AARHUS
UNIVERSITY
DEPARTMENT OF ENGINEERING

OCTOBER 2012

FORMAL METHODS INTRODUCTION

PETER GORM LARSEN (PGL@IHA.DK)
PROFESSOR
(MANY YEARS COLLABORATION IN PARTICULAR WITH JOHN FITZGERALD)

WHO AM I?

- **Professor Peter Gorm Larsen**; MSc, PhD
- 20+ years of professional experience
- ½ year with Technical University of Denmark
- 13 years with IFAD
- 3,5 years with Systematic
- 7 years with Aarhus School of Engineering
- Reviewer for EU on Research projects and applications
- Consultant for most large defence contractors on large complex projects (e.g. US Joint Strike Fighter)
- Relations to industry and academia all over the world
- Has written books and articles about VDM
- See http://pglconsult.dk/private/peter.htm for details

FORMAL METHODS INTRODUCTION

- **Formal Methods Characteristics**
- Formal Methods Europe
- Tool Support for FM

WHAT ARE FORMAL METHODS?

- **Formal Methods** refers to the use of techniques from logic and discrete mathematics in the specification, design and development of computer systems and software.
- Mathematics is NOT as difficult as many thinks
- Mastering of **complexity** using **abstraction**.
- Reduce argumentation to a **calculation** which can be checked by mechanical means.
- Replace manual reviews with a **repeatable** analysis.
- Formal methods can be used at different levels of **rigour**.

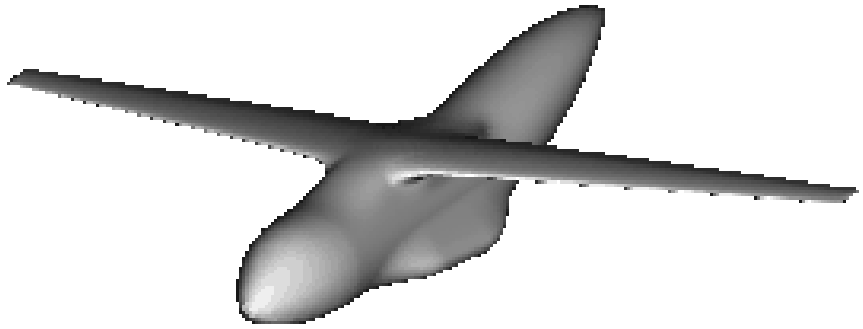
VALIDATION TECHNIQUES

- Inspection: organized process of examining the model alongside domain experts.
- Static Analysis: automatic checks of syntax & type correctness, detect unusual features.
- Testing: run the model and check outcomes against expectations.
- Model Checking: search the state space to find states that violate the properties we are checking.
- Proof: use a logic to reason symbolically about whole classes of states at once.

LEVELS OF RIGOUR

- Level 1: Use of concepts and notation from discrete maths
- Level 2: Use of formalized abstract specification languages with some mechanized support tools
- Level 3: Verification of the abstract precise specification
- Level 4: Fully formal development (refinement from abstract specifications)

MODELLING COMPUTING SYSTEMS



MODELLING COMPUTING SYSTEMS

- In other engineering disciplines (Mechanical, Electrical, Aeronautical etc.) system models are built *to help gain confidence* in requirements and designs.
- For example: wind tunnels, stress models
- Two characteristics of these models are crucial to their successful use: abstraction and rigour.

ABSTRACTION

Engineering models omit details that are not relevant to the purpose of the model.

For example:
›Windtunnel – assess aerodynamics
›Cockpit mockup – assess human factors, train pilots

The omission of detail not relevant to a model's purpose is called abstraction. The choice of which details to omit is a matter of *engineering skill*.

See for example: "Abstraction – the key to Computing?", by Jeff Kramer

ABSTRACTION

Compare these extracts from two descriptions of the same system.

The FlightFinder System is to be used by travel agents and their customers. Details are entered, including point of departure, destination, preferred dates and times. The system will respond with a range of itineraries and fares, along with the relevant restrictions.

"What"

The system records locations as nodes in a connected graph structure. Each node struct contains an array of pointers to reachable destinations plus, for each pointer, a timetable of flights stored as a hash table. Each record in the hash table has a flight number (8 character string), departure and arrival times (standard time formats) and operating dates (standard date format). To obtain the optimal route, the graph must be traversed using a shortest path algorithm on a modified adjacency matrix ...

