SAND2016-6996D

CASA

Extend xdm to Support T-Spice

Xyce Data Model Integration

Rachel Campbell, Oklahoma State University

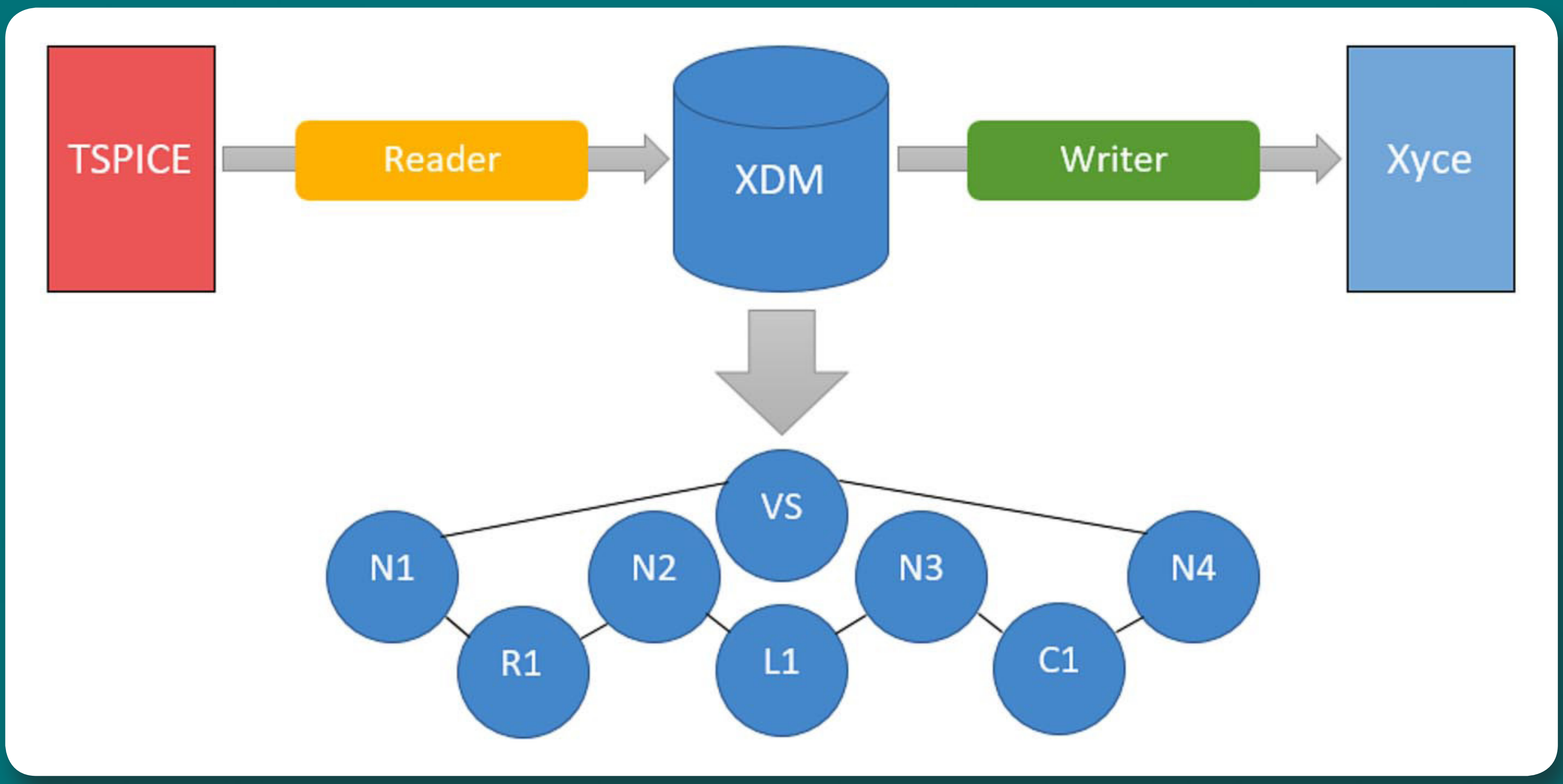
Project Mentors: Randy Lober, Org. 5561; Jonathen Kwok, Org. 8973

Problem Statement:

Xyce is a high performance, parallel electrical circuit simulation code (Org. 1355). Xyce Data Model (xdm, Org. 5561/8973) is a collaborative effort to compliment the current Xyce input parser with a modernized LL parser and abstract data model. The xdm LL parser and data model leverage the open source Boost Spirit parsing engine and currently support conversions between both Pspice and Spectre netlist formats into Xyce netlist and model formats.

