Profile of IFIP Working Group 2.1

WG2.1 members

April 2009

This information brochure is an attempt to survey knowledge and expertise represented within IFIP Working Group 2.1.

- Section 1 lists the title, aim, scope and web-page of the working group.
- Section 2 lists the title, aim, scope and other working groups of TC2, WG2.1's parent body within the IFIP hierarchy.
- Section 3 lists the locations and dates of recent meetings of the group.
- Section 4 lists all members of the Working Group together with their specific research interests.

1 WG2.1: Title, aim, scope and web-page

1.1 WG2.1 Title

IFIP Working Group 2.1 — Algorithmic Languages and Calculi

1.2 WG2.1 Aim

To explore and evaluate new ideas in the field of programming, possibly leading to the design of new languages.

1.3 WG2.1 Scope

1. The study of calculation of programs from specifications.
2. The design of notations for such calculation.
3. The formulation of algorithm theories, using such notations.
4. The investigation of software support for program derivation.
5. Continuing responsibility for ALGOL 60 and ALGOL 68.

1.4 WG2.1 Web-pages

- `http://web.comlab.ox.ac.uk/people/Jeremy.Gibbons/wg21/`
- `http://www.cs.uu.nl/wiki/bin/view/IFIP21/WebHome`

2 TC2 context

Within the IFIP hierarchy, WG2.1 is part of TC2.

2.1 TC2 Title

Software: Theory and Practice

2.2 TC2 Aim

To obtain a deeper understanding of programming concepts in order to improve the quality of software by studying all aspects of the software development process, both theoretical and practical.

2.3 TC2 SCOPE

The scope of the committee encompasses all aspects of the software development process including the specification, design, implementation and validation of software systems. Areas of present activity are:

1. formal models of software concepts
2. programming languages and techniques
3. models for information storage and processing
4. programming support environments
5. user interface to software systems
6. software quality

2.4 Other working groups within TC2

- WG2.2 — Formal Description of Programming Concepts
- WG2.3 — Programming Methodology
- WG2.4 — Software Implementation Technology
- WG2.5 — Numerical Software
- WG2.6 — Database
- WG2.7 — User Interface Engineering
- WG2.8 — Functional Programming
- WG2.9 — Software Requirements Engineering
- WG2.10 — Software Architecture
- WG2.11 — Program Generation
- WG2.12 — Web Semantics

3 Recent WG2.1 related meetings

3.1 WG2.1 meetings

- #64 Weltenburg, Germany, March 30 – April 3, 2009
- #63 Kyoto, Japan, September 10 – 14, 2007
- #62 Namur, Belgium, December 11 – 15, 2006
- #61 Belek, Turkey, March 27 – 31, 2006
- #60 Pajaro Dunes, USA, May 22 – 26, 2005
- #59 Nottingham, England, September 6 – 10, 2004
- #58 Rome, Italy, January 26 – 30, 2004
- #57 New York City, USA, March 30 – April 3, 2003

- #56 Ameland, The Netherlands, September 10 – 14, 2001
- #55 Cochabamba, Bolivia, January 15 – 19, 2001

3.2 Other WG2.1 events

- IFIP TC2 Working Conference on Domain-Specific Languages, St Anne's College, Oxford, July 15th to 17th 2009.
- IFIP TC2 Working Conference on Generic Programming, Schloß Dagstuhl, Germany, July 11th and 12th 2002. [J. Gibbons and J. Jeuring (eds), *Generic Programming.* Kluwer Academic Publishers, 2003. ISBN 1 4020 7374 7.]
- IFIP TC2 WG2.1 Working Conference on Algorithmic Languages and Calculi, Le Bischenberg, Alsace, France, February 17th to 22nd, 1997. [R. S. Bird and L. G. L. T. Meertens (eds), *Algorithmic Languages and Calculi.* Chapman & Hall, London, 1997. ISBN 0 412 82050 1.]
- IFIP TC2/WG2.1/WG2.2/WG2.3 Working Conference on Programming Concepts, Methods and Calculi (PROCOMET 940), San Miniato, Italy, June 6th to 10th, 1994. [E.-R. Olderog (ed.), *Programming Concepts, Methods and Calculi.* IFIP Trans. A-56. Amsterdam: North-Holland Elsevier, 1994. ISBN 0 444 82020 5.]
- IFIP TC2 WG2.1 State-of-the-Art Seminar on Formal Program Development, Itacuruçá Island, Brazil, January 10th to 18th, 1992. [B. Möller, H. A. Partsch, S. A. Schuman (eds.), *Formal program development.* IFIP TC2/WG 2.1 State-of-the-Art Report. Lecture notes in computer science **755**. Berlin: Springer-Verlag 1993.]
- IFIP TC2/WG2.1 Working Conference on Constructing Programs from Specifications, Pacific Grove, CA, USA, May 13th to 16th, 1991. [B. Möller, (ed.), *Constructing programs from specifications.* Amsterdam: North-Holland 1991.]
- IFIP TC2/WG2.1 Working Conference on Program Specification and Transformation, Bad Tölz, Germany, April 15th to 17th, 1986. [L. G. L. T. Meertens (ed.), *Proceedings of the IFIP-WG2.1 Working Conference on Program Specification and Transformation.* Amsterdam: North-Holland, 1987.]

