The World
and
The Machine

Michael Jackson
MAJ Consulting Ltd and AT&T Bell Laboratories
ICSE-17 Seattle 28th April 1995
Ways of Looking at Software

- ‘Programming should be literate’
- ‘... they regarded my programs as logical poems ...’
- ‘The goal of any system is organisational change’
- ‘Software development is engineering’
  - Because we make machines to serve useful purposes in the world
  - The problem is in the World
  - The Machine is the solution
WHAT and HOW

- WHAT does an automobile do?

- It carries people and their baggage, travelling over roads where its driver directs it to go

- WHAT is in the world, HOW is in the machine
The Machine, the Model, and the World

- **Formal Methods** concern the left arrow
- We have no theory for the right arrow

Brian Cantwell Smith; *The Limits of Correctness*
Talking about the World and the Machine

- To develop software we must talk both about the World and about the Machine

- But it's hard to maintain the right balance between these two universes of discourse
  - The relationship between them is varied and often subtle
  - Often we have personal preferences to exploit or resist
Three Topics and a Button

- 4 Facets of the Relationship
- 4 Kinds of Denial of the World
- 4 Principles for Accepting the World
- a Button:
4 Facets of the Relationship

- Modelling:
  the Machine as a model of the World

- Interface:
  what the Machine shares with the World

- Engineering:
  how the Machine changes the World

- Problem:
  the structure the Machine must have to fit the problem in the World
Modelling a Reality

- ‘An SADT system description is called a “model” …’
- R L Ackoff (Scientific Method, 1962):
  - Iconic models — pictures, 3-D representations, eg a child’s model farm
  - Analytic models — manipulable formal descriptions, eg differential equations forming an economic model
  - Analogic models — an analogous reality, eg an electrical network modelling the flow of water in pipes
- Software models are analogic: eg, a database, an assemblage of objects, a process network
The Machine As a Model of the World

Authors, Novels, PublishedBy

$A(x) = 'x$ is an Author'

$N(x) = 'x$ is a Novel'

$P(x,y) = 'x$ is PublishedBy $y'$

$A(x) = x$ is A-record

$N(x) = x$ is N-Record

$P(x,y) = 'x$ PointsTo $y'$

$\forall x : N(x) \cdot \\
\exists y : A(y) \cdot \\
P(x,y)$
Modelling and $\bigcirc$

- A data model fragment:

- Three sets of descriptions:
Non-Modelling and

- Both the World and the Machine have properties that are private and not shared

- Record Deletion
- Normalisation
- Record Sequencing
- Null Field Values

- Multiple Authors
- Anonymous Works
- Multiple Pseudonyms
- Linked Novels
The Machine – World Interface

- Shared phenomena: events, other shared individuals, facts visible in both domains
- No communication without sharing:

\[ \text{transmission without sharing?} \]

is ‘really’ ...

![Diagram showing the flow between two individuals through Royal Mail with shared events 'post letter' and 'deliver letter']
Shared Phenomena

- **Operator's Panel Domain**
  - Levers
  - FlipUp events
  - FlipDown events

- **Circuits and Contacts Domain**
  - Switches
  - TurnOff events
  - TurnOn events

- **Shared phenomena:**
  - Levers
  - FlipUp events
  - FlipDown events
  - Switches
  - TurnOff events
  - TurnOn events

- **Private phenomena:**
  - Links
  - LinkedBy
    - (x:Lever, y:Link)
  - Contacts
  - LocatedOn
    - (x:Contact, y:Switch)
The shared phenomena are in the (small) intersection between two sets of phenomena:
Modelling and Shared Phenomena

- *Sharing phenomena* and *modelling* are different relationships between the Machine and the World
  - Shared phenomena \(\rightarrow\) modelling
    - Any description that is true of the shared phenomena is a shared description
  - But ...
  - ... \(\rightarrow\) (modelling \(\rightarrow\) shared phenomena)
    - The database shares no phenomena with the reality it models
Engineering: Requirements, Specifications, and Programs

- The purpose of the Machine is to change the World: this is the requirement.

- The required changes are expressible entirely in terms of phenomena of the World ...

- ... but not usually entirely in terms of phenomena shared with the Machine.

