Formal Specification F28FS, Lecture 1

Jamie Gabbay

January 27, 2014

About me

My name is Murdoch James Gabbay—everybody calls me Jamie. I got a PhD in Cambridge in 2001. Since then I have been a professional researcher—that is, I make my living by discovering and proving mathematical certainties.

My job consists of 50% academic research (into logic and lambda-calculus), 50% teaching (Formal Specification and Technology in Society), and 50% administration.

Lectures are Mondays 12:15-13:15 in EM3.36 and 15:15-16:15 in DB1.13, and Tuesdays 16:15-17:15 in EM1.82. There is a lab Tuesdays 17:15-18:15 in EM2.50.

The course webpage is `http://www.macs.hw.ac.uk/~gabbay/F28FS/`. Keep an eye on it, please.

At the end of the semester I will set exams, which I will mark.

About you

You are about twenty years old (unless you are a mature student). You are in your second year at university.

You need to turn up to all lectures, do all labs, and understand this course as it is delivered. No exceptions.

You need to complete and understand all exercise sheets, and keep up to speed with the lectures.

About you

I'm not going to teach you.

You're going to learn.

You aren't at school. You have taken control of your own learning.

It is up to you to take advantage of exercise sheets, the internet, books in the library, the forum, your colleagues, and your own intelligence—to learn.

My job is to offer you an opportunity to learn. Your job is to accept that opportunity.

My lectures will be clear—but the material is difficult. Your job is to understand it.

About the course

This is a course on formal specification as applied to programming.

The module descriptor (see course webpage) says that you should acquire the following ‘Personal Abilities’:

- Understand syntax and informal semantics of Z.
- Go from a real world example to a formal specification.
- Reason with the specification.
- Go from a specification to a discussion of what it models.
- SML: polymorphic types, recursive functions, higher order functions, pattern matching, structured types.
- Interpret a specification as operational requirements on a program.
- Refine a specification to a program.

Specification vs. Implementation

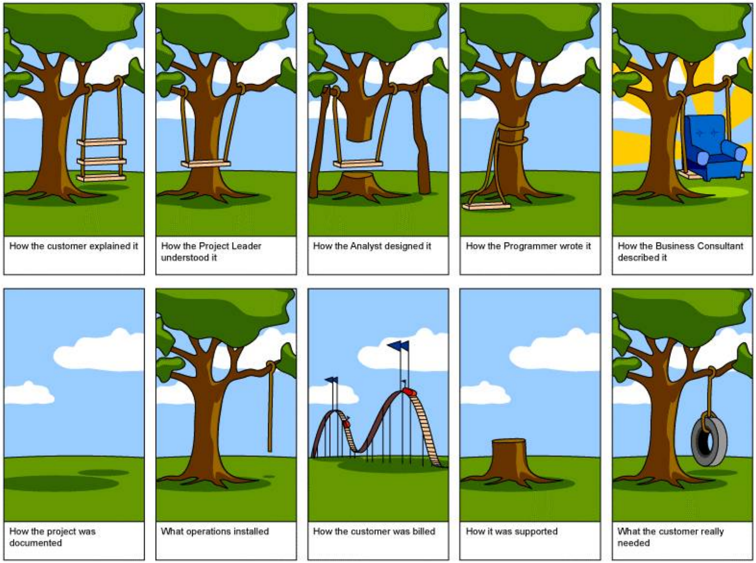
What is specification? How does it differ from implementation?

If I'm right, you come from a culture that identifies a better program with a faster or more efficient program.

Not always so. If the program is doing the wrong thing, then doing it faster won't help (it might even be worse).

So what does it mean to be doing the right thing, or the wrong thing, and how can we measure this?

Specification Is Challenging (SIC) (a swing)


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

SIC (the flag)

Suppose that we are in the USA and we want to specify:

"*Put a flag up outside your home on the 4th of July.*"

Looks pretty simple, right?

Let's think about this like programmers.

SIC (the flag)

Does that mean midnight to midnight, or just during the day?

Do you mean exactly midnight to midnight, at least midnight to midnight, or at most midnight to midnight — or dawn to dusk — or dawn to dawn?

What if you put it up and the flag falls down? Does that count? Are you authorised to put the flag up twice (e.g. if it fell down)?

SIC (the flag)

What does ‘your home’ mean?

What if you have two homes? What if your home is being redecorated and you’re in a hotel?

Do you put flags in all your homes, or just your one? Do you have to do it, or can your spouse do it for you?

SIC (the flag)

Suppose you have a flag up all year (americans are patriotic). Do you have to take it down so you can put it up?

SIC (the flag)

What is a flag?

What size? What if the colours are faded, or it's muddy?

Does it have to be somewhere people can see it?

Does it have to be an American flag, or will any flag do? A Confederate flag? A Chinese flag? A flag you made up yourself?

SIC (the flag)

What if you steal a flag? Does that count?

What if you have bought the house but you have not yet signed the completion. Who's responsible for the flag: the seller or the buyer?

SIC

You are a professional programmer instructed to “write a program to put up a flag outside the customer’s house on the 4th of July”. Do you have enough information? Who decides on the answers to all the questions above? Is that a good thing?

You have bought a software library. Documentation states “this procedure puts up a flag outside the customer’s house on the 4th of July”. Can you use this procedure without reading its code?

You are managing a software project. Legacy code states “this procedure puts a flag outside the customer’s house on the 4th of July”. You want to update this to account for timezones across the US. Do you have enough information to proceed?

The perils of specification

The problem of specification is hard, plagued by ambiguity, and inescapable.

