It has been identified that there is a small risk of a brake problem in certain Scientific models. While customer safety has always been the primary focus for Scientific, we never intended the handbook to be a description of how the brakes should behave but rather as a theory or prediction of their behaviour. In keeping with this, we encourage drivers to test their brakes under the most extreme circumstances. We understand that this might cause nightmares among the infirm, but Scientific would greatly appreciate receiving details of the resulting information. At no time shall we recall cars that are unsafe, but be assured that, in the fullness of time, we shall issue new handbooks that provide a better theory of how the breaks will perform.
SPECIFICATION and ARTEFACT
Outline

I. Specifying Artefacts
II. The Varieties of Specification
III. Definitions
IV. Physical Artefacts
V. Abstract Artefacts
VI. Theory and Specification
I. SPECIFYING ARTEFACTS
Why Specification?

- The specification and construction of computational artefacts is a central activity of Computer Science.
- This stretches beyond software engineering. As an activity it is to be found throughout the discipline. Indeed, it is partly constitutive of it.
- Unpacking the nature of specification will provide some conceptual insight into the nature of CS.
Computational Artefacts

Much of computer science involves the design, specification, analysis and implementation of computational artefacts

1. Programs
2. Algorithms
3. Programming Languages
4. Databases
5. Compilers and Interpreters
6. Operating Systems
7. Machines
8. Type Inference Systems
9. Theorem provers
10. Systems on a chip
11. Natural Language systems of various kinds
12. Speech recognition
13. Visual processing systems
14. Computer games...
The Role of Specification

- To describe the artefact not how to build it.
- In software development, a functional specification describes the behaviour of a computer program or larger software system.
- Knowing what versus knowing how (to Build).
- More information about implementation provides more knowing how.
- Different levels of both for users and designers.
II. VARIETIES OF SPECIFICATION
Primary coolers and cooling water system.

Sample temperature shall be reduced using full counterflow sample coolers to achieve a temperature within 5°F of the temperature of the primary cooling water. All coolers are furnished as an integral part of the conditioning rack. Certified performance data on all coolers shall be provided at design conditions.

All coolers shall be of the coil in shell design, with counterflow of sample and cooling water. Coolers shall be Sentry type TLF, FLF or FXF, with a single flange and gasket shell design. For maintenance, the sample coolers shall have removable shells. All coolers shall have 316 stainless steel tubing and 304 stainless steel shells.

All coolers shall be arranged side by side in the rack, and shall be connected to an inlet and an outlet cooling water header. Coolers shall be located for easy access to the throttling valves and for simplified maintenance and replacement, as necessary. In addition, cooling water piping shall not disturb the sample piping. Sample piping shall not interfere with the removal and replacement of the cooler coils and/or shells.

All coolers shall be equipped with outlet globe valves for throttling the flow of cooling water, and a 3-way ball valve on the inlet. Cooling water headers shall be of carbon steel and of the appropriate size to handle the required flow rate to the coolers. The inlet cooling water header shall be equipped with a relief valve sized to adequately relieve excess cooling water pressure. Outlet cooling water headers shall include a vent for air removal, as required.
Arithmetic

1. \(G(x) \equiv 3x^3 + 2x^2 + 7x + 9\)

2. \(P[x, y] \equiv \exists z. y = x + z^2\)

3. \(F(0) \equiv 12\)
   \(F(n+1) \equiv 3 \times F(n) + 8\)
Z-Notation

Birthday book

- known: NAME
- birthday: NAME → DATE

Known : dom birthday
Class Diagrams

Class diagrams typically describe the different entities of a system as classes and the relation between these.
Architectural Patterns

The object adapter pattern expressed in UML. The adapter hides the adaptee's interface from the client.
An ADL Example (in ACME)

System simple_cs = {
    Component client = {Port send-request}
    Component server = {Port receive-request}
    Connector rpc = {Roles {caller, callee}}
    Attachments : {client.send-request to rpc.caller;
                   server.receive-request to rpc.callee}
}
\( \Pi - \text{Calculus} \)

The core of the handover protocol for the GSM Public Land Mobile Network. One component is a \textit{Mobile Station (MS)}, mounted in a car.
Some Questions

- Logically, what are specifications?
- Do they all function the same way?
- What is the nature of the relationship between specification and artefact? Is it always the same?
- Is their role similar to that of definitions in mathematics?
- What is the difference between programs and specifications?
- Do they need to be formally expressed?
III. DEFINITIONS

SEP
Specifications as Definitions

- Specifications seem to be definitions.
- But what kind?
- How do they differ from mathematical definitions?
Forms of Definition

- Real and Nominal
- Dictionary and Ostensive
- Stipulative
- Descriptive
- Expliciative
Real and Nominal

• The nominal essence of the name ‘gold’, *is that complex idea the word gold stands for*, let it be, for instance, *a body yellow, of a certain weight, malleable, fusible, and fixed*. (Locke)

• *The real essence of gold is the constitution of the insensible parts of that body, on which those qualities [mentioned in the nominal essence] and all other properties of gold depend* (III.vi.2).

