TBASSCO – Template-based Assurance of Semantic Interoperability in Software Composition

Robert Neches, Ke-Thia Yao, In-Young Ko
USC Information Sciences Institute

DASADA Kickoff Meeting
Santa Fe, New Mexico
Sept 11-14, 2000
Component-based Software Development

How can I build an correct application with these components?

How do I best configure this application for my local environment?

Component Developer

How do I make a new component interoperable?

Component Database
- by function, by I/O data

Architectural Styles
- synchronous, asynchronous

System Architect/Administrator

How do I best configure this application for my local environment?
Challenges

- Current tools help up to a point
  - Assume exact matching on data types and behaviors
    - Plug-compatible substitutions: yes
    - *Adaptation* of closely related components: no
  - Capture interface and method specifications
    - Syntax level composition: yes
    - Qualitative considerations in composition: no
      - implementation effort
      - performance
      - semantics

- Need better component / system metadata
  - Help developers understand what they’re working with
  - Support advanced component selection and adaptation
  - Make gauge implementation feasible
TBASSCO Hypothesis

We can:

- Develop a *maintainable, understandable* metadata framework for describing software components, and

- Support *useful* comparison of components wrt:
  - Relative ability to perform desired functions;
  - Modifications needed to handle each other's i/o;
  - Relative performance of resulting alternative systems.
TBASSCO Sub-hypotheses

- “Simple” metadata representations are sufficient
- Maintaining component library is not unduly burdensome
- Effort of maintaining the component library is repaid when systems are assembled
Experimental Approach

- Test with real-world component application: GeoWorlds
- Development tasks requiring assembly of sensible systems
  - Evaluate effort required to build the metadata
  - Build gauges that use objective measures
  - Validate the gauges' measures against expert assessments
  - Track the effort required to set up and use the gauges
  - Assess the expressiveness of metadata representation
GeoWorlds: Test-bed Application

- Large component-based system in use at PACOM
- Geographic Information Systems plus Web processing
- Framework for adding components

- Operations and intelligence uses
  - Mapping terrorist bombings
  - Locating recurring natural disasters
  - Investigating drug trafficking and piracy in various locales
## Tested by Four Analysts

### Natural disasters in China and India
(for liaison to Pacific Disaster Center)

- Six different analyses on Thailand/Burma:
  - Thai Political Parties / Key Political Figures
  - Thai Royal Family
  - Thai Military and Politics
  - Khmer Rouge Officials
  - "Press Coverage on Narcotics" *
  - "Involvement of X with Drugs in Burma" *

(* The last two topics have been sanitized.)

### Patterns of terrorist bombings in the Philippines

- Estimated effort *w/o GeoWorlds*: 3-5 days per analysis

### Nationalist movements and nuclear testing incidents in Tahiti

- Handled 352 reports in 8 man-hours
- *W/o GeoWorlds*: "3-4 weeks of mundane, time consuming labor-intensive work."

### Results

- Analyses were performed in about an hour each
  - Five of six found and organized collections of well over a thousand documents apiece
  - Sixth found 112 documents; *w/o GeoWorlds*, the analyst had found only seven
- *W/o GeoWorlds*: "much more time consuming."
- Results forwarded to PACOM J2, recommended for inclusion in briefing to the CINC in preparation for his pending visit to that area
GeoWorlds Test-bed Application

- Illustrates common challenges
  - Restructuring new version releases
  - Integrating new functionality
  - Adapting for local environments
TBASSCO Technologies and Impact

- **Data flow scripting language**
  - Lets developers specify how components interoperate
  - Aids combining alternative components, supports gauge development

- **Functional and data gauges**
  - Help compose semantically correct, customized applications
  - Attack garbage-in-garbage-out errors

- **Component insertion gauges**
  - Indicate how new components are incompatible
  - Reduce time and effort required to register new components

- **Architecture-level selection & deployment gauges**
  - Adapt proposed architectures to computing environment, expected loads

- **Translators: data flow descriptions to ACME**
  - Interface our work to other DASADA projects
Technical Approach

