The State of SIDL: Quarterly Status Report

Scott Kohn
with
Tammy Dahlgren, Tom Epperly, and Gary Kumfert

Center for Applied Scientific Computing
Lawrence Livermore National Laboratory

October 2, 2001

Overview

- Babel status report for v0.6 (release mid-October)
- Where to send feedback and comments
- Adding component semantics descriptions to SIDL
- Build and component deployment issues
- CCA-compliant framework using SIDL (Gary)
- Integration path discussion (all)

Babel release update (v0.6)

- **Early release motivated by example CCA framework**
- **New capabilities**
  - Java client finished (except for arrays of objects)
  - Python server finished (Python client already completed)
  - minor changes in C casting support by user request
  - simplified command-line arguments for Babel driver
  - numerous bug fixes for memory leaks
  - multi-language exceptions (e.g., throw in C++, catch in Java)
  - expanded support for component dynamic loading
  - 3200 test cases in nightly regression test suite
  - regression tests pass on Linux, Solaris, Cygwin (Windows)…
  - CCA framework (to be discussed later by Gary)

Planned capabilities (v0.7)

- **Release target date – just before next CCA meeting**

- **Some planned capabilities (beyond v0.6)**
  - improved portability (please help us prioritize platforms)
  - finish Java server-side support
  - more Python test cases
  - expand documentation in “Babel Users’ Guide”
  - generate methods in the order of SIDL declaration (not sorted)
  - improved parser error messages (maybe haiku or Klingon, too)
  - (your suggestion here…)

We want your comments!

- **Seriously, we want feedback and suggestions…**
  - need community buy-in to be successful
  - want language bindings to be as natural as possible
  - topics: array mappings, C++ language bindings, …
- **Please be patient – we can't always do what you want**
  - certain multi-language issues or IDL technology constraints
  - but we'll work with you to try to come up with a good solution
- **Contact information**
  - project web site: http://www.llnl.gov/CASC/components
  - bug web site: http://www-casc.llnl.gov/bugs
  - project mail alias: components@llnl.gov
  - mail lists: babel-annouce@llnl.gov and babel-users@llnl.gov

Tammy is investigating component semantics in SIDL for her Ph.D. research

- **Why component semantics?**
  - describe component interface constraints
  - frameworks can check constraints at connection time
  - automatically generate run-time checking code via SIDL/Babel

- **Possible semantics approaches (from literature)**
  - argument constraints (pre- and post-conditions)
  - method invocation sequencing using object state diagrams
  - component properties (e.g., machine dependencies)

Vector example: State with pre- and post-conditions

```
interface vector {
  state {
    uninitialized, initialized
  };
  void setData(in double data)
    postcondition {
      initialized;
    };
  double dot(in vector v)
    precondition {
      v != null; v.size == self.size;
      initialized;
    };
}
```

Vector can be in one of these two states

Initializes vector and transitions to initialized state

Check if vector argument is not null and proper size

Matrix example: Using state constraints

```
interface Matrix {
  state {
    uninitialized, initialized, assembled
  };

  void setData(in Matrix data)
    precondition { data != null }
    postcondition { initialized; };

  void assemble()
    precondition { initialized; }
    postcondition { assembled; };
}
```

Matrix states

Initialize matrix data and transition to initialized state

Assemble the matrix and transition from initialized to assembled

Tammy wants feedback concerning semantics for scientific components!

- **When using your components…**
  - what errors do others make?
  - where is most of their time spent?
- **What features are important for determining (scientific) component compatibility?**
- **What kind of specification-related compatibility information do you think should be added to SIDL?**

Please direct suggestions to:

Tammy Dahlgren, dahlgren1@llnl.gov, 925-423-2685

Build and deployment issues

- **Heads-up: this is going to be important for CCA**
  - building portable shared libraries is difficult for C++
  - can be difficult due to compiler idiosyncrasies (e.g., ALPS)
  - need common deployment method for components (e.g., jar)
- **Build/deployment almost as important as interfaces**
  - if you can't build and link to it, it's useless
  - must be easy for non-experts to build portable re-useable software
- **We will need to address these issues in the future**

And now, Gary: Babel/SIDL CCA framework

Work performed under the auspices of the U. S. Department of Energy by the University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48