Objective:

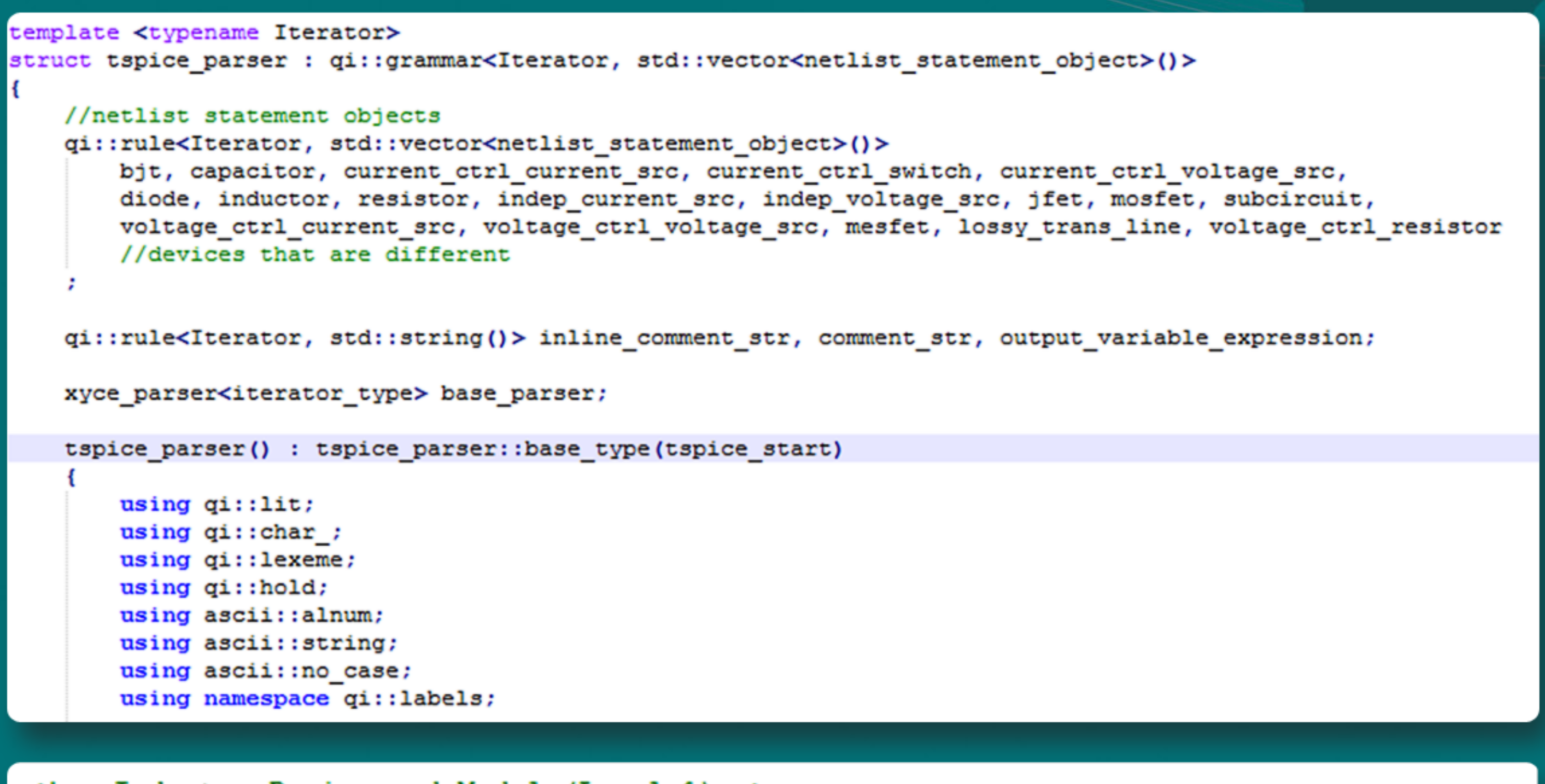
The objective is to extend support in xdm for T-Spice netlist data which represents another significant electrical design format (Mentor Graphics). The successful completion of this work will enable a new and third customer base (Org. 5443) for Xyce and xdm.