4 List of research interests

(as communicated by the members)

Roland Backhouse (`http://www.cs.nott.ac.uk/~rcb`):

- mathematics of program construction
- programming calculi
- category theory, relation algebra, datatype algebra, functional programming
- algorithmic problem solving

F. L. Bauer

- methodology of program development
  - formal specification (in particular, pre-algorithmic, algebraic, and logical specification)
  - development by transformation (in particular, from non-algorithmic to algorithmic form)
- concepts and semantics of programming languages
- formal logic (in particular, modal logics)
- cryptology

Richard S. Bird

- the investigation of the properties, feasibility, and usefulness of languages helping the specification and construction of good algorithms

Hendrik Boom (`http://www.pooq.com/`):

- program transformation systems
- type theories
- natural languages

Ernie Cohen

Jules Desharnais (`http://www.ift.ulaval.ca/~desharnais`):

- programming methodology
- program specification and transformation
  - mathematics of program construction and analysis
  - notations for expressing and manipulating programs
- relational and Kleene-algebraic methods in computer science

Robert B. K. Dewar (`http://www.cs.nyu.edu/faculty/dewar/`):

- programming languages and compilers
- microprocessor architectures
- software engineering and tools

Martin S. Feather (`http://www.isi.edu/software-sciences/feather/home-page.html`):

- specification and transformation
- formalization and mechanization of design and development
- specification languages
- correctness-preserving and evolutionary (meaning-changing) transformations
- goal directed design/development
- reuse and adaptation of designs/developments
- requirements engineering

Jeremy Gibbons (`http://www.comlab.ox.ac.uk/jeremy.gibbons/`):

- programming methodology
- program specification and transformation
  - notations for expressing and manipulating programs
  - algebraic approaches to programming
  - program calculation
  - model-driven development
- generic, functional, object-oriented programming

Allen Goldberg

- program transformation
- data refinement
- derivation structuring and recording

- formal approaches to System Specification
- run time analysis

Eric Hehner (`http://www.cs.utoronto.ca/~hehner/`):

- the relation between logic and programming
- formal methods of software design
- unified algebra
- high-level circuit design
- programming language semantics and design
- compiler design

Ralf Hinze (`http://www.comlab.ox.ac.uk/ralf.hinze/`):

- generic programming
- functional programming
- Haskell
- algorithm design
- purely functional data structures
- co-algebraic methods

Graham Hutton (`http://www.cs.nott.ac.uk/~gmh/`):

- reasoning about programs
- functional programming
- calculational methods
- coalgebraic methods
- recursion operators

Patrik Jansson (`http://www.cs.chalmers.se/~patrikj/`):

- long term goal: create systems (theories, programming languages, libraries and tools) which make it easy to develop reusable software components together with proofs of their correctness.
- generic programs and proofs: theory and implementation
- functional programming, semantics, modelling
- dependent types in programming, type theory, constructive logic
- specification driven software development: combining functional programming with testing and first order logic

Johan Jeuring (`http://people.cs.uu.nl/johanj/`):

- generic programming
  - programming methodology
  - type systems
  - applications
- domain reasoners
  - strategies
  - feedback
  - implementation
- functional programming

- algorithm design

Peter R. King (`http://www.cs.umanitoba.ca/~prking/`):

- specification and prototyping of reactive multimedia documents
- formal descriptions of static and dynamic properties of documents
- algorithm development based on reusability
- electronic publishing

C. H. A. Koster (`http://www.cs.kun.nl/~kees`):

- syntactic description of programming languages and natural languages
  - affix grammars
  - compiler construction tools
  - syntactic programming
- application of natural language processing in information retrieval
  - document classification
  - text mining

Charles H. Lindsey (`http://www.cs.man.ac.uk/~chl`):

- in regard to algorithmics:
  - implementation of prototype transformation systems
  - semantics of specification languages
  - domain and power domain theory
- programming languages
- ALGOL 68

Y. Annie Liu (`http://www.cs.indiana.edu/~liu/`):

- general and systematic methods and supporting tools for improving the efficiency and assuring the correctness of computations;
- program analysis and transformation for incremental computation and parallel/concurrent computation;
- applications in optimizing compilers, interactive environments, real-time and reactive systems, algorithm design, program development methods, database systems, semantic web, and security.