In this course, we will learn to use the Z specification language.

We will use Z to specify the behaviour of some programs in ML (do you know ML?). We will then learn how to write ML implementations of Z specifications.

The Z programming language

- E. Currie “The essence of Z”. Buy it if you can (get it out the library if you can’t). Read it.
- J. Jacky “The way of Z”. Light reading.
- D. Lightfoot “Formal Specification Using Z”. More light reading.
- M. Spivey “Understanding Z”. A mere trifle.

The internet

I will set up a course webpage and tell you where to find it.

Remember to go to the forum (follow the links from my teaching webpage) and set up a user account.

I record my lectures. I will put .avi files on the internet for you to enjoy, again and again and again, as often as you like.

About logic (right; let's do some maths)

A proposition is not something you do to a pretty lady.

A proposition is a statement to which we can assign truth-value.

There are just two truth-values:

- true (written T) and
- false (written F).

Examples of propositions

It's raining $\qquad 2+2=5$

There exists a species of flightless bird

$$\pi > 3$$

$$3^2+4^2=5 \qquad 3^2+4^2=5^2$$

$$T \qquad F$$

T is true and F is false.

What is the truth-value of $2+2=5$? How about $\pi > 3$? What is the truth-value of T? How about F?

Propositions vs. truth

Note: a proposition is an ‘inanimate thing’.

We are at liberty to assign a truth-value to a proposition as we see fit.

Propositions model the things we say.

Truth-values model whether those things are true.

Examples of non-propositions

Questions: Is it raining?

Commands: Breathe in ... breathe out!

Instructions: $x := x + 1$.

Numbers: 2.

Nonsense jumble of letters: All your base are belong to us. Oxford is a worthwhile university.

Combining propositions

Let P and Q range over propositions.

- Read $P \wedge Q$ as P and Q.
- Read $P \vee Q$ as P or Q.
- Read $\neg P$ as not P.
- Read $P \Rightarrow Q$ as P implies Q.
- Read $P \Leftrightarrow Q$ as P if and only if Q.

You need to know these symbols and understand what they mean.

Combining propositions

Remember that propositions get assigned truth-values.

The meaning of $\wedge$ is given by how it combines the truth-values of its propositions.

When is $P \wedge Q$ assigned truth-value true?

When is $P \vee Q$ assigned truth-value true?

... and so on.

The truth-values depend on the truth-values of P and Q:

Truth-tables

Suppose propositions p and q. Write T and F for truth and false.

p	q	$p \vee q$	$\neg p$	$p \wedge q$	$p \Rightarrow q$	$p \Leftrightarrow q$
T	T	T	F	T	T	T
T	F	T	F	F	F	F
F	T	T	T	F	T	F
F	F	F	T	F	T	T

You need to really understand how these tables work. Go play with ttable (or any number of similar tools).

`users.isr.ist.utl.pt/~etienne/cs275/ttable-page.html`

Some conventions

$\wedge$ and $\vee$ associate to the left.

$$P \wedge Q \wedge R \qquad \text{is} \qquad (P \wedge Q) \wedge R$$

$\Rightarrow$ and $\Leftrightarrow$ associate to the right.

$$P \Rightarrow Q \Rightarrow R \qquad \text{is} \qquad P \Rightarrow (Q \Rightarrow R)$$

Operators bind in the following precendence: $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$.

$$P \wedge Q \Rightarrow \neg R \Leftrightarrow S \qquad \text{is} \qquad ((P \wedge Q) \Rightarrow (\neg R)) \Leftrightarrow S$$

Some less trivial examples

Are the truth-values of $(P \wedge Q) \wedge R$ identical to those of $P \wedge (Q \wedge R)$?

Are the truth-values of $(P \Rightarrow Q) \Rightarrow R$ identical to those of $P \Rightarrow (Q \Rightarrow R)$?

So let's look at the truth tables. Let's use ttable.

Some special classes of propositions

A tautology is a proposition whose truth-value is always T.

T is a tautology. So is $P \vee \neg P$. So is $P \Rightarrow P$. So is $((P \Rightarrow Q) \Rightarrow P) \Rightarrow P$.

Check it out.

A contingency is a proposition whose truth-value is sometimes T, sometimes F. P is a contingency. So is $P \Rightarrow Q$.

A contradiction is a proposition whose truth-value is always F.

F is a constradiction. If ϕ is a tautology then $\neg\phi$ is a contradiction. So are $P \wedge \neg P$ and $P \Rightarrow \neg P$.

Example tautologies

(**Commutativity**)

$P \wedge Q \Leftrightarrow Q \wedge P$ $\qquad$ $P \vee Q \Leftrightarrow Q \vee P$

(**Associativity**)

$P \wedge Q \wedge R \Leftrightarrow P \wedge (Q \wedge R)$ $\qquad$ $P \vee Q \vee R \Leftrightarrow P \vee (Q \vee R)$

(**Distributivity**)

$P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$ $\qquad$ $P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$

(**Complement**)

$P \wedge \neg P \Leftrightarrow F$ $\qquad$ $P \vee \neg P \Leftrightarrow T$

(**Identity**)

$P \wedge T \Leftrightarrow P$ $\qquad$ $P \vee F \Leftrightarrow P$

What to do now

Read chapter 1 and sections 2.1 to 2.7 of Currie's book.

Do exercises 2.1, 2.2, 2.3, 2.4.

Check your answers to the solutions in the back of the book.

Some study tips

Prepare the next lecture by glancing through the slides, if your lecturer is efficient enough to get them online.

Revise the slides while the material is fresh in your mind.