• Presumably, the chemist aims at a real definition, whereas the lexicographer aims at nominal one.
Dictionary and Ostensive

- Dictionaries aim to provide definitions that contain sufficient information to impart a semantic understanding of the term.
- We can teach the term ‘meter’, by giving the ostensive definition “This stick is one meter long” while pointing at a meter stick.
Descriptive Definitions

• Descriptive definitions spell out meaning, but they also aim to be adequate to existing usage.

• Three grades of descriptive adequacy
  i.  **Extensional**: no actual counterexamples to it;
  ii.  **Intensional**: no possible counterexamples to it;
  iii.  **Sense**: endows the defined term with the right sense.

• *Water is H₂O* is intensionally adequate – presumably, the identity of water and H₂O is necessary.

• Assuming the Kripke-Putnam view about the rigidity of natural-kind terms, it is also extensionally adequate.

• It is not sense-adequate, for the sense of ‘water’ is not at all the same as that of ‘H₂O’.
Sometimes a definition is offered as, what Rudolf Carnap called, an *explication*.

An *explication* aims to respect some central uses of a term but is stipulative on others. The explication may be offered as an absolute improvement of an existing, imperfect concept.

A simple illustration of explication is provided by the definition of ordered pair in set theory. *Viewed as an* explication, this definition does not purport to capture all aspects of the antecedent uses of ‘ordered pair’ in mathematics but only the essential uses.
Stipulative Definitions

- A *stipulative definition* is a type of definition in which a new (or even an existing term) is given a specific meaning in a given context.
- They introduce new things i.e., relations, objects, functions, properties,...
- For example, *grue* was *stipulated* to be a property of an object that makes it appear green if observed before some future time $t$, and blue if observed afterward.
- They involve no commitment that the assigned meaning agrees with any prior use.
- A stipulative definition cannot be correct or incorrect.
Relations, Functions and Objects

- We take the view that stipulative definitions introduce new:
  1. Relations
  2. Functions
  3. Objects

- We shall develop a more precise logical account in the lecture on the logic of definitions.
Mathematical Definitions

- Not all simple stipulative definitions.
- Definitions evolve e.g. the history of the notion of prime number.
- May have to take into account past and informal usage.
- Current topic of investigation in the philosophy of mathematics aimed at mathematical practice.
- The right definition may take time: the role of explanation in mathematics.
The Specification of Computational Artefacts

• In contrast, specifications do seem to be stipulative definitions.
• They do not have such a sense of history.
• They are local to the system under definition
• However, they point beyond the definitions themselves:
  i. They are aimed at the construction of artefacts.
  ii. They tell us what to build.
  iii. They tell us what we have.
Correctness and Malfunction

- Specifications give substance to the notions of correctness and malfunction.
- The conceptual role of specification is not tied to the actual process of construction. We may arrive at the artefact through various routes.
- The conceptual role of specification is to provide a criterion of correctness.
- If it says beans on the can, we expect it to contain beans not bananas. How they got there may explain how any mistake was made, but not why it is a mistake.
Artefacts and Correctness

- Artefacts maybe be physical or abstract e.g. computers versus type inference systems.
- Physical ones are subject to the laws of physics.
- Abstract ones are subject to mathematical analysis.
- This generates two different notions of correctness / malfunction.
IV. ABSTRACT ARTEFACTS
Correctness

* S is correct relative to S (written R < S) 
  iff
  I. Dom(R) ⊆ Dom(S)
  II. S(x, y) → R(x, y)
  Where S = S ↾ Dom(R)

- Where S is functional II reduces to: R(x, S(x)).
- It states that S correctly implements R
- R provides the correctness criteria for S
- Proof of correctness is a mathematical affair.
Programs as Abstract Objects

• Programs are the occupants of programming languages. Programs, as semantic things, are not strings of symbols or physical devices but abstract objects.

• Given some semantic account of programs where they are taken as relations between states of the underlying abstract machine, i.e., via their semantics each generates a relation/function.

• The correctness of programs is a relationship between abstract objects.

• Their properties are explored by mathematical means.
Programs as Specifications or Artefacts

Via their semantics, programs may be viewed as specification or artefact i.e., via their semantics each generates a relation/function

\[ x := 0; \ y := 1; \]
\[ \text{while } x < n \text{ do } (x := x + 1; \ y := x * y) \]
\[ \text{Fac}(0) = 1 \]
\[ \text{Fac}(n+1) = n+1 * \text{Fac}(n) \]
Correctness Jurisdiction

- The difference between program and specification is not attached to any characteristic of the syntax of a programming or specification language but to their semantic account.
- Something is a specification when deemed to have correctness jurisdiction over its artefact.
- Levels of specification reflect levels of implementation detail versus levels of abstraction.

\[ S_1 < S_2 < S_3 < S_4 \ldots \ldots M. \]

\( S_i \) provides the correctness criterion for \( S_{i+1} \)
Implementable Definitions

- Do specifications differ from mathematical definitions in terms of their implementable status?
- Should we insist that they are computable?
- More of this in the lecture on the logic of definitions.
V. PHYSICAL ARTEFACTS
Physical Artefacts