"How"

RIGOUR

The most important property of a model of a computing system is its suitability for analysis: must be

› objective (not down to the opinion of individual engineers)
› repeatable
› susceptible to machine support.

The language in which a model is described should be rigorously defined:

› little room for disagreement about what a model actually says
› analysis tools reach the same conclusion about model's properties

MATHEMATICAL REPRESENTATION OF SOFTWARE

- Formal specifications are expressed in a mathematical notation with precisely defined vocabulary, syntax and semantics.
- Algebraic approach
- The system is specified in terms of its operations and their relationships.
- Model-based approach
- The system is specified in terms of a state model that is constructed using mathematical constructs such as sets and sequences.

CLASSES OF FORMAL METHODS

- Model-based approaches (VDM, Z, B)
- Algebraic approaches (Act One, Larch, OBJ)
- Process algebras (CSP, CCS)
- Logic-based approaches (RTL, TLA)
- Reactive approaches (Petri-nets, SDL, SAO)

- Combinations like RAISE (VDM + CSP) and LOTOS (Act One + CCS).
- ISO standards for VDM, Z, LOTOS and ITU standard for SDL

- Different strengths
- Very different kind of tool support

FORMAL METHODS IN ACADEMIA

- Tradition with Abstract Models in Europe
- Focus on Formal Development
- US focus on Automatic Verification
- FM taught at many European Universities
- Spreading from the UK
- Japan usage and supply gaining strength
- Different opinions about way ahead (e.g. Parnas, "Really rethinking formal methods", IEEE Computer January 2010)

FORMAL METHODS INTRODUCTION

- ✓ **Formal Methods Characteristics**
- ➢ **Formal Methods Europe**
- › Tool Support for FM

FORMAL METHODS EUROPE

- www.fmeurope.org
- FME = Formal Methods Europe
- Stimulate the use of formal methods by industry.
- Promote international co-operation among researchers and users of formal methods.
- Exchange ideas & identify common interests.
- Provide links between research and application areas.
- Conference approximately every 18 months

FME (VDM) CONFERENCES

Brussels, Belgium	(1987)	LNCS 252
Dublin, Ireland	(1988)	LNCS 328
Kiel, Germany	(1990)	LNCS 428
Noordwijkerhout, Netherlands	(1991)	LNCS 551/2
Odense, Denmark	(1993)	LNCS 670
Barcelona, Spain	(1994)	LNCS 873
Oxford, UK	(1996)	LNCS 1051
Graz, Austria	(1997)	LNCS 1313
Toulouse, France	(1999)	LNCS 1708/9
Berlin, Germany	(2001)	LNCS 2021
Copenhagen, Denmark	(2002)	LNCS 2391
Pisa, Italy	(2003)	LNCS 2805
Newcastle, UK	(2005)	LNCS 3582
Hamilton, Canada	(2007)	LNCS 4085
Turku, Finland	(2008)	LNCS 5014
Eindhoven, Netherlands	(2009)	LNCS 2850
Limerick, Ireland	(2011)	LNCS 6664
Paris, France	(2012)	LNCS 7436
Singapore	**(2014)**	

