Why software needs to be developed using rigorous engineering-based methods

Dr Jim Cooling - Lindentree Associates
Do we *really* have a problem?
Bad day at the office!

After two accidents involving police cars of Berlin, Germany, at first the drivers were blamed and appointed to a security training.

But taking into consideration the driver's nearly identical reports, which claimed that the cars on-board drive dynamic control systems had failed, BMW took on and inspected the case.

The result was: Yes, after an emergency brake exceeding a certain preset pressure on the pedal, all stability systems are disabled and can only be re-enabled by switching off the ignition for five seconds.
March 2014, RT report:
Software problems will set back F-35 joint strike fighter another year

70 percent over initial cost estimates and years behind schedule

The GAO said the plane needs eight million new lines of software code to overcome the current functionary glitches.

The F-35 program's head, Maj. Gen. Christopher Bogdan, said in a statement that "software continues to remain our number one technical risk on the program, and we have instituted disciplined systems engineering processes to address the complexity of writing, testing and integrating software.”
In **1993** the California Department of Motor Vehicles abandoned development of their driver and vehicle registration system (started 6 years earlier) having spent $44.3 million. The delivery date had receded to 1998 and the projected cost had increased by 650%.

UK Parliament PAC Report: **2014**.

- The BBC’s Digital Media Initiative (launched in 2008) was a complete failure.
  - Licence fee payers paid nearly £100 million for this supposedly essential system but got virtually nothing in return.
    - The main output from the DMI is an archive catalogue and ordering system that is slower and more cumbersome than the 40 year-old system it was designed to replace.
    - It has a running cost of £3 million a year, compared to £780,000 a year for the old system.
    - In reality the BBC only ever used the DMI to make one programme, called ‘Bang Goes the Theory’.
A double-whammy - car and house damaged!

This is the second time this car was involved in an accident involving sudden uncommanded acceleration.

Vehicle was stopped awaiting opening of automatic garage door when the throttle fully engaged, driving car into garage door.
The worst consequences of not doing things right

**Toyota's killer firmware: Bad design and its consequences**
Michael Dunn - October 28, 2013

In September 2007 75 year old Jean Bookout had been exiting an Oklahoma highway in her 2005 Toyota Camry when it suddenly accelerated. Bookout was unable to stop the car and it fell into a culvert injuring her and killing her passenger and best friend Barbara Schwarz.

Toyota recalled more than 10 million vehicles from 2009 to 2011, was subjected to congressional hearings on the problem and how it was handled, and has paid **out $1.1 billion** to settle lawsuits.

**Ford sued over problems with vehicle acceleration**
EDT March 29, 2013

The suit alleges there is a design defect in vehicles equipped with an electronic throttle control system but not a brake override system.
And we didn’t learn from the Therac 25 incident

In November of 2000, Victor Garcia and 27 other patients at the National Cancer Institute in Panama were jolted with massive overdoses of gamma rays partly due to limitations of the computer program that guided use of a radiation-therapy machine.

In the 40 months that have passed, 21 patients have died.

The International Atomic Energy Agency (IAEA) said in May 2001 that at least five of the deaths were probably from radiation poisoning and at least 15 more patients risked developing "serious complications" from radiation
Even more medical issues

Article in Le Monde (in French) lists radiation overdoses in French hospitals:

23 patients get a 20% overdose. 1 patient dies, 13 suffer complications.

In 2006, Cardinal Health, of Dublin, Ohio, stopped production of its Alaris SE infusion pumps because one reportedly killed two patients, including a 16-day-old baby that got 44.8 milliliters of intravenous nutrition, rather than 4.8 milliliters.

During the investigation into the malfunctioning pumps, nurses complained about frequent keyboard errors, while the manufacturer blamed nurses for entering the wrong drug information and then failing to double-check.

The FDA's software specialists, once they got a copy of the pump's source code, confirmed that the machine was at fault.
Why do we have these problems?
Software error: “any feature of a program which produces a system malfunction”

Three major sources/causes of errors:

- System design
- Software design
- Environmental effects

Design and coding
A wing-mounted missile on an aircraft failed to separate properly after ignition, causing the aircraft to go violently out of control.

The problem was caused by closing the missile retaining mechanism too quickly.

Estimates of the times involved (made during the system design phase) proved to be WRONG.
Environmental factors:

- How software behaves in its *normal* working environment.

A mechanical fault in a fly-by-wire control system set up conditions that the flight computer wasn’t programmed for.

The aircraft crashed.
Errors in the design process

Design and coding errors

- Syntactic
- Semantic
- Logical
- Algorithmic
A problem of syntax

Knowing syntax enable one to speak proper like wot I do

SYNTAX
'The rules concerning the arrangement of words and phrases in sentences'
Syntax issues

- There are two distinct types of syntax error.
- In the first, the wrong symbol is used.
- In the second the symbol is used wrongly.

Consider writing
\[
X = Y \ldots
\]

in a C program when what was wanted was:
\[
X = Y \ldots .
\]

Both are valid constructs; yet they produce quite different results.

Is this a serious problem?

‘a misplaced comma in one NASA program sent a Voyager spacecraft towards Mars instead of Venus’
Semantic issues

SEMANTICS → "Relating to meaning in language"

What does the customer mean? What do I mean? Program

Concept

LDA.................
MVI A..........
A problem of semantics

- Specification: If a temperature fault occurs, leave all variables ‘as is’ and sound the alarm.

- Intended response: Hold the reactor temperature steady if a fault occurs.