• Template-based, ADL extensions that model and reason about component application systems semantics
Example: Semantically-based Application Scripting

Extended-ADL representation of the abstract system template

(a)

(b)

(c)

Metadata descriptions of the abstract system template

SubClassOf: Sensor function
Input: Mobile sensor status data
Output: Zero-dim geo-location

SubClassOf: Sensor data analyzer
Input: 2 Zero-dim geo-locations
Output: One-dim geo-location

SubClassOf: Geographical data displayer
Input: none
Output: none

Sensor
Location
Detector

Route
Planner

Geo-line
Displayer

((a))

(b)

(c)

S1 Sensor Location Detector
Lat-long Location Input

InstanceOf: Location input
Input: none
Output: Lat-long coord

InstanceOf: Sensor location detector
Input: S1 sensor status data
Output: Lat-long coord

ArcView Route Planner

InstanceOf: Route planner
Input: 2 Lat-long coords
Output: ArcView geo-line

InstanceOf: Sensor location detector
Input: S1 sensor status data
Output: Lat-long coord

ArcView Geo-line Displayer

InstanceOf: Geo-line displayer
Input: ArcView geo-line
Output: none

InstanceOf: Location input
Input: none
Output: US street address

MapQuest Route Planner

InstanceOf: Route planner
Input: 2 US street addresses
Output: MapQuest route data

MapQuest Geo-line Displayer

InstanceOf: Location input
Input: none
Output: US street address

InstanceOf: Geo-line displayer
Input: MapQuest route data
Output: none
Design-time Semantic Gauges
Component Insertion

New Components

- Location to insert
- Semantic differences

Language: same (Java)
Data: organization differ
Service: interface differ
Design-time Semantic Gauges
Application Scripting

Compatibility Gauge

Overall:
0% 50% 100%

Function:
Input 1:
Input 2:
Output 1:

Source: Keyword Extractor
Replc.: Place Name Extractor

To check the interoperability level of a connection

Interoperability Gauge

From:
Language Identifier
To:
Category Tree Viewer

To select an alternative component

Interoperability Gauge

From:
Category Comparison
To:
Document Summary

To select a component to be combined
Design-time Semantic Gauges
Architecture Selection

Data Flow Description

To select an architecture based on performance gauge feedback

Selection Gauge

Architectural Style
• Asynchronous
• Synchronous

ACME description

Architectural Description
TBASSCO – Template-based Assurance of Semantic Interoperability in Software Composition

**Impact**
- Lower cost by enabling the rapid assembly of semantically correct component systems
- Lower cost by facilitating integration of new components into component frameworks
- Increase performance by allowing system architects to tune system architecture

**New Ideas**
- Lightweight semantically-based metadata to describe components
- Granularity and organization schemas to identify new component mismatch
- Meta-level data flow to facilitate switch among architectural styles
- Semantic gauges to guide design

**Schedule**
- 9/00 – Joint collaboration and demonstration scenario for assured system assembly
- 4/01 – Component-level gauges to guide application and component development
  - Component compatibility, insertion gauges
- 4/02 – Architectural-level gauges to guide system level refinements
  - Architecture selection, deployment gauges
Collaboration Plans

Design Time Aids

- ISI
  - Gauges to select interoperable components
  - Gauges to determine difficulty of adding new components
  - Gauges to adapt architecture to computing environment

Existing Software
(From Library of Available GeoWorlds / SDC Components)

Prospective Software
(Same Library: Alternative Extensions, Compositions)

Run Time Aids

- BBN
  - Network (bandwidth, latency) gauges
  - Uptime gauges

- Columbia / WPI
  - Protocol gauges (partial matching on event posets)
  - Run-time gauge plugin and modify

- Object Services
  - Application profiling gauges and topology gauges on configuration, component usage
    - Component binding
    - Dead libraries
    - Versioning
    - Activity

Requirements / Capability Descriptions

- ISI
  - Semantic function and data descriptions

- BBN
  - Architecture requirement documents

Other Participants

- ISI
  - 2001
  - 2002

ACME ADL XML/FleXML