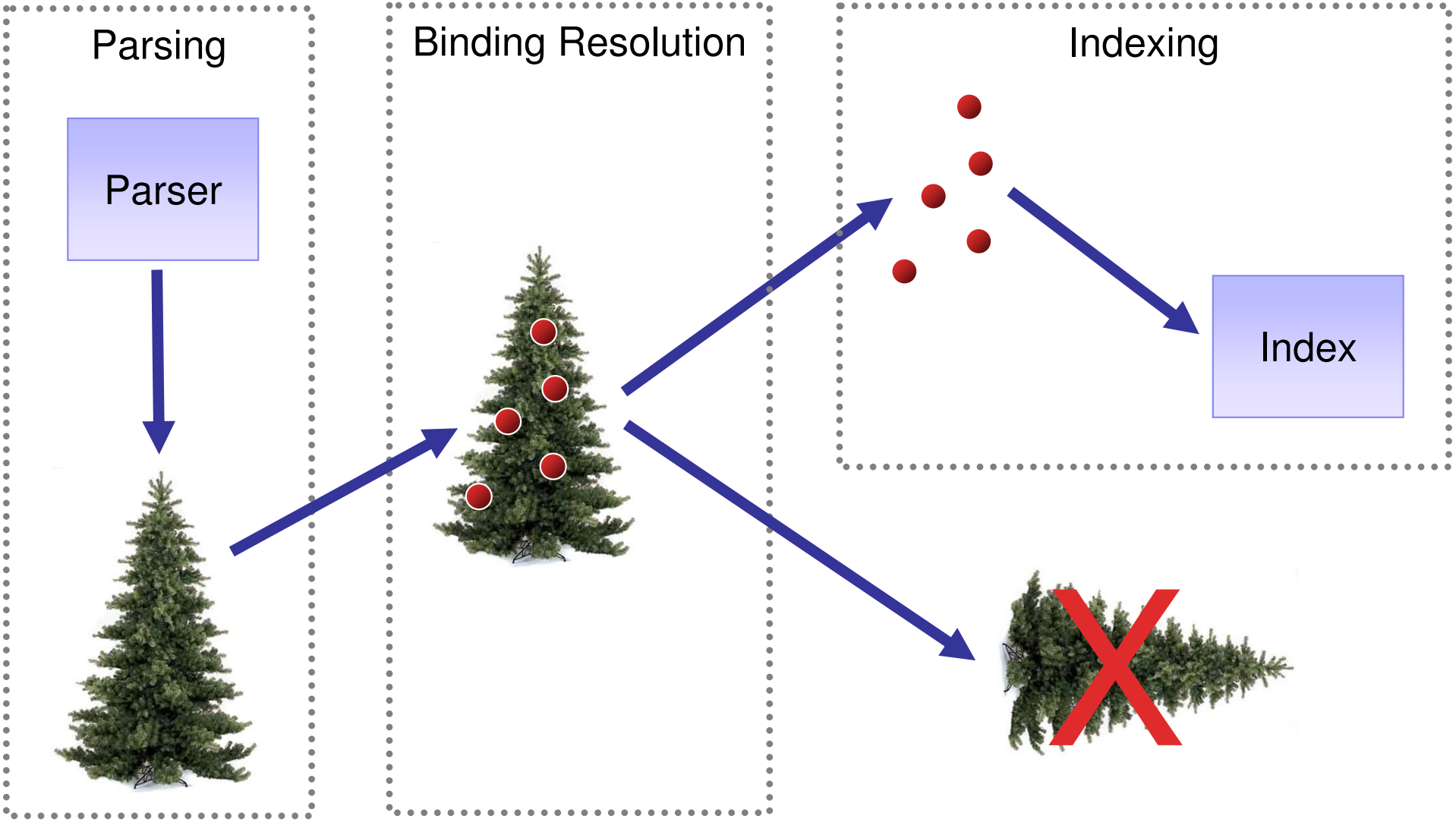
Mike Kucera
Jason Montojo
IBM Eclipse CDT Team

Indexer Optimizations

- We have developed a prototype parser that is capable of indexing much faster than the DOM parser.
  - Prototype is limited in functionality at the moment.
  - Proof of concept to demonstrate that optimization approach is sound.
  - Based on C99 parser.