- The final engineering product:
  - Machine behaving according to the program ...
  - ... thus satisfying the specification and ...
  - ... thus ensuring achievement of the requirement.
Requirements, Specifications, Programs

- PW Phenomena of the World
- PW ∩ PM Shared Phenomena
- PM Phenomena of the Machine

- A specification is also a requirement
- A specification is also a program
Engineering and \( \mathcal{O} \)

- Programs can satisfy specifications only by virtue of \textit{properties of the machine} (p/l semantics)
- Specifications can satisfy requirements only by virtue of \textit{properties of the world}
- The engineering is in determining, describing and exploiting the properties of the world
A Little Engineering Example

- \( R: \) on_runway \( \leftrightarrow \) can_reverse
- \( D1: \) wheel_pulses \( \leftrightarrow \) wheels_turning
  \( D2: \) wheels_turning \( \leftrightarrow \) on_runway
- \( S: \) can_reverse \( \leftrightarrow \) wheel_pulses
- We have: \( S, D1, D2 \vdash R \) — is it enough?
The Problem Facet of the Relationship

- Solution structure should reflect problem structure
  - There’s less need for invention
  - It’s easier to validate the solution
- Traditional solution structures are often hierarchical and homogeneous ...
  - Procedure hierarchies, class hierarchies, layered abstract machines, process/dataflow structures
- ... but the World rarely exhibits such structures
A Simple Editing Tool

- Three requirements:
  - *Editing* allows users to create and edit texts
  - *GUI* provides convenient and efficient operation
  - *Revision History* provides progress reporting by users and texts
- The requirements are related by conjunction:
  - *Editing* \( \land \) *GUI* \( \land \) *Revision History*
- The requirements share phenomena
Two Requirements Sharing Phenomena

Editing

- insert
- find_word

Edit ∩ Histy

- document
- open_to_update
- save_document

Revision Histy

- user
- log_on
- delete_document
Problem Structures

- Problems are usually structured as subproblems that are:
  - heterogeneous
  - related by superimposition
  - pinned together at shared phenomena

- The appropriate metaphor is ...
  - ... not assemblies and sub-assemblies
  - ... but CYMK separations in colour printing
The World and Us (1)

“The world is too much with us ...”
— William Wordsworth
4 Kinds of Denial

- How we may deny our involvement
  - Denial by Prior Knowledge
  - Denial by Hacking
  - Denial by Abstraction
  - Denial by Vagueness
Denial by Prior Knowledge

“We don’t need a requirements capture phase. The problem is already well-defined; our task is merely to solve it.”

- Automobile designers don’t have a requirements capture phase ...
  - The car shall be able to travel over snowdrifts and under water
  - The car shall be able to lift a load of 5 tons
  - The car shall accommodate 10 passengers each of weight up to 500 pounds
- ... it would be called ‘Rethinking the Motor-car’
Denial by Prior Knowledge

- Legitimate only in applications that are both *specialised* and *standardised*
- Both bridge-design and automobile design are *specialised*
- But only automobile design is *standardised* (human beings, roads and baggage don’t vary much)
- Bridge design is not *standardised* (each location has unique characteristics)
Denial by Hacking

- Computers are beautiful and fascinating

  “... Miss Byron, young as she was, understood its working and saw the great beauty of the invention.”
  Mrs De Morgan, on Ada’s visit to Babbage, 1828

- Applications are often much less interesting

  “I came into this job to work with computers, not to be an amateur stockbroker.”
  Member of failed development team, 1993

- The Machine is the developers’ own creation; the World is not
The Royal Albert Bridge, Saltash

I K Brunel, Engineer, 1849
Looking at the Problem Context

- Which is the World? Which is the Machine?
- Which do you describe at the next level of DFDs?
Denial by Abstraction

“We come now to the decisive step of mathematical abstraction: we forget what the symbols stand for.”

Hermann Weyl, quoted by Abelson & Sussman

- Abstraction is a valuable intellectual tool ...
- ... but it must not be a rule of life for software developers
- Too much abstraction blinds you to the nature of many problems
Doing Justice to the Problem

“One tribe always tells the truth and the other always lies. A traveller meets two men, and asks the first: ‘Are you a truth teller?’. The reply is ‘Goom’. The second says: ‘He said Yes, but he is lying’.

