



Mathematical Foundations of Software Engineering: a roadmap



TOM MAIBAUM
(King's College London)
tom@maibaum.org



Theory and software engineering



- > Why do we need **mathematical foundations**?
 - * because software engineering **is** engineering!
- > Where do we need **mathematical foundations**?
 - * to support all activities related to SW construction and use.
- > What kind of **mathematical foundations** do we need?
 - * appropriate for design, analysis and use of SW artifacts.
- > How will we develop these **mathematical foundations**?
 - * by a lot of hard work and **by focusing on engineering design!**



What is engineering?



- > According to Rogers, “engineering refers to the practice of organising the design and construction of any artifice which transforms the physical world around us to meet some recognised need”.

- > Hence, **the very nature of engineering is informed by the**

observational vs theoretical dichotomy

- > **the physical (observable) world**

vs

- > **the language/medium of design (and the expression of ‘recognised need’).**

7-9 June 2000

ICSE/roadmap/tsem

3



What is engineering?



- > According to Vincenti (and in accordance with the latter part of the above statement by Rogers), the day to day activities of engineers consist of *normal design*, as comprising “**the improvement of the accepted tradition or its application under ‘new or more stringent conditions’**”.

‘COOKBOOK METHODS’!

- > He goes on to say: “**The engineer engaged in such design knows at the outset how the device in question works, what are its customary features, and that, if properly designed along such lines, it has a good likelihood of accomplishing the desired task.**”

7-9 June 2000

ICSE/roadmap/tsem

4



What is engineering?



- > Jackson sings the praises of this concept of 'normal design', although he does not use this phrase himself:
 - * "An engineering handbook is not a compendium of fundamental principles; but it does contain a **corpus of rules and procedures** by which it has been found that these principles can be most easily and effectively applied to the particular design tasks established in the field. **The outline design is already given, determined by the established needs and products.**" ...
 - * "The methods of value are micro-methods, closely tailored to the tasks of developing particular well-understood parts of particular well-understood products."

7-9 June 2000

ICSE/roadmap/tsem

5



What is software engineering?



- > An implied view of engineering design is that engineers **normally design devices as opposed to systems** [Vincenti].
 - * A *device* (in this sense) is an entity whose **design principles are well defined, well structured and subject to normal design principles.**
 - * A *system* (in this sense) is an entity that **lacks some important characteristics making normal design possible (and necessitating radical design).**
- > Systems become devices when their design attains the status of being normal, i.e.,
 - the level of creativity required in their design becomes one of systematic choice, based on well defined analyses, in the context of standard definitions and criteria developed and agreed by the relevant engineers.**

7-9 June 2000

ICSE/roadmap/tsem

6



What is software engineering?

- > If we begin to think about the underlying activities that need to be supported by SW development environments, we should see:
- * *elucidation*, finding a **requirements specification**;
 - * *illumination*, finding a **constructive architectural specification** satisfying the requirements;
 - * *systematic observation*, **validating** through **experiments**, by either explaining the behaviours or predicting them; and
 - * *calculation*, using the underlying **formalism**, formal derivation of validation and verification tests, of refinements, etc.

The parallel between these activities and what is informally called 'the scientific method' is too strong to be considered merely a coincidence.

7-9 June 2000

ICSE/roadmap/tsem

7




The Two Level Theory of the Language of Science

- > The idea of using this method as a sound and systematic foundation for the activity of constructing software systems is further strengthened and well founded when we start thinking of the activity of **testing** (or **validation**).
- > The fact that a positive result of a test does not confirm the correctness of a program, while a negative one refutes it, is obviously a late restatement of the **hypothetico-deductive method** in its simple version – stated by Newton and refined by Carnap, Hempel and others – which is at the heart of the method of Natural Science and Engineering.
- > The conceptual architecture of a framework should be based principally on the epistemological model of the method of Natural Science known as the *Statement View*, or, alternatively, *the two level theory of the language of science*.