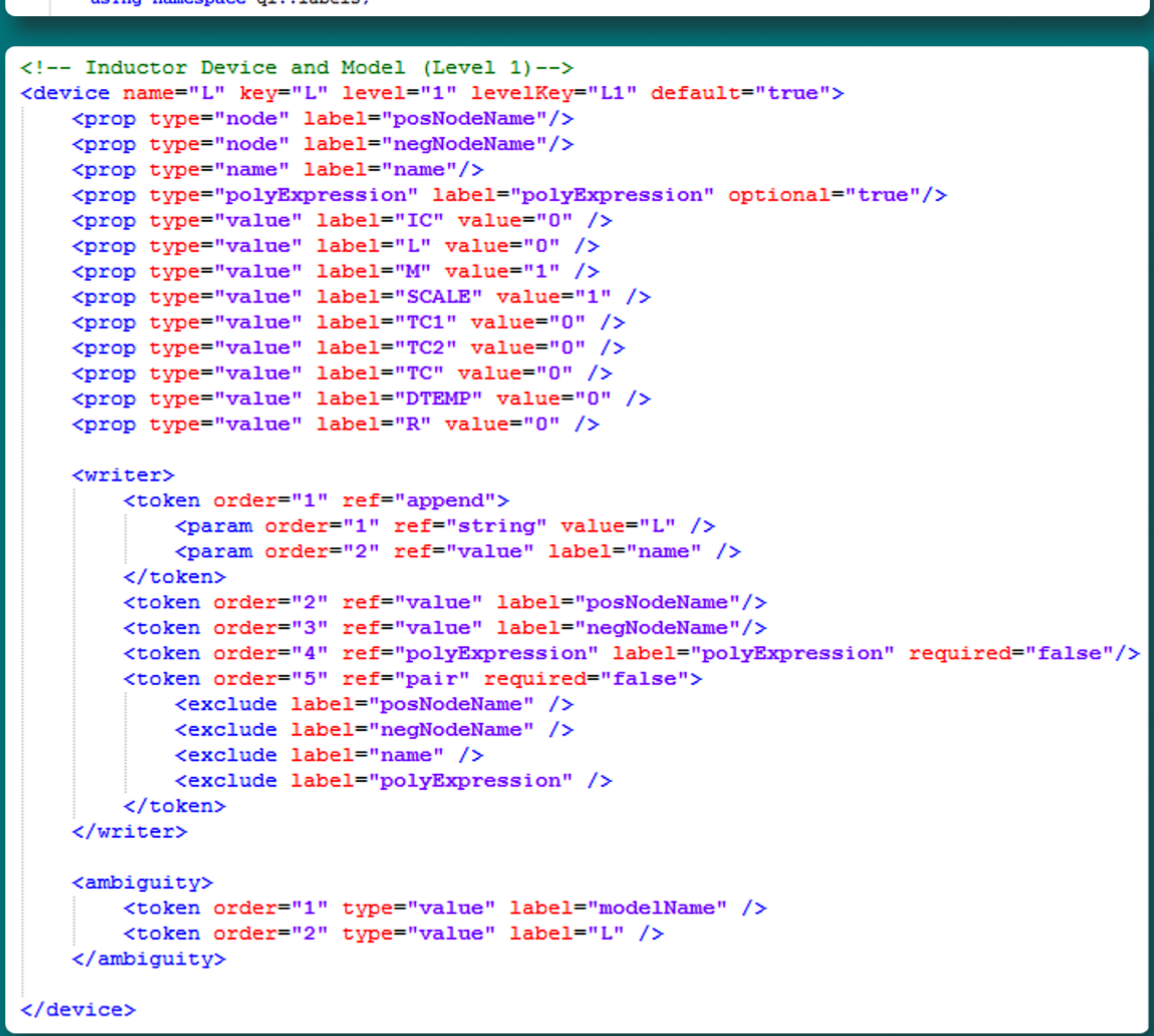
Approach:

In order to extend xdm to support T-Spice, C++ class definitions containing the grammar of T-Spice and xml containing the syntax of T-Spice were developed. A parser created from Boost Spirit had to be tailored to suit T-Spice netlists. Finally, changes had to be made to existing code to support T-Spice integration. For testing purposes, a unit test of this T-Spice reader was also implemented.