Martin Gardner, 2nd Book of Puzzles

- Abstract answer:
  “The reply must always be Yes; so the second man is a truth-teller, and the first is a liar”

- Lucy Jonelis’ answer:
  “The first man clearly can’t speak English: ‘Goom’ must mean ‘What?’ or ‘Welcome to our land’. So the second man is a liar, and the first is a truth-teller.”
The Package Router

- Incoming Packages
- Reading Station
- Sensors at Top and Bottom of each Pipe
- Two-position Switch at each node
- Bins
Denial by Vagueness

- **Central technique:**
  - Describe the Machine, but imply that you’re describing the World

- **Prerequisite:**
  - Avoid saying explicitly what is being described

- **Facilitators:**
  - The modelling relationship (the same description is true of both)
  - The shared phenomena at the interface (two sides of the same penny, isn’t it?)
The System and the Real World

“... the Z approach is to construct a specification document which consists of a judicious mix of informal prose with precise mathematical statements. ... the informal text can be consulted to find out what aspects of the real world are being described..... The formal text in the other hand provides the precise definition of the system and hence can be used to resolve any ambiguities present in the informal text.”

- Machine = system?  World = real world?
- Which is being described?
Talking About the World: 4 Principles

von Neumann’s principle
  • Knowing what you’re talking about

The principle of reductionism
  • Finding the solid ground

The Shanley principle
  • Recognising versatility

Montaigne’s principle
  • Minding your language
von Neumann’s Principle

“There is no point in using exact methods where there is no clarity in the concepts and issues to which they are to be applied.”

von Neumann & Morgenstern: Theory of Games

- Designations
  - Mother(x, y) \approx ‘x is the genetic mother of y’
  - Formal term \approx recognition rule
- Anticipate interventions of the form: “It all depends on what you mean by mother”
Aligning a Description

- Designated terms and phenomena are like triangulation points on the map and on the ground.
The Principle of Reductionism

- In any informal world many terms — often nouns in English — are obviously important ...
  - in telephony: calls
  - in a meeting-scheduling system: meetings
  - in an airline system: flights
- ... but difficult or even impossible to designate
- They must be reduced to elementary designated phenomena — often events
Reducing Domain Concepts

- The rebuilt defined terms are not the original informal terms
- Definition is not designation
The Shanley Principle

“In civil engineering design it is presently a mandatory concept known as the Shanley Design Criterion to collect several functions into one part.”

Pierre Arnoul de Marneffe, cited by D Knuth, 1974

- 1940-1945 rockets had separate components for fuel tank, outer skin, body frame
- Saturn-B had a tubular body that was at once its fuel tank, outer skin, and body frame
- It may (or may not) be good to engineer Machines in this way, but the World is certainly like this!
- No class hierarchy, no strong typing!
Shanley and Many Descriptions

Editing Requirement:
Operation O requested on text T

Revision History Requirement
Operation O requested on text T by user U

GUI Requirements
Operation O requested by clicking button B

• One description is not enough
Montaigne’s Principle

“The greater part of this world’s troubles are due to questions of grammar.”

- Demanded for some Government contracts:
  “Absolute tense ‘shall’: a binding, measurable requirement ....
  “Future tense ‘will’: a reference to the future, ... not under control of the system being specified.
  “Present tense: for all other verbs ....”

- The distinction is not of tenses, but of moods
  - Optative: desired in the World
  - Indicative: true regardless of the Machine
Indicative and Optative

- Natural language distinctions are impractical:
  - “I shall drown, no-one will save me!”
  - “I will drown, no-one shall save me!”
- Mood of a sentence in development changes with its context:
  - In handling the *Revision History* requirement, the *Editing* requirement should be treated as satisfied — not optative but indicative
- So indicative and optative sentences should be kept apart in separate descriptions
Three Topics and a Button

- 4 Facets of the Relationship
  - The Machine as a *model* of the World
  - The interface of *shared phenomena*
  - *Engineering* the World and the Machine
  - Problem and solution *structures*

- 4 Kinds of Denial of the World
- 4 Principles for Accepting the World
The World and Us (2)

“I accept the universe”
— Margaret Fuller

“By Gad! she’d better!”
— Thomas Carlyle