



Overview

- Organisation
- Questions for you
- How to find (scientific) information
- Research in SET
- Overview of Master Assignments in SET
- Generic Language Technology

TU/e Technische Universiteit Eindhoven University of Technology
080606 PAGE 2

Organisation

- Lecturers:
 - Members of Software Engineering and Technology
- How to reach us:
 - Via Christine van Gils
 - Room: HG 5.60
 - Phone: 5145
- Course information:
 - <http://www.win.tue.nl/~mvdbrand/courses/seminar/0809>

TU/e Technische Universiteit Eindhoven University of Technology
080606 PAGE 3

Organisation

- Block A:
 - 2 hours lectures:
 - Weekly
 - Wednesday:
 - Time: 7th and 8th hour (15:30-17:15)
 - Location: Auditorium
- Block B, C
 - 2 hours lectures:
 - (B)weekly? (depending on number of participants)
 - Wednesday:
 - Time: 7th and 8th hour (15:30-17:15)
 - Location: Auditorium

TU/e Technische Universiteit Eindhoven University of Technology
080606 PAGE 4

Questions for you

- Where did you do your bachelor:
 - TU/e
 - Manipal
 - HBO
 - other university?
- What is your motivation to follow the SET seminar:
 - Area of interest?
 - Background?
 - Ideas about MSc assignment?

TU/e Technische Universiteit Eindhoven University of Technology
080606 PAGE 5

Goals

- Getting acquainted with the research topics of the SET group
- Getting acquainted with the basic principles of academic research:
 - reading of scientific papers
 - presenting scientific results in oral and written form
- Preparation for Master Thesis project

TU/e Technische Universiteit Eindhoven University of Technology
080606 PAGE 6

Skills

- Preparing and executing a literature search; including preparation of bibliographic references
- Critical reading, reviewing of literature; judging relevance/quality, summarizing, classifying, comparing, etc.
- Critical listening to presentations, asking probing questions in discussions
- Setting up a small-scale academic investigation
- Preparing and giving oral presentations
- Writing a technical article/report

SET Staff



SET Support staff



Christine van Gils
secretary



Erik Scheffers
software developer

SET Ph.D. students



How to find (scientific) information

- Physical library (6th floor, hoofgebouw)
- Google scholar: <http://scholar.google.com/>
- Digital libraries:
 - <http://w3.tue.nl/en/services/library/about/section/mcs/>
 - Digital library ACM: <http://portal.acm.org/portal.cfm>
 - Digital library IEEE: <http://ieeexplore.ieee.org/Xplore/dynhome.jsp>
- Lecture Notes on Computer Science (Springer): <http://www.springerlink.com/home/main.mpx>
- ScienceDirect (Elsevier): <http://www.sciencedirect.com/science/journals/>

Mission of SET

- SET's overall objective is the creation of methods and supporting tools for development and maintenance of reliable software.
 - In order to master complexity and to enable verification and validation the methods have to be formal.
 - To be cost effective, the methods have to be generic and support reuse and composition of designs, verifications and code.
 - To be able to apply the methods to projects on a realistic scale, the methods have to be supported by a well-integrated set of tools.

Generic Language Technology

- (Interactively) specify syntax and semantics of (programming) languages
- (Interactively) describe analysis and transformations
- Goals of generic approach:
 - Any (existing) programming language
 - Any analysis and transformation
 - Scalability, any data size

Generic Language Technology

- Applications:
 - Compilers
 - Integrated Development Environments
 - Language prototyping
 - Domain specific languages
 - Specification formalisms
 - Reverse engineering
 - analyze: Java, C/C++
 - transformations: COBOL

Generic Language Technology/MDE

- From code to model
- Analysis of source code
 - multi-lingual, not restricted to a specific language: C, Java, Cobol, etc.
 - flexible level of detail: parse trees, abstract syntax trees, type information, call graphs, modular dependencies, control flow graphs, data flow graphs
 - generic relations in combination with relation calculus

Generic Language Technology/MDE

- From code to model:
 - Why relevant?
 - Shift from building new code to maintaining existing code
 - How do we facilitate the software maintainer?
 - Migration from existing code to new code:
 - How can we prevent the loss of business critical information?
 - Globalization of software development:
 - off shoring
 - open source software
 - How do we check/guarantee the quality of this code?

Scannerless Generalized LR Parsing (1)

- **Scannerless:** in a traditional compiler lexical syntax is implemented by a scanner and context-free syntax by a parser. SGLR: scanner and parser are integrated
 - makes resulting parser more expressive
 - simplifies the implementation

Scannerless Generalized LR Parsing (2)

- **LR:** left-to-right (bottom-up) parsing as used by Yacc and Bison.
- **Generalized:** extends the class of accepted grammars to all context-free grammars
 - Context-free grammars are closed under composition (as opposed to, e.g., LR grammars)
 - Enables modular grammars
 - Important for large grammars and language dialects

Scannerless Generalized LR Parsing (3)

- An LR-based parser generator does not allow conflicts: (shift/reduce, reduce/reduce)
- Key ideas in SGLR:
 - split a concurrent parse when a conflict occurs
 - merge concurrent parses as soon as possible
 - an **ambiguity node** represents alternative parses
- It is undecidable whether a context-free grammar is ambiguous, but heuristics might help.

Research questions

- Syntax: SGLR (Scannerless Generalized LR)
 - Every character is a token, thus stack operation
 - Reduce stack operations by detecting regular patterns
 - Scaling to Unicode

MDE research projects in SET

- Ideals: research project on applying MD(DAVE) within ASML:
 - transformation and extension of models
 - transformation of C code to extract and integrate aspects
- Falcon: research project on applying software generation techniques in distribution centra
 - quality of model transformations
 - version management of models
- Repleo: software generation based on syntax and semantic correct templates

MSc projects

- Future:
 - Conversion of RFG code to AOP in the Thinkwise Software Factory (Sigro)
 - Porting .NET Micro Framework application to NXP platform (NXP)
 - Extending SGLR with Unicode
 - Code convention checker (Philips)
 - Controlling a paint mixing factory with the ToolBus (WB)
 - Desynchronization (Handshake Solutions)
- Current:
 - Static semantic checking of Java (Templates) using relations (TU/e)
 - State machine reconstruction from object C (Vanderlande Industries)
 - GUI for Verifying IDE (TU/e)
- Past:
 - Model checking to validate architectural rules (Philips Research)
 - Model checking the usage of protect/unprotect in ATerm library (TU/e)
 - Detection of maintenance critical components in PLC code (Vanderlande Industries)

Literature

- *The Syntax of Programming Languages: A Survey*, R.W. Floyd, IEEE Transactions on Electronic Computers 13(4), Aug. 1964.
- *Simple LR(k) grammars*, F.L. DeRemer, Commun. ACM 14(7), 453-460, 1971.
- *Semantics of context-free languages*, D.E. Knuth, Theory of Computing Systems, 29(2), 127-145, 1968.
- *The genesis of attribute grammars*, D.E. Knuth, In Proceedings of the International Conference on Attribute Grammars and their Applications, P. Deransart and M. Jourdan (Eds), 1-12, 1990.
- J. Paalkki, *Attribute grammar paradigms—a high-level methodology in language implementation*, ACM Comput. Surv. 27(2), 196-255, 1995.
- T. Reps and T. Teitelbaum, *The synthesizer generator*, SIGSOFT Softw. Eng. Notes 9(3), 42-48, 1984.
- J. Heering and P. Klint, *Semantics of programming languages: a tool-oriented approach*, SIGPLAN Not. 35(3), 39-48, 2000.

Questions?