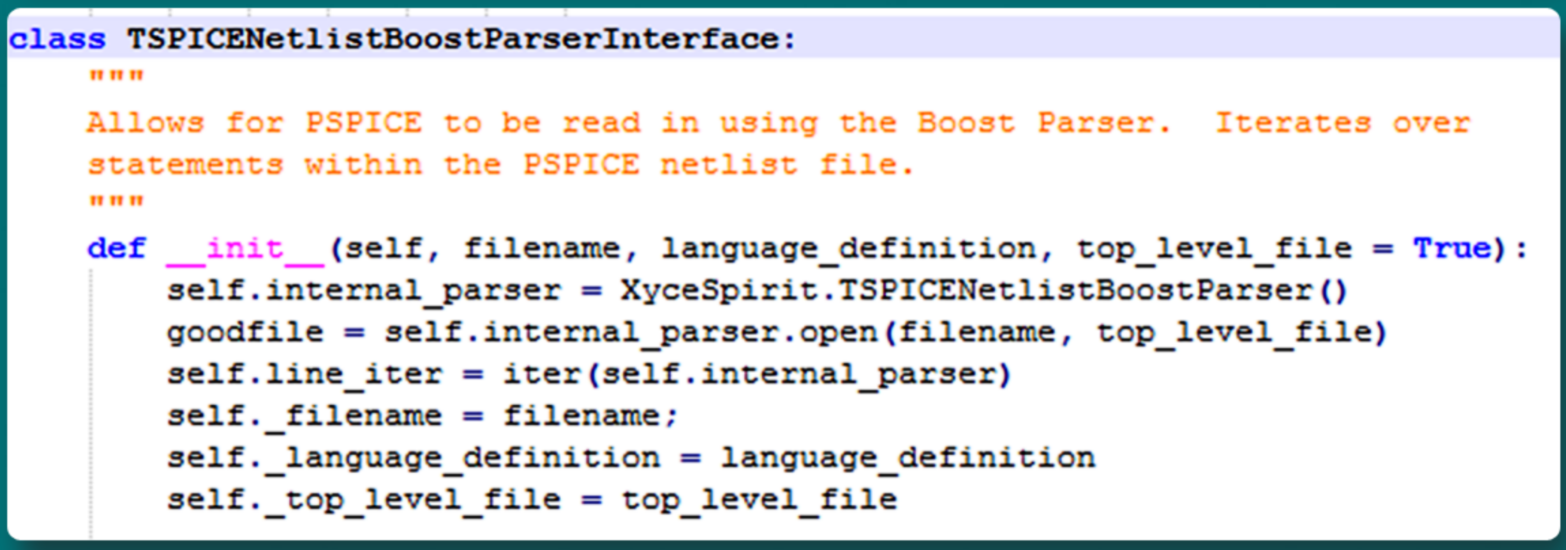
Results:



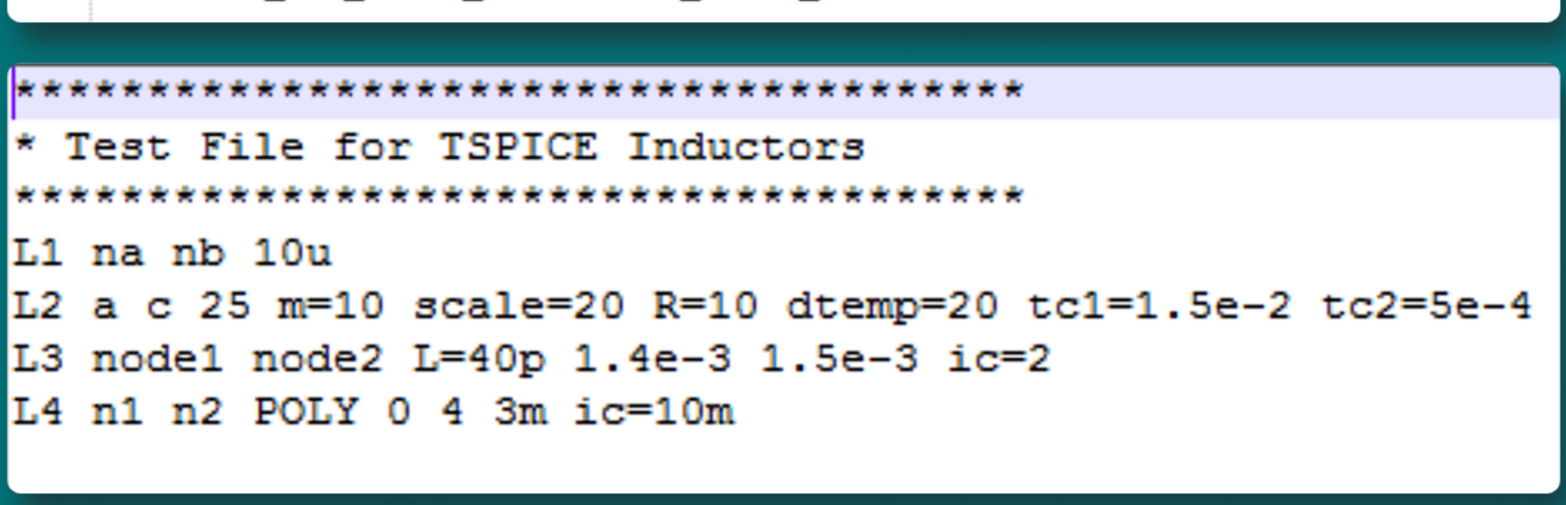
```cpp
template <typename Iterator>
struct tspice_parser : qi::grammar<Iterator, std::vector<netlist_statement_object>()>
{
    //netlist statement objects
    qi::rule<Iterator, std::vector<netlist_statement_object>()>
        bjt, capacitor, current_ctrl_current_src, current_ctrl_switch, current_ctrl_voltage_src,
        diode, inductor, resistor, indep_current_src, indep_voltage_src, jfet, mosfet, subcircuit,
        voltage_ctrl_current_src, voltage_ctrl_voltage_src, mesfet, lossy_trans_line, voltage_ctrl_resistor
        //devices that are different
    ;

    qi::rule<Iterator, std::string()> inline_comment_str, comment_str, output_variable_expression;

    xyce_parser<iterator_type> base_parser;

    tspice_parser() : tspice_parser::base_type(tspice_start)
    {
        using qi::lit;
        using qi::char_;
        using qi::lexeme;
        using qi::hold;
        using ascii::alnum;
        using ascii::string;
        using ascii::no_case;
        using namespace qi::labels;
```



```xml
<!-- Inductor Device and Model (Level 1)-->
<device name="L" key="L" level="1" levelKey="L1" default="true">
    <prop type="node" label="posNodeName"/>
    <prop type="node" label="negNodeName"/>
    <prop type="name" label="name"/>
    <prop type="polyExpression" label="polyExpression" optional="true"/>
    <prop type="value" label="IC" value="0" />
    <prop type="value" label="L" value="0" />
    <prop type="value" label="M" value="1" />
    <prop type="value" label="SCALE" value="1" />
    <prop type="value" label="TC1" value="0" />
    <prop type="value" label="TC2" value="0" />
    <prop type="value" label="TC" value="0" />
    <prop type="value" label="DTEMP" value="0" />
    <prop type="value" label="R" value="0" />

    <writer>
        <token order="1" ref="append">
            <param order="1" ref="string" value="L" />
            <param order="2" ref="value" label="name" />
        </token>
        <token order="2" ref="value" label="posNodeName"/>
        <token order="3" ref="value" label="negNodeName"/>
        <token order="4" ref="polyExpression" label="polyExpression" required="false"/>
        <token order="5" ref="pair" required="false">
            <exclude label="posNodeName" />
            <exclude label="negNodeName" />
            <exclude label="name" />
            <exclude label="polyExpression" />
        </token>
    </writer>

    <ambiguity>
        <token order="1" type="value" label="modelName" />
        <token order="2" type="value" label="L" />
    </ambiguity>

</device>
```



```python
class TSPICENetlistBoostParserInterface:
    """
    Allows for PSPICE to be read in using the Boost Parser.  Iterates over
    statements within the PSPICE netlist file.
    """
    def __init__(self, filename, language_definition, top_level_file = True):
        self.internal_parser = XyceSpirit.TSPICENetlistBoostParser()
        goodfile = self.internal_parser.open(filename, top_level_file)
        self.line_iter = iter(self.internal_parser)
        self._filename = filename;
        self._language_definition = language_definition
        self._top_level_file = top_level_file
```



```
**************************************
* Test File for TSPICE Inductors
**************************************
L1 na nb 10u
L2 a c 25 m=10 scale=20 R=10 dtemp=20 tc1=1.5e-2 tc2=5e-4
L3 node1 node2 L=40p 1.4e-3 1.5e-3 ic=2
L4 n1 n2 POLY 0 4 3m ic=10m
```

Impact and Benefits:

T-Spice support in xdm will remove the need for T-Spice electrical designers at Sandia to hand convert their models to use the Xyce modeling and simulation capability: this capability further strengthens the generality of the xdm abstract data model and expands the impact of the Xyce simulation code.