- Demo

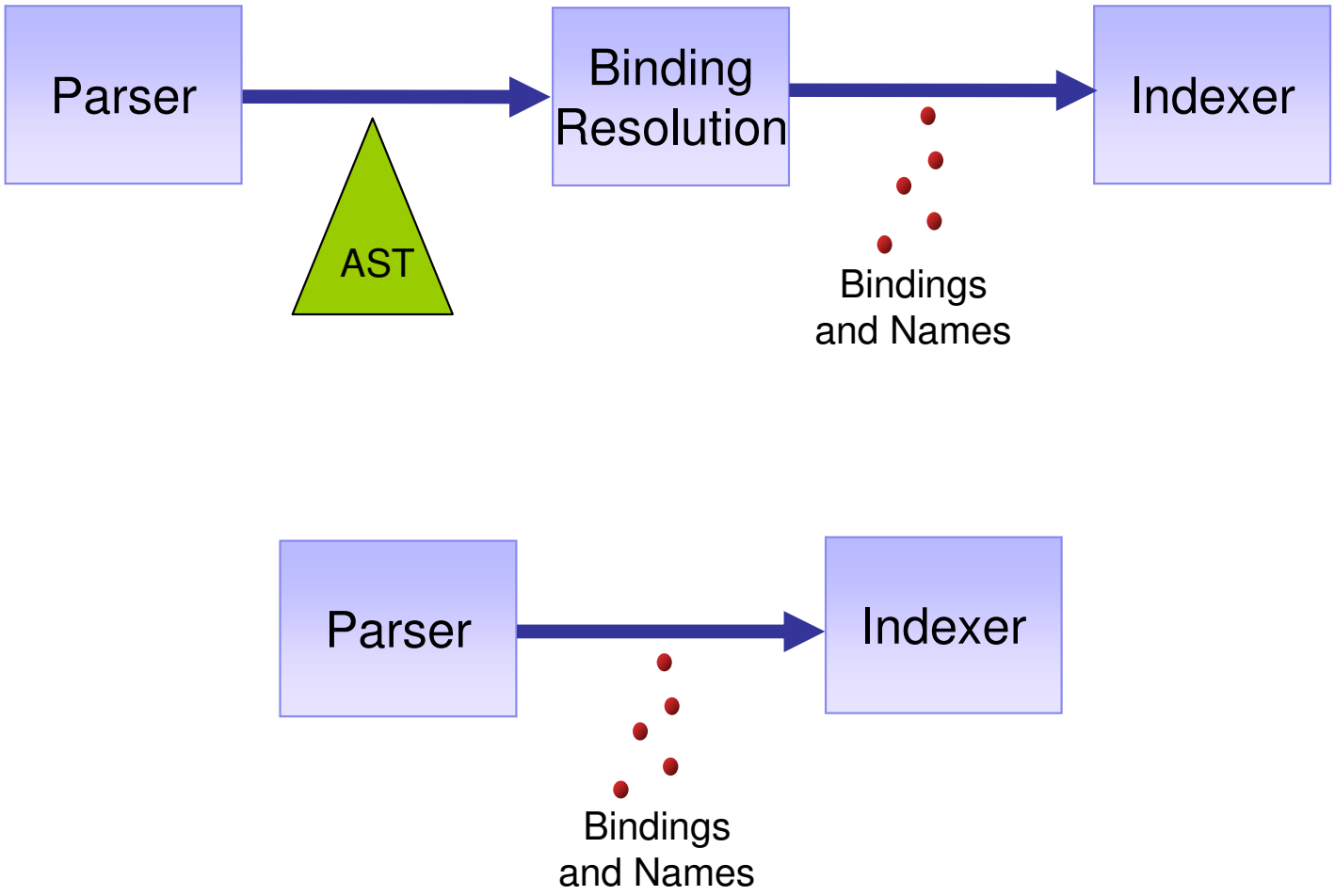
DOM Parser: AST-Based Indexing



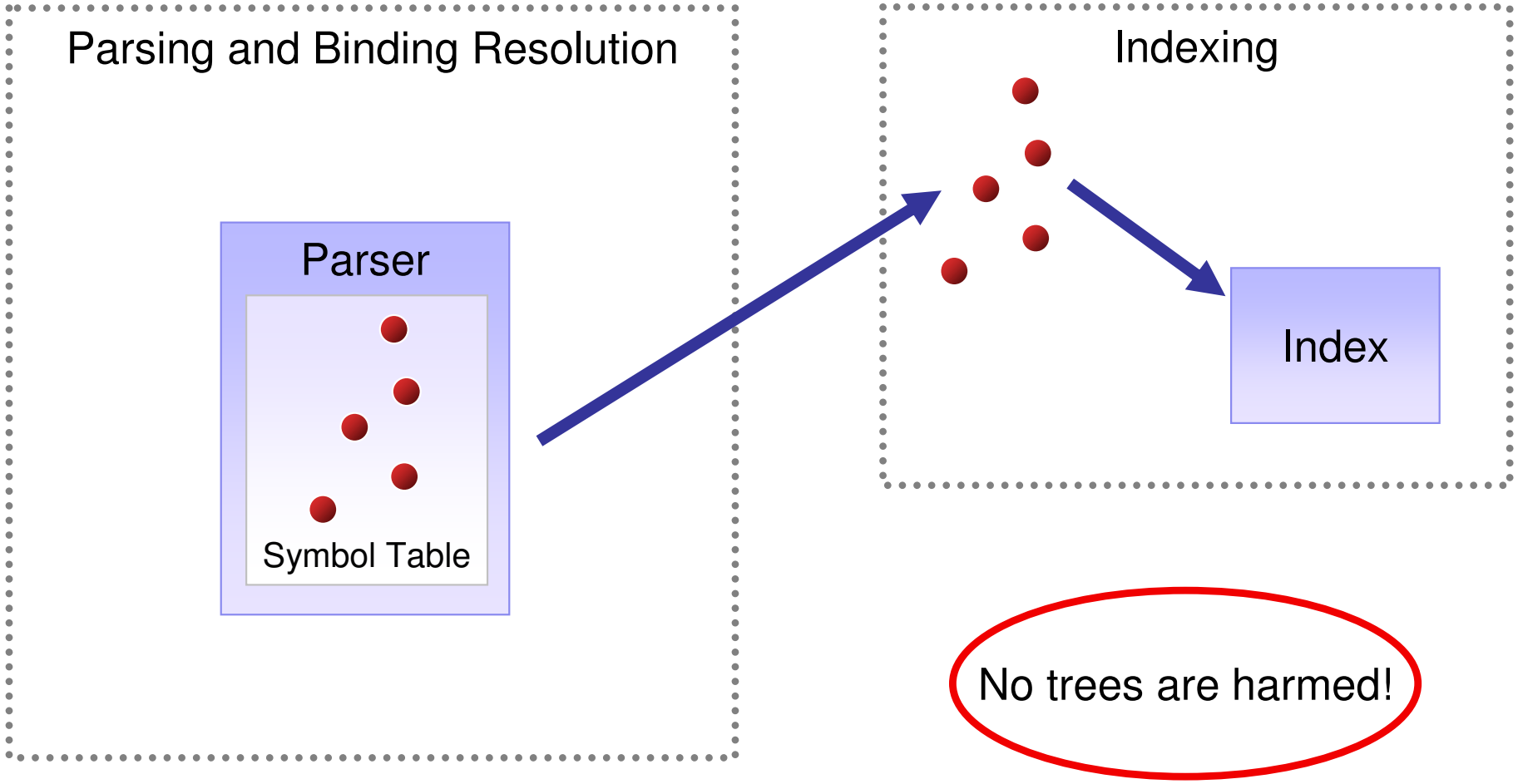
AST Based Binding Resolution

- AST is a large data structure
  - Contains information not needed by the indexer
  - Binding resolution involves a huge amount of tree traversal
- Assumption
  - Binding resolution is slow
- Solution used by CDT
  - Lazy binding resolution
  - Searches through the AST for binding info
  - Efficient if you want only one binding
  - Tree traversal algorithms are incredibly complex

Solution – Eliminate the Middleman



Symbol Table-Based Indexing



Symbol Table

- Maps identifiers to IBinding objects
- Simple hash table, with a stack to keep track of scopes
  - Constant time O(1) lookup on average
- Binding resolution algorithm is straightforward
  - When the parser encounters a declarator it creates a binding object and inserts it into the symbol table.
  - When the parser encounters an identifier it looks up the binding in the symbol table, Bam!... binding resolved.
- Parser accuracy
  - More semantic information available during parse.