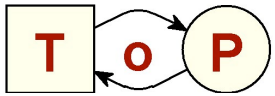
ECSS Zürich, Sept. 10, 2008

In Praise of a Theoretical Basis for New Software Paradigms

W. Reisig

Humboldt-Universität zu Berlin



Theory of Programming

Prof. Dr. W. Reisig

Software paradigms

“classical”,
OO,
components based,
model based,
service-oriented.

does all this require a (new) theoretical basis?

how it started, 1950ies and 1960ies

early days of informatics:
technology harmonized with theory.

Von-Neumann architecture: Turing machines.

Expressive power: Computable functions.

Compilers: Context free languages

Complexity: O(n)

University Teaching: much of theory

late 1960ies, early 1970ies

The “software crises”

Engineering principles to systematically develop software.

Theoretical contributions:
semantics of programming languages,
verification of programs.

Broadly taught in many university courses,
but did not catch on in industrial software design.

late 1970ies, 1980ies

modelling- and software architecture paradigms:

Algebraic Specifications,
Petri Nets,
Process Algebras,
Statecharts,
Temporal Logic.

Limited but stable interest in the software industry,
software tools supporting their use.

Teaching fell apart,
each department to select its own preferences.

1990ies

software architectures:
Middleware and internet software,
sparse theoretical considerations.

exception: arguments on performance and complexity.

Teaching:
concrete middleware languages (Corba),
little emphasis on general principles.

recent years

hype for the UML

Actual buzzwords:

model driven software design,

service orientation,

cloud computing,

the internet of things.

Actual text books:

- in plain English + informal graphics,
- data structures represented in XML,
- behaviour represented in pseudocode

We need clear statements about

hype for the UML

Actual buzzwords

model driven software design,

service orientation,

cloud computing,

the internet of things.

- the meaning and logical structure of constructs;
- the expressive power of specification techniques;
- equivalence, correctness, relevant system properties.
- the representation of a life insurance contract: " may I replace the service S by S' ? "

This requires

a decent theoretical background
for the emerging
software- and system paradigms.

A fundamentally new approach is necessary !

This is the challenge of "Modelling"

… just three aspects:

1st aspect: diverging runs

conventional: A terminating run makes sense,
a diverging run is insane.

new: A diverging run makes sense,
a terminating run is insane.

operating system, embedded systems, SOA

gains some attention already,
but not enough

2nd aspect: real world items

Software interacts with real world items.

sensors, lift cabin,

A model must cover all relevant aspects.

An algorithm may have uncountably many initial states, hence,
"each real world item is symbolically represented"
is too simple.

! separate algorithms from programs!

3rd aspect: many-agent-systems

… can not be conceived as

generalizations of one-agent-systems.

What problem solves a mutual exclusion algorithm?

Interaction of agents is

- a constituting aspect of systems,
- a fundamental issue of any new theory of computation.

more aspects of the wish list

Theory should

- comply with the basic laws of physics
  (Turing machines don't) ;

- characterize the border line between
  "the implementable" and "the non-implementable" ;

- provide robust links to (or include parts of)
  the psychology of design principles,
  interface design etc.

state of the art

In science, sometimes theory is before practice
(e.g. modern physics)
and sometimes behind.

In software it was always behind
(except in the early 1950ies, and for OO).
Now it is far behind.

A good starting point: SOA

Voices from software industry about SOA:

"*THE* most relevant emerging paradigm"

"A substantial change of view
as it happens at most once each decade"

"The next fundamental software revolution after OO"

"much more than just an other type of software!"

Resume

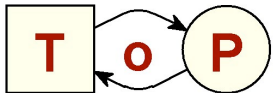
There is no comprehensive
Theory of Software.

It is time to strive for one!

ECSS Zürich, Sept. 10, 2008

In Praise of a Theoretical Basis for New Software Paradigms

many thanks *W. Reisig*



Theory of Programming

Humboldt-Universität zu Berlin

Prof. Dr. W. Reisig