Andres Löh (`http://people.cs.uu.nl/andres/`):

- functional programming
- generic programming
- Haskell
- type systems

Lambert G. L. T. Meertens (`http://www.kestrel.edu/HTML/people/meertens/`):

- the investigation of the properties, feasibility, and usefulness of a language helping the specification and construction of good algorithms.

Bernhard Möller (`http://www.Math.Uni-Augsburg.DE/~moeller`):

- formal semantics
- algebraic system calculation
- infinite objects (theory and applications)
- theory of parallelism and nondeterminacy
- transformational program development
- design of high-level language concepts
- relational programming
- Kleene algebra
- modal algebra

Carroll Morgan (`http://www.cse.unsw.edu.au/~carrollm/`)

- specification and refinement
- program development calculi
- probabilistic programs/specifications/logic
- μ-calculi and games
- refinement and security

Alberto Pardo (`http://www.fing.edu.uy/~pardo/`):

- programming methodology
- program specification and transformation
  - algebraic approaches to programming
  - program calculation
- generic programming
- functional programming
- semantics of programming languages
- type theory

Helmut Partsch (`http://www.informatik.uni-ulm.de/pm/helmuth`)

- requirements engineering
- model-driven development, model-driven architecture
- formal specification (theory, methodology and application)
- transformational program development and refinement
- integration of formal methods and traditional software engineering

Manfred Paul (`http://wwwpaul.informatik.tu-muenchen.de`):

- parallel and distributed programming
- methods and tools for the construction of parallel programs
- intelligent systems for teaching

John Peck

- no information

Peter Pepper (`http://www.cs.tu-berlin.de/~pepper/`):

- modeling of safety-critical systems

  - hybrid and embedded systems
  - specification and validation
  - safe development process
  - autonomic computing and self-healing
- formal program development
  - formalization and explanation of development process
  - strategies
  - transition from specifications to operational solutions
  - functional languages
- integration of language paradigms
  - functional
  - constraint-logic
  - object-oriented

Alberto Pettorossi (`http://www.iasi.cnr.it/~adp/`):

- transformational program development:
  - finding, formalizing and studying the power of tactics and strategies
  - incorporation of efficiency considerations
  - development of logic and constraint logic programs
  - development of parallel, communicating and distributed programs
- related topics:
  - classification of algorithms (and derivation of classes of algorithms)
  - understanding the invention of suitable data structures during program development
  - development of Software Engineering techniques for programming 'in the small'
  - mechanization of tactics and strategies, relationship to Theorem Proving techniques
  - model checking via transformation
- foundations:
  - mathematics of Algorithm Derivation
  - semantics of functional and logic programming languages
  - algebras for concurrency

W. L. van der Poel

- no information

Michel Sintzoff (`http://www.info.ucl.ac.be/people/cvsintzo.html`):

- mathematics of program construction
- hybrid and reactive programs
- algorithmic schemes

Doug Smith (`http://www.kestrel.edu/HTML/people/smith/index.html`):

- automating the production of software:
  - formalizing classes of algorithms
  - automating the derivation of algorithms
  - automated deduction
  - program optimization
  - data structure design

  - computational complexity
  - models of software systems and their design and evolution.

S. Doaitse Swierstra (`http://www.cs.uu.nl/staff/doaitse.html`):

- programming methodology
  - programming techniques, especially functional programming
  - program transformations techniques
  - transformation strategies
  - type systems
- systems programming
  - grammar based systems, esp. attribute grammars
  - program transformation systems

Eiiti Wada (`http://www.wide.ad.jp/~wada/index.html`):

- program development
  - environments, interfaces
  - language oriented tools
  - mental models that support program development
- character sets
  - synthesis of Chinese character fonts

David S. Wile (`http://mr.teknowledge.com/wile.htm`):

- specification languages
  - notations for notation
  - algebraic programming concepts
  - domain-specific language design
- programming methodology
  - metaprogramming calculus
  - transformational semantics
  - transformation systems
- software architecture
  - grammar-based support tools
  - dynamic architecture semantics
  - architecture description interchange