- Actual response: Fault occurred - cooling water flow control valve ‘frozen’ in position.

- Result: Reactor overheated and discharged into the atmosphere.
Logic errors - what and when

Mariner 18 spacecraft lost due to missing ‘not’ statement in program

When
During design and coding

Typified by
Post-check instead of pre-check
Accidental infinite loops
Forgetting to preset variables
Deadlock situations
Wrong logic condition checked
The Mars climate orbiter spacecraft ‘fly-by’ mission became a ‘fly-into’ because of a mix-up between English and metric units.
Poor software - the reasons

Root causes of poor software

- Incorrect
- Unreliable
- Unsafe
- Late
- Expensive

Company ethos
Design capability of the software team
Specific design implementations
How can we produce dependable software?
Qualities of dependable software

Correctness: “the static property that a program is consistent with its specification”

Reliability: “the extent to which a program can be expected to perform its intended function with required precision”

Safe software: “the consequences of software failure are limited by design”
Software suffers from the ‘fashion’ syndrome
The PRAGMA process is a technique for developing systems and software for real-time embedded applications.

• It is primarily intended for developing real-time embedded systems.

• It is applicable to a wide range of system structures.

• The overall approach is essentially a system-oriented top-down one but has sufficient flexibility to also handle bottom-up and middle-out activities.

• Its individual stages form an integrated structure (note: not 'seamless').

• There is a clear traceable path through the individual stages, from requirements to source code.

• Performance modelling, evaluation and analysis are an integral (though optional) part of the process.
The software ‘Guru’ at work

Also known as the ‘hack-it and bash-it’ technique.
Developing products - from problem to solution

User requirements

“What the client wants”

Analysis

Design specifications

“What is the problem?”

Design

“How can we solve the problem?”

Build specifications

“What to design”

Build

“How can we best build the product?”

The product
A general RTES development process - from problem to solution
The basics of the PRAGMA real-time development process

User requirements
“What the client wants”

The PRAGMA real-time development process

Build specifications

Analysis

Design

Describe

Understand

Specify

Design specifications

Produce an ideal design

Convert to a practical design
The extent of the PRAGMA process - software development only

User requirements → Analysis phase → Software design specifications → Software design phase → The software build specifications
The analysis phases of the PRAGMA process - major steps

- **User requirements**
  - Evaluate the system-level aspects of the problem

- **Software design specifications**
  - Specify what the objectives of the software are.

- **Analysis phase**
  - **Describe**
    - How the system is used.
    - Why the system is used.
    - The scope of the system.
    - The context of the software within the system.

- **Requirements elicitation, collation, outline analysis and recording**

- **System usage**
  - The system use case model

- **System scope**
  - The system scope model

- **System behaviour**
  - The system dynamic model

- **User interactions**
  - The system interaction model

- **Software specifications**
  - The software use case model
The design phase of the PRAGMA process - major steps

- Develop an ideal design
  - Develop a high-level (large-grain) ideal design: THE SUBSYSTEM MODEL
  - Develop a detailed (fine-grain) ideal design: THE IDEAL OBJECT MODEL
- Partition the software
  - Allocate software to processors: THE PARTITIONED OBJECT MODEL
  - Transform the design: THE SPECIFICATION OBJECT MODEL
- Develop a practical design
  - Specify concurrency aspects of the ideal design: THE IMPLEMENTATION MODEL
  - Develop a practical real-time concurrent design: THE STATICAL STRUCTURE MODEL

Software design specifications
The software build specifications

Software design phase
The PRAGMA implementation model

- Interrupt support only
- RTOS support
- Concurrent language support

Tasking diagram
Getting to the implementation model - THE crucial design step

**Ideal model**
- Based on objects

**Implementation model**
- Based on
  - Tasks
  - Threads
  - Processes

**Problem:** Inherent mismatch between the two models

**Implementation mechanisms**
- Sequential execution (e.g. C++ code units).
- Concurrency via:
  - Interrupts.
  - RTOS (e.g. VxWorks, Linux).
  - Language (e.g. Java, Ada95)
Introducing the specification model

Purpose of the specification model? To make implementation as simple as possible.
Mapping objects to tasks - \textbf{basics}

- **Ideal model**
  - \texttt{BowThruster}
  - \texttt{TunnelThruster}

- **Specification model**
  - \texttt{BowThruster}
  - \texttt{TunnelThruster}

- **Implementation model**
  - \texttt{BowThruster}
  - \texttt{TunnelThruster}

**Transformations**
- \texttt{SetSpeed}
- Channel
Example mapping - active and passive objects

Passive 'object'
Overview of the software development process for multi-computer systems

1. **Refine**
   - Develop a high-level (large-grain) ideal design
   - **THE SUBSYSTEM MODEL**

2. **Partition**
   - Develop a detailed (fine-grain) ideal design
   - **THE IDEAL OBJECT MODEL**

3. **Transform**
   - Allocate software to processors
   - **THE PARTITIONED OBJECT MODEL**

4. **Map**
   - Specify concurrency aspects of the ideal design
   - **THE SPECIFICATION OBJECT MODEL**

5. **Define the code structure of the design**
   - **THE STATIC STRUCTURE MODEL**

6. **Develop a practical real-time concurrent design**
   - **THE IMPLEMENTATION MODEL**
Recap - design phase of the PRAGMA process

For detailed information on the PRAGMA process, see www.lindentreeuk.co.uk
Conclusion of the presentation

AND ON THAT EXCITING NOTE...