7-9 June 2000

ICSE/roadmap/tsem

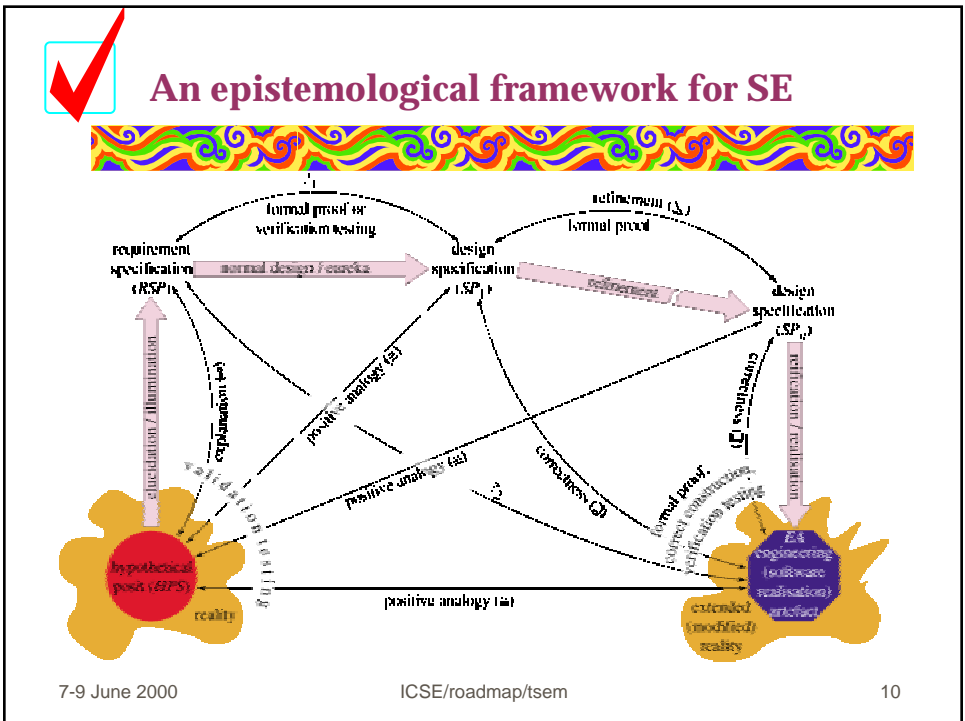
8



The status of the Statement View

- We have been able to use modern methods and techniques in logic to undermine many longstanding criticisms of the Statement View:
 - * new tools in logic
 - * a new science (CS!), invented independently of traditional sciences, providing a rich source of counterexamples to the criticisms
- Further, whatever the status of the Statement View is wrt science, we believe that we can apply related ideas to a **normative theory** (and **praxis!**) of (SW) **engineering**.
 - * scientists may or may not follow the rules as rationalised by epistemologists
 - * **engineers, by their very nature, must follow predefined methods and standards so as to earn and retain the right to be called engineers!**

7-9 June 2000
ICSE/roadmap/tsem
9





The (nondeterministic!) roadmap



- Some observations which could be discussed:
 - * SE ignores the principles of engineering design and almost always adopts radical design methods, to its detriment.
 - * SE curricula tend not to teach engineering, i.e., they ignore the concept of devices and their principles.
 - * Engineering methods are (completely?) domain specific.
 - * The science and mathematics of engineering is often quite different from real science and mathematics. (Of course they are linked!)
 - * Engineers use analyses which are not formally related (they may even be 'inconsistent'!), but are related by the method.
 - * Engineers know, because of the method, where the critical analyses may be useful. They know how to avoid working at the same level of detail in all parts of the design.

7-9 June 2000

ICSE/roadmap/tsem

11



The (nondeterministic!) roadmap



- Vincenti organises engineering knowledge in terms of:
 - * fundamental design concepts
 - * criteria and specifications
 - * theoretical tools
 - * quantitative data
 - * practical considerations
 - * design instrumentalities
- and we might do the same with our discussion.

7-9 June 2000

ICSE/roadmap/tsem

12



Fundamental design concepts



- > We need many and varied **in-the-small theories** to support design.
- > Examples might include:
 - * representing **behaviour** and being able to analyse it
 - many different **kinds of behaviour**
 - many different **notations**
 - standard **analyses**
 - * defining **'ilities'** and incorporating them into (normal) design
 - * systematising the idea of **normal configuration**, as in **SW architecture, problem frames, patterns**

7-9 June 2000

ICSE/roadmap/tsem

13



Criteria and specifications



- > (To achieve high ratings via CMM,) we need to systematise design knowledge.
- > This can only be done by using a **domain focus**.
- > So systematising design knowledge involves at least:
 - * systematising **domain** knowledge (very difficult!)
 - * finding the specification and refinement patterns/architectures required to encapsulate design choices; define **'cookbooks', product line architectures, code generation by transformations**

7-9 June 2000

ICSE/roadmap/tsem

14



Theoretical tools



IMPORTANT MESSAGE!

- > The theories and maths used by scientists and engineers are not the same!
- > We need in SE specialised theories and analysis methods, which may often be domain specific, i.e., **particular to a specific design method**.
- > There are a whole host of specific topics which require attention, amongst which are:

7-9 June 2000

ICSE/roadmap/tsem

15



Theoretical tools



- > **modularity principles** (see my work with Veloso, Fiadeiro, Dimitrakos and Fiadeiro's work):
 - * not presentation modularity (as in Z and algebraic specification)
 - * modularity of programs is not the same as that of specifications and designs
 - * relationships between these different notions of modularity
- > **behavioural principles** (again see my work with Fiadeiro, much work by many on process algebras, temporal logic, statecharts, etc)
 - * the very basis of SW design
 - * (analogous to differential equations in other engineering domains?)
 - * **scalability** (and relation to modularity)
 - * **feature interaction** and **emergent properties**

7-9 June 2000

ICSE/roadmap/tsem

16



Theoretical tools



- > **specialised analysis tools**
 - * normal design incorporates standardised analyses (e.g., type checking)
 - * tools must support them directly (see D Jackson's work at MIT)
- > **abstract interpretation**
 - * many properties that we are interested in can be (only) determined by abstracting away from the real details
 - * what are the useful properties? Are they related to 'ilities'?
- > **decision procedures** for interesting properties (see the work on timed automata by Henzinger, Alur, etc)

7-9 June 2000

ICSE/roadmap/tsem

17



Theoretical tools



- > **systematisation of testing**
 - * related to **experiment** in science
 - * ideas from the epistemology of the sciences may be useful (see my work with Haeberer)
 - * e.g., the recent work of Haeberer and Cengarle, based on ideas from the philosopher Glymour, prove the point
 - * **simulation** and **animation** are important related topics (see Magee's work at Imperial College)
- > **(design) theories of 'ilities'**
 - * least promising starting position!

7-9 June 2000

ICSE/roadmap/tsem

18



Quantitative data, Practical considerations, Design instrumentalities



- > We need **quantitative analyses** of our tools, processes and building blocks, e.g., Knuth on algorithms.
- > (In engineering) theory only aids the organisation of **practical knowledge**. Standards and organisational principles are needed ... Professionalisation is needed ...
- > Hmm ... We require much more experience to begin addressing **design instrumentalities** seriously ... This is off my 'map'!