- Machines
- Physical memory
- Chips
- Physical manifestations of programs
- CPU
- GPU
- ..
- What does correctness mean for physical artefacts?
Machine Specification

- A computer store is to have named locations that hold numerical values.
- There has to be some means of obtaining the content of any given named location – a *Lookup operation*.
- There has to be a means of changing its contents – an *Update operation*.
- When the location’s contents are changed, the contents of another location should not be changed.
Abstract Machine in Z notation

**Update**: $\text{States} \times \text{Location} \times \text{Values} \rightarrow \text{States}$

**Lookup**: $\text{States} \times \text{Location} \rightarrow \text{Values}$

\[
\text{Lookup}(\text{Update}(s, x, v), y) = \text{Lookup}(s, y) \quad \text{where } x \neq y
\]

\[
\text{Lookup}(\text{Update}(s, x, v), x) = v
\]
The Correctness of Physical Artefacts

- In itself it is just a **definition** of an abstract machine.
- But it may also be taken a specification of a physical one.
- Correctness is an empirical claim about the physical machine i.e., it meets the specification given by the abstract one, i.e., correctness of the physical device is tested by empirical means.
- If when the machine is updated with input 3 into location \( z \), it puts 7 in location \( u \), then the physical machine has **malfunctioned**.
- But it is the **artefact** not the specification that is being tested. If the above happens the physical machine is to blame. It must be modified.
The Abstract and the Physical

An abstract machine may be considered as a

1. A specification of the physical machine (present view)
2. A theory of how the physical machine will behave (Fetzer)

The difference between 1 and 2 goes to the heart of the difference between scientific methodology and engineering design
The difference involved here is precisely that between pure mathematics and applied mathematics. (F 1988)

....the crucial problem confronting program verification is establishing the truth of claims of form (C) above, which might be done in two possible ways. The first is to interpret rules and axioms of form (C) as definitional truths concerning the abstract machine thereby defined. The other is to interpret rules and axioms of form (C) as empirical claims concerning the possible behaviour of the target machine thereby described... (F 1988)
Colburn (2000)

- DP is actually a **prediction** of what will occur when statement S is ultimately executed. ..... 
- *It is a static prediction of its later dynamic execution.*
VI. THEORY and SPECIFICATION
A Rough Guide to Engineering Design

- Engineers deal with *constructed artifacts* that are designed and built to meet some design *specification*.
- Specifications are not intended to be explanatory but normative.
- They determine the constructed *artifact*.
- Of course, the latter may not measure up to its abstract specification. If so, it needs to be *reconstructed*. 
A Rough Guide to Theory Construction

- Given an artifact, a scientist formulates a *theory* about it. Such theories are often abstract and mathematically expressed.
- The predictions of the theory provide the mechanism whereby the *theory maybe tested empirically*.
- If the predictions turn out to be false, the *theory may have to be revised*. Subsequently, theory construction cycles through alternate stages of theory articulation and empirical verification.
Theories of Artefacts

- I find an artefact in field. It has a row of boxes with numbers in them. It has a keyboard attached with numbers on.
- Above each box is a button. I press one and a number appears. I press a number on the keyboard and press the button above a box and the number in the box is replaced.
- I postulate that it is a simple store machine. This is now a theory of the artefact.
From Theory to Specification

- I decide to use it as a store machine.
- At this point my *theory* becomes a *specification* for future use.
- I adopt a different intentional stance towards it.
- This stance will remain in place until I doubt the original theory.
Specifications versus Theories

- Theories are intended to be descriptive or explanatory. They are evidenced by natural artefacts. The world is not correct or incorrect.
- While specifications have a similar logical form to scientific theories they have an entirely different function: they are not intended to be descriptive or explanatory.
- A specification is not something that is to be tested; it is not correct or incorrect. They fix what a correct artefact is.
- It is the artefact that is under test. When things go wrong, it is the artefact that is to blame.
Tests and Experiments

- Artefacts are tested
- Theories are tested
- Specifications are not correct or incorrect
- The world is not correct or incorrect
All is Specification and Artefact

- In the chain $S$-$E$-$M$,
  \[
  \text{Spec} \leftarrow \text{Code-Machine}
  \]
  $S$ is the specification and the chunk $C$-$M$ is the physical artefact.
- It is the artefact ($C$-$M$) that is the focus of any empirical testing.
- Of course, we may chose to assume the veracity of the physical machine and focus on $C$. 
It has been identified that there is a small risk of a brake pedal problem in certain Scientific models. While customer safety has always been the primary focus for Conquest, we never intended the handbook to be a theory of how the brakes should behave but rather as a specification of their behaviour. In keeping with this, we encourage drivers to test their brakes under the most extreme circumstances. We understand that this might cause nightmares among the infirm, but Scientific would greatly appreciate receiving details of the resulting information. While we shall not issue new handbooks that change the specification, we shall, in the fullness of time, recall all cars that are deemed unsafe.
Next Two Lectures

II. The Logic of Definition: Computable Models

III. Semantics and Ontology: Programming Languages as Mathematical Objects