FORMAL METHODS INTRODUCTION

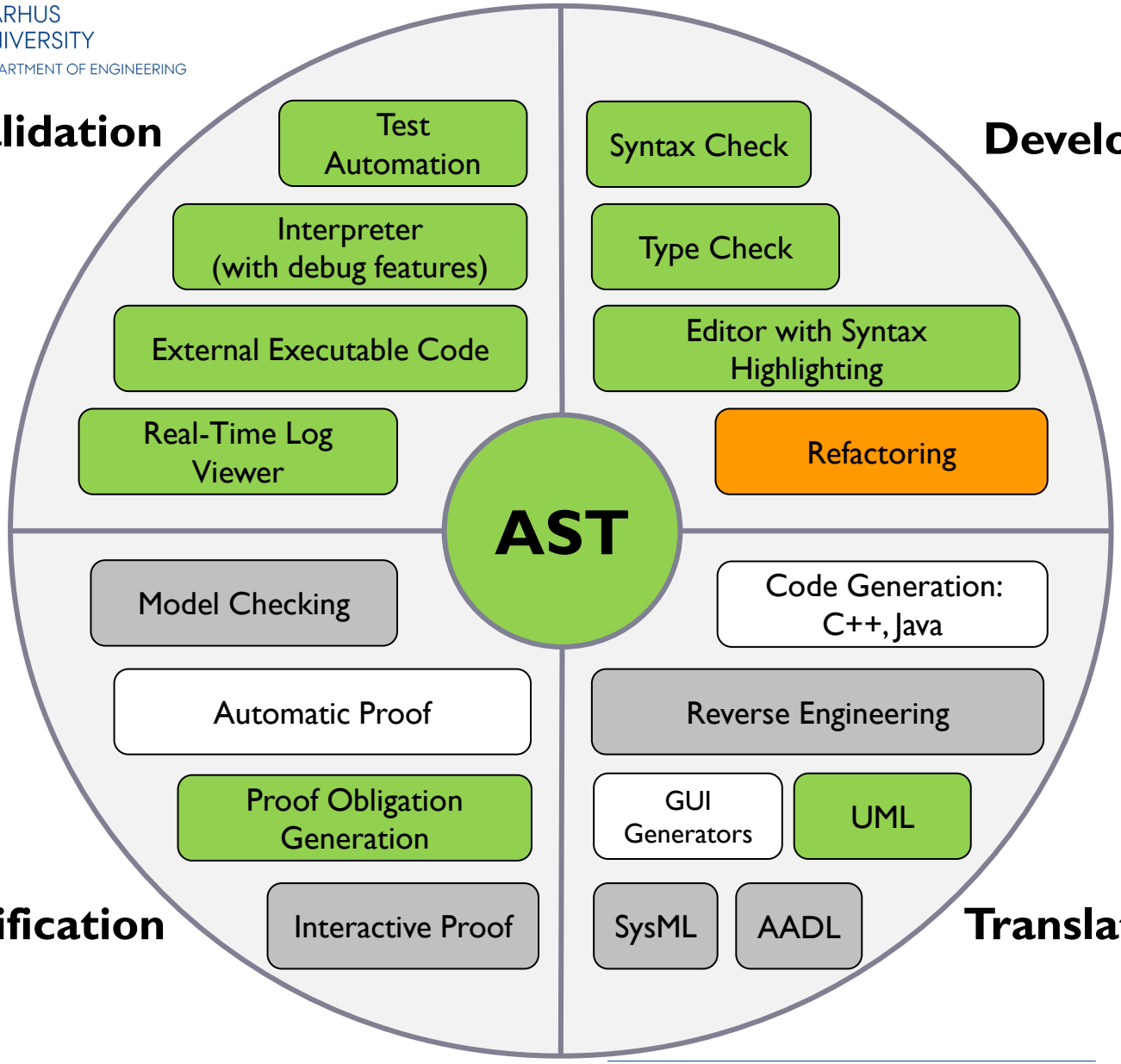
- ✓ **Formal Methods Characteristics**
- ✓ **Formal Methods Europe**
- ➢ **Tool Support for FM**

FORMAL METHODS TOOL SUPPORT

- VDMTools from SCSK
- Atelier-B from ClearSy
- FDR from Formal Systems Europe
- SCADE from Esterel Technologies
- Lots of prototype/academic tools
- Overture open-source development for VDM

FORMAL METHODS TOOL FEATURES

- Syntax checking
- Type checking
- Proof obligation generation
- Proof support
- Model checking
- Pretty printing
- Animation/Execution
- Refinement
- Test automation
- Code generation
- Graphical User Interface


Validation
Test Automation
Interpreter (with debug features)
External Executable Code
Real-Time Log Viewer
Development
Syntax Check
Type Check
Editor with Syntax Highlighting
Refactoring
AST
Verification
Model Checking
Automatic Proof
Proof Obligation Generation
Interactive Proof
Translators
Code Generation: C++, Java
Reverse Engineering
GUI Generators
UML
SysML
AADL
Available
Prototype available
Under development
Not yet started

TAKE AWAY POINTS

- There is nothing magic about formal methods
- It is just a sensible **engineering** approach
- Strong European FM supplier side
- USA still strongest on model checking for hardware
- Japan usage and supply gaining strength
- Tools are essential for success
- Consider attending FM 2014 in Singapore

THANKS FOR YOUR ATTENTION

Any questions?