Ahead, along Robin’s track

— hopefully

Davide Sangiorgi

University of Bologna

Milner Symposium, Concurrency Panel, 18 April 2012
JFIT '93
Conference
(Uiv. of Keele)
Plenary lecture by
Robin Milner
APPLICATION of THEORY —

or

THEORY from APPLICATIONS?

• Don't divorce science from application
  ~ "Science of the Artificial"; Herbert Simon
  ~ The subject matter is human constructors

• Don't defer science to application
  ~ Demand tempts us to do so
  ~ But intellectual resource is finite...
  ... else delay future application!
  ~ Human constructions rival nature's
    in complexity
  ~ Scientific faith!
How does an “informatic” scientist work in the “LABORATORY” of Applications?

TWO RHYTHMS:

- **FAST MOVEMENT:** ~ contract research ~ cooperative projects

- **SLOW MOVEMENT:** ~ observe phenomena of constructions (Systems, programming method, ...)
  ~ build models to inform future construction methods

**TODAY:**

- Look at the slow movement.
How SCIENTIFIC UNDERSTANDING may react with APPLICATION

APPLICATIONS

SCIENTIFIC

CULTURE

FOUNDATIONS

TIME
A FAILURE

A purist's idea gets nowhere...

A SMALL SUCCESS

... A one-off problem is solved
A BIGGER SUCCESS

- Observation attracts insight
- Meeting creates solutions...
- ... and precipitates stronger foundations

Simultaneously { An engineering discipline
A science of the artificial
Please don’t confuse Them!

...... WHAT FOUNDATIONS?
Strachey's Progress

1960

NEED BETTER LANGUAGE DESIGN

CPL LANGUAGE

INSIGHT FROM LOGIC

P. Landin
Today’s computing systems

– Distributed
even a single machine:
  * heterogeneity
  * failures
  * latency

– A vast number of independant entities

– Interaction

– Composition

Needed: a different way of thinking
where concurrency is central

[Milner: “computing is interaction”]
Example: web services

– distributed components that must interact and carry out dialogues to achieve some common goal

– they may be composed

NB: industrial languages for web services based on process calculi (eg BPEL wrt $\pi$-calculus)
Example of work in Bologna: Jolie

A fully-fledged service-oriented programming language

– interaction primitives of pi, including sessions
  + architectural constructs for service composition

– interface

– fault handling and compensations

– interoperability
  (programmable communication protocols, hooks towards different internal languages)

– usability and efficiency
  (light-weighted, network-layer optimizations, manipulation of structured data)

We are experimenting with it (eg, a start-up launched)
Challenges
– Linguistic primitives for dialogues among components
– Component discovery and contracts
– Adaptability and evolvability
  (cf: the Hats project)
– Failure and compensation
– Split and merge
  (from global descriptions to interacting local behaviours)
– Resource consumption
– Emergent behaviours
  (social networks)
Faults and failure recovery
Big impact:

- Main obstacle to distribution transparency
- European organisations with $> 50$ employees
  $\Rightarrow$ over 17 bilion E loss each year by IT downtime and recovery
  (13% EU budget for 2011)
- Amazon EC2 outage, April 2011
  * an error in a minor local reconfiguration
  * several causally-related events
  * each event locally meaningful, disastrous global effect

Hard to avoid faults

**Need:** programming primitives to cope with faults and recovery
The REVER project

[Bologna (Focus)/Inria Grenoble (Sardes)/Paris (PPS)]

Composable constructs for recoverable and dependable programming

– meaning of reversibility in distributed processes
– combination of reversibility and compensation
  (compositionality, hierarchical structures, evolvability)
– implementation
Integration
Integration...

– Technologies

– People

  NB: humans no more action observers!
  ⇒ humans act, computers coordinate

– Knowledge

  knowledge propagation, discovery of resources, planning
Integration...

Traditional concurrency with AI concepts

It is interesting to remark that AI, with its demand for a kind of programming vastly different from Fortran, has provided some of the impetus towards a study of semantics. Perhaps because a large part of AI’s subject is automated inference. Perhaps because the languages inspired by AI, such as LISP and POP2, were themselves challenging objects, since they were informed by the $\lambda$-calculus.

— Milner
Integration...

Models

Milner’s tower of models (2009)

“What distinguishes the science of informatics is that its artifacts demand explanation at many levels”

“... need to combine informatic models”

“Such combination is best seen as a construction, not a relationship; it combines the entities of different models, with extra behavioral description of how they interact.”
Metamodels and metatheory
Some 1991 slides from Milner

Processes as Objects

Robin Milner

Is there a calculus for concurrency as basic as λ calculus?
Conclusion

The $\pi$-calculus aims to be for processes what the $\lambda$-calculus is for functions.
Scientific status of the Tower of Models

- Useful models, and validations, may well be informal

- Different models suit different people, including non-experts

- Many instances of models and validations exist

- Can we derive languages from models, not vice-versa?
– From CCS and $\pi$-calculus to bigraphs
– From labelled operational semantics to reduction semantics
– From labelled bisimilarity to barbed bisimilarity

**Not really a change of mind !**

– foundational models well understood
– specific needs emerge
– a variety of models
A related issue: **emerging behaviours**

... from local to global

- social networks
- Amazon EC2 outage problem
- system management in distributed systems
  hard to predict the effect of a local event
  (eg, update or stop a machine)