# ORC: Highway or Cul-de-Sac?

M. Clint*, J. Gabarró, T. Harmer, P. Kilpatrick, R. Perrott, A. Stewart

Institutes:

(1) Queen's University Belfast, N. Ireland: QUB

(2) Universitat Politècnica de Catalunya: UPC

# Context

- ORC (Misra): notation for web orchestration
- Can ORC be fluently extended to embrace all of the essentials of Grid Computing?
- Plans
  - ⋆ Develop a realistic ORC specification of key Grid operations – for example, develop a probabilistic version of ORC.
  - ⋆ Develop a refinement calculus for ORC.
  - ⋆ Refine ORC specifications into operations of the GCM
  - ⋆ Use ORC to specify key features of the GCM.
  - ⋆ Develop an ORC-based Grid simulator for experimentation.

# Grid Component Model

A grid is a collection of sites. A site is a sextuple

$$gridsite == (N\ ,\ C\ ,\ S\ ,\ ID\ ,\ E\ ,\ M)$$

comprising

- a unique name, $N$,
- a set of components, or jobs, $C$,
- a collection of services $S$ that can be utilised by users,
- a directory $ID$ providing information about grid sites,
- an engine $E$ which has the potential to execute components to produce results,
- a local manager $M$.

# Components

A component is a sextuple

$$component == (E\ ,\ o\ ,\ f\ ,\ d\ ,\ MC\ ,\ VC)$$

comprising

- an external interface (a set of component names E). This defines input and output dependencies.
- an output component $o$ is simply an output file.
- a functionality $f$ may be supplied as a combination of program code and service invocations.
- data $d$ (a component may be used simply to store data).
- a minimum constraint (a predicate $MC$).
- a value constraint (an expression $VC$).

# Managers

The site manager interacts with the grid (and with applications programmers) and controls dynamic behaviour.

A manager controls:

- acceptance of new jobs for execution
- the range of services currently offered
- the updating of information directories through searches
- where appropriate, site selection, job placement and monitoring

# ORC (Misra): a language for site orchestration

ORC is a language for orchestrating the use of web resources by means of site calls.

A call to a site $S$ may update the local site, call other sites and reply.

Some special sites:

- if $b$ returns a signal if $b$ is true and remains silent otherwise;
- $RTimer(t)$ always responds after $t$ time units;
- *let* always returns (publishes) its argument.

# ORC Expressions

Let $E1$ and $E2$ be ORC expressions:

1. operator $>$ (sequential composition)
   $E1 > x > E2(x)$ evaluates $E1$, receives a result $x$, calls $E2$ with parameter $x$.
   If $E1$ produces two results, then $E2$ is evaluated twice.

2. operator $|$ (parallel composition)
   $(E1 \,|\, E2)$ evaluates $E1$ and $E2$ in parallel. Evaluation returns a merge of the individual output streams.

3. where (asymmetric parallel composition)
   $E1(x)$ where $x :\in E2$ begins evaluation of both $E1$ and $E2$ in parallel.
   Evaluation of $E1(x)$ may proceed until a dependency on $x$ is encountered.
   The *first* value delivered by $E2$ is returned in $x$ and $E2$ is terminated.

# Site Selection: find an appropriate site

Suppose that a user (or manager) knows a set of names of grid sites,

$$\mathcal{F} = \{s_1, \ldots, s_n\}.$$

An appropriate site on which to execute $c$ may be selected as follows:

$$Select(c, \mathcal{F}) \triangleq let(s) \text{ where } (s, v) :\in \{ \mid_{s_i \in \mathcal{F}} \quad s_i.can\_execute(c)\}$$

# Site placement: one example (multiple sites, dependent components)

---

Let $Place(u, \{c\}, \mathcal{F})$ be an expression by which a user $u$ places a component $c$ on a grid, $\mathcal{F}$.

Consider components $c_1$, $c_2$ where $c_1$ has output component name $f$ (i.e. an output file) and $c_2$ has an external interface (input) $\{f\}$.

$Hub(u, \{c_1, c_2\}, \mathcal{F}) \triangleq$

$let(r)$ where $r :\in (Place(u, \{c_1\}, \mathcal{F}) > f > Place(u, \{c_2, f\}, \mathcal{F}))$

Here, the user acts as a hub for controlling the orchestration.

## Probability

To reduce the risk of failure it is possible to construct an ORC expression with redundancy.

$$||Search \doteq \ \mathsf{let}(r) \ \mathsf{where} \ r :\in \{ \,|_{s_i \in \mathcal{F}} \ s_i\}$$

Assume that the site calls are statistically independent.
The probability of failure $\mathsf{Pr}(F_{||Search})$ is

$$\mathsf{Pr}(F_{||Search}) = \mathsf{Pr}(F_{S_1}) \times \mathsf{Pr}(F_{S_2}) \times \cdots \times \mathsf{Pr}(F_{S_n}).$$

Often grid events are not independent. For example,

$$z :\in (CNN \,|\, Rtimer(t) \gg CNN)$$

Here the CNN site is called at two different times. If the site is down at the time of the first call then the likelihood of it still being down on the second call is higher.
Conditional probability can be used to reason about such situations.

# Conclusion

- Is ORC a suitable basis for specification language for GRID features?
- Is it valuable to specify features of the GCM?
- Is the idea of a simulator useful?