Content-Based Communication

Identify-based communication:
• Identities of one or more the communicating parties is necessary
• Examples:
  • CSP - both sender and receiver identity is needed
  • RPC/CORBA/SOAP - identity of receiver is needed

Content-based communication:
• Message delivery is achieved based on the message’s type/structure/values
• Examples:
  • tuple spaces
  • event models

Tuple Space Concepts
Tuple Space Operations

tuple: a series of typed fields
examples: ("label", 10, 2.15)
(5, "term")
(100)

Operations
• out(t) insert the tuple t into the tuple space (non-blocking)
• in(t) find and remove a “matching” tuple from the tuple space;
  block until a matching tuple is found
• rd(t) like in(t) except that the tuple is not removed
• eval(t) add the active tuple t to the tuple space

Tuple Matching

Let t(i) denote the ith field in the tuple t.

A tuple t given in a in(t) or rd(t) operation “matches” a tuple t’ in the tuple space iff:

1. t and t’ have the same number of fields, and
2. for each field
   if t(i) is a value then t(i) = t’(i)
   or
   if t(i) is of the form ?x then t’(i) is a valid value for the
   type of variable x

If more than one tuple in the tuple space matches, then one is selected
nondeterministically.

As a result of tuple matching if t(i) is of the form ?x, then x := t’(i)
Examples of Tuple Matching

The tuple defined by:

```plaintext
int i;
float f;
(“label”, i, f, 10)
```

Matches these:                     Does not match any of these:
(“label”, 20, 1.5, 10)           (“label, 20, 1.5)
and i := 20; f:= 1.5;                       (“label”, 20, 1.5, 2)
(“label”, 0, 2.7, 10)                        (“other”, 20, 1.5, 10)
and i:=0; f:=2.7                           (“label”, 20, 1.5, 5)

Client-Server Example

```
\[
\text{client}
\quad \text{out}
\quad \text{in}
\quad \text{in}
\quad \text{out}
\quad \text{out}
\quad \text{in}
\quad \text{in}
\quad \text{out}
\]
```

(`“server index”,i) (`“client”, i, resp)
(`“server”, i, req) (`“client”, i, resp)
Client-Server Example

server()
{ int index = 1;
 request req;
 response resp;
 . . .
 while(1) {
 in("server", index, ?req);
 //compute resp
 out("client", index, resp);
 index = index + 1;
 }
}

client()
{ int index;
 request req;
 response resp;
 . . .
in("server index", ?index);
 out("server index", index+1);
 . . .
 out("server", index, req);
in("client", index, ?resp);
}
Events

Definition

On-Line Computing Dictionary: an occurrence or happening of significance to a task or program.

Webopedia: an action or occurrence detected by a program.

High Tech Dictionary: An occurrence that is significant to a program, and which may call for a response from the program.

Examples

information monitoring: “tell me when the price of stock X drops below Y dollars per share”

collaborative computing: “tell me when this document has been changed by another user.”

command-control: “tell me when anyone enters the building”

Event Operations

terminology: publish-subscribe model
Event Delivery Concepts

Event Supplier
- time=0; produce event1
- time=2; produce event2
- time=4; produce event3
- time=6; produce event4

Event Model Infrastructure

Event Consumer X
- time=1; subscribe to events
- time=2; receive event2
- time=3; unsubscribe

Event Consumer Y
- time=1; subscribe to events
- time=2; receive event2
- time=4; receive event3
- time=5; unsubscribe

From: Nigel Edwards

Delivery Models

push-model
- producer send Event Service send consumer

pull-model
- producer send Event Service ask send consumer
- producer send Event Service ask send consumer

combination
- producer send Event Service ask send consumer
- producer send Event Service ask send consumer
Event Filters

Filter: a set of criteria determining which subset of events for which a consumer is subscribed are delivered to that consumer.

Filters may be based on: values, time, history.
Filters may alter an events properties: priority.

Typed Events

To support subscription, advertising, and filtering events are often structured (or typed).

Example: Corba Notification Service

From Nigel Edwards
Filter Expressions

Cambridge Event Model

**Event Definition:**

```
Badge: INTERFACE =
    Seen : EVENTCLASS [ badge : BadgeId;
                        sensor: SensorId];
END.
```

**General Filter Definition:**

```
template = EventTypeName( par1, …, par n);
```

**Examples:**

```
templateWhere = Seen(17, R);
templateWho = Seen (P, 29)
templateAll = Seen (P, R)
```

SIENA

**Filter Definition:**

A filter is specified by attribute names, their types and constraints on their values.

**Example:**

```
string event == account/*
time date >= 01.01.2000
float amount > 10000.00
```