ADIC 2.0 Status and Plans

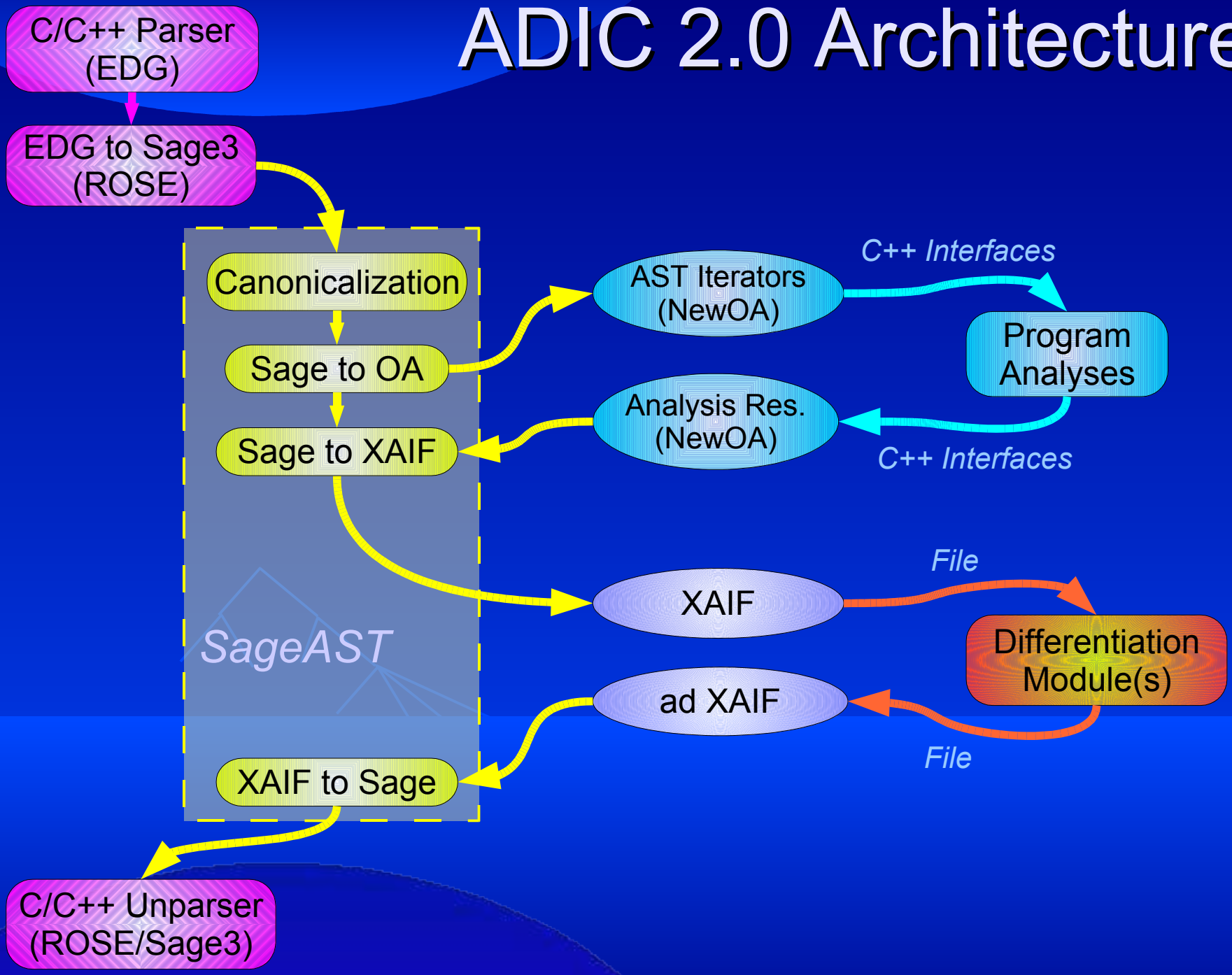
ADIC 1.1

- Currently distributed, handles most of ANSI C, a little C++
- Based on old (abandoned) research C/C++ parser, Sage++
- Available differentiation modules:
  - Jacobian
  - Hessian

What's new in ADIC 2? Everything

- Underlying software
  - New C/C++ parser and unparser: EDG
  - New AST toolkit: Sage3/ROSE
  - New differentiation modules
- Internals
  - XAIF representation
    - forward mode
  - Integration with OpenAnalysis
    - call graph, control flow, memory access

ADIC 2.0 Architecture



XAIF

- Language-independent program representation (XML), nested graphs:
  - Call graph (including symbol tables)
    - Control flow graph
      - Expression DAG
- XAIF 1.0: http://www.mcs.anl.gov/xaif

XAIF Example

```
<xaif:CallGraph prefix="ad_TempVar" program_name="None" xmlns:xaif="http://www.mcs.anl.gov/XAIF"
                xmlns:x si="http://www.w3.org/2001/XMLSchema-instance"
                xsi:schemaLocation="http://www.mcs.anl.gov/XAIF /home/nor ris/adic-2.0/share/xaif.xsd">
 <xaif:ScopeHierarchy>
  ...
 </xaif:ScopeHierarchy>
 <xaif:ControlFlowGraph controlflowgraph_scope_id="2" scope_id="1" symbol_id="foo"
vertex_id="136548200 ">
  <xaif:Entry vertex_id="1"></xaif:Entry>
  <xaif:BasicBlock scope_id="1" vertex_id="2">
   <xaif:Marker annotation="begin_135925608" statement_id="0"></xaif:Marker>
   <xaif:Assignment statement_id="135925608">
    <xaif:AssignmentLHS>
     <xaif:SymbolReference annotation="136553552" scope_id="3" symbol_id="t"
                           vertex_id="1"></xaif:SymbolReference>
    </xaif:AssignmentLHS>
    <xaif:AssignmentRHS>
     <xaif:VariableReference vertex_id="1">
      <xaif:SymbolReference annotation="136552488" scope_id="3" symbol_id="*p_136552488"
                            vertex_id="1"></xaif:SymbolReference>
     </xaif:VariableReference>
    </xaif:AssignmentRHS>
   </xaif:Assignment>
   <xaif:Marker annotation="end_135925608" statement_id="0"></xaif:Marker>
  </xaif:BasicBlock>
  <xaif:Exit vertex_id="3"></xaif:Exit>
  <xaif:ControlFlowEdge edge_id="1" source="1" target="2"></xaif:ControlFlowEdge>
  <xaif:ControlFlowEdge edge_id="2" source="2" target="3"></xaif:ControlFlowEdge>
 </xaif:ControlFlowGraph>
</xaif:CallGraph>
```

Ongoing work

- Language support:
  - Full support of ANSI C and C++ (except templates): templates, operator overloading
  - File scope issues
- Intrinsics definition
- Exception handling at points of nondifferentiabilty
- OpenAnalysis
  - alias, UDDU chains, reaching definitions, side-effect, activity analysis
- Test suite

Future work

- Reverse mode support
  - canonicalization
  - reverse mode templates
  - checkpointing
- Extend C++ template support
- Sparsity (static & dynamic)
- MPI codes
- ADIC server extensions