A Unified Approach to Verification and Validation of Software Systems
CS395T – Fall 2008
Unique Number 55925
TTH 9:30AM-11AM RLM 6.126
http://www.cs.utexas.edu/~browne/uvv395f2008/

J.C Browne and W. Hunt
browne@cs.utexas.edu – hunt@cs.utexas.edu

1. Goal and Purpose

Correctness is the most critical concern of the software industry. Yet there does not exist a unified approach to verification and validation which integrates the several methods and tools for verification and validation. This course is part of an effort to provide such a unified and integrated approach. The lectures will cover the principles and methods. The 395T class will meet with and share lectures with the 378 class. The readings, analyses and projects will be quite different. Enrollees in the 395T will be expected to make some contribution to the realization of the unified approach to verification and validation which is illustrated in Figure 1. The class for the 395T is described in more detail in Section 4, "Course Work and Grading." Participants will come away from this course with a unique perspective on verification and validation.

The vision for a unified approach to verification and validation is sketched in the schematic diagram following.

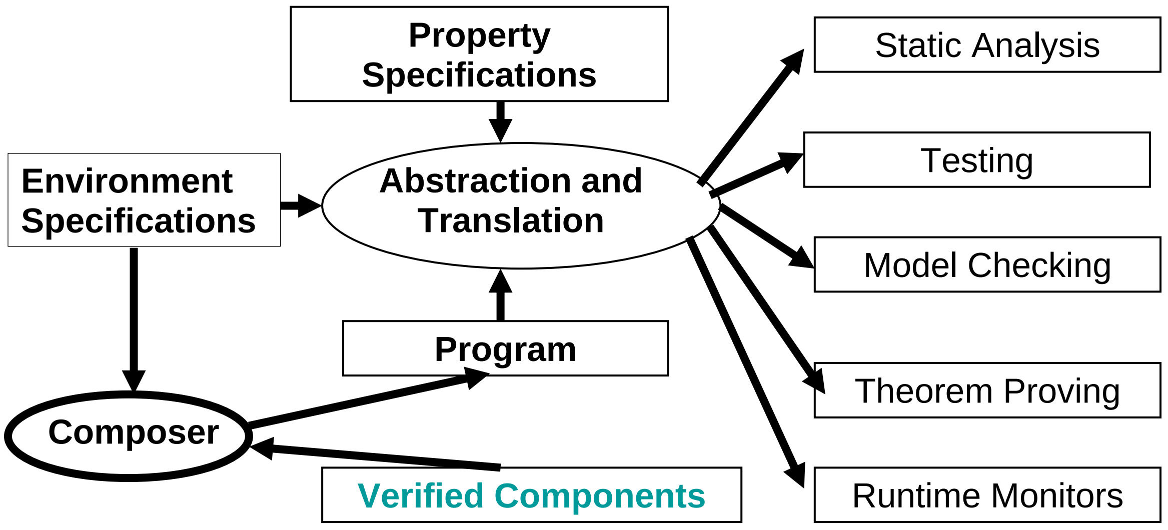

Figure 1 – Schematic of a Unified Approach to Verification and Validation of Software

This course is part of an ongoing project to create a comprehensive approach to verification and validation which is language independent, to formulate this approach as a framework for tools for specific languages, and to develop undergraduate and graduate courses that teach this unified approach to V&V. The unified framework will be used to select methods and tools for validating and verifying programs in any programming paradigm or language.

1.1 Background

The range of methods and tools which are available for validating and verifying software include static analysis of program code, conventional and systematic testing, model checking for temporal properties, runtime monitoring, and formal proofs of correctness. Each of these methods has been studied extensively in isolation, but there does not seem to be any comprehensive analysis of the synergisms and overlaps of these methods, nor have these methods and tools been integrated into a comprehensive framework for verification and validation. There are many properties such as type properties which can be verified by static analysis. If a system has not been thoroughly tested then model checking is intractable. Static analysis and model checking are both state space analysis methods which are applied to different representations of a program. Static analysis is a critical enabling technology for effective model checking. The representation of the system in the model checkable representation need span only those states and behaviors of the system that impact the property being checked. Therefore, the property and the software system are translated/abstracted together and, to the extent possible, states and behaviors not necessary for model checking of the property are projected out of the system. This projection is based on static analysis techniques. Sometimes there are properties that cannot be model checked. These properties can be compiled to monitors that validate properties at runtime. Generation of runtime monitors is based on static analysis to determine how and where to add monitoring code. Model checking, where a formally specified property is verified over all paths and states, can be viewed as exhaustive testing based on formal specifications. Proof methods are often applied to pre-condition/post-condition pairs. There is a set of pre-condition/post-condition pairs which can be expressed in temporal logic and verified by model checking. Pre-condition/post-condition pairs can also be validated by runtime monitors.

One unifying conceptual element is a “universal” property specification language from which properties can be verified by static analysis, testing, model checking, proof methods or compiled to runtime monitors as appropriate or required. The prototype for this unified property specification language is the Property Specification Language [Acellera,2004] developed by the Acellera Consortium for simulation-based testing and model checking-based verification of hardware systems. A second unifying conceptual element is the common set of component-oriented set of design principles which enable effective and scalable application of both formal and informal validation and verification. A third unifying concept is a consistent formalism for generating models through abstraction and static analysis.

1.2 Course Content

Students enrolled in the 395T will not only learn about the several methods for verification and validation. They will be a part of the development of a unified approach through the analyses and projects which are a part of the course.

The principles and mechanisms for validation and verification are language independent but the tools implementing the mechanisms are language specific. The lectures will be largely language independent. A substantial portion of the lectures will be devoted to design for verification and validation and an integrated and comprehensive approach to specification of properties to be verified and evaluated.

The content for the course will include:

a. Design for test and verification.
b. Unified Property Specification
c. Introduction to program analysis (static analysis methods).
d. Formal and complete approaches to testing:
    - Specification of properties, behaviors and assertion
    - Test coverage algorithms based on static analysis processes
    - Testing as a continuous process integrating runtime monitoring with conventional testing, model checking and proof-based verification.
e. Applied model checking:
    - Model checking as the endpoint of testing
    - Property formulation
    - Compositional reasoning
f. Classical Dijkstra/Hoare and other proof-based verification.
    - This material is already covered in other courses and will not be repeated but the role of this material in a comprehensive approach to verification and validation will be covered.
g. Run-Time Monitoring
    - Methods and Tools
    - Automated compilation of property monitors.
h. Integration of all the methods in a coherent, complete structure for validation and verification.
i. Extension of verification and validation to security policy issues such as information flow.
j. Failure analysis, fault-tolerance, practical self-stabilization, etc.

2. Student Prerequisites

Graduate standing in Computer Sciences. Students are advised to consult with the instructor before registering for this course.

3. Texts and Course Materials

There are many monographs and texts focusing on each topic concerning validation and verification (particularly testing). The text for the CS378 section is "Software Testing and Analysis" by Pezze and Young. The text most nearly matching the coverage of this course from the graduate perspective is "Software Reliability Methods" by Doron A. Peled which is listed as an optional text for this class.   There are survey and tutorial articles and a large amount of web-based material is available on each topic. Each topic will be covered through assignment of a set of papers to be read with reports on key papers.

4. Course Work and Grading

The responsibilities for those enrolled in the 395T will include reading and presenting one or more key papers dealing with some aspect of unification of verification and validation methods and execution of a project which contributes to the realization of the unified approach sketched in Figure 1. The project may be an analysis or an implementation of some aspect of the process represented in Figure 1. The projects will include a presentation of the project and a report which should in the fashion of a conference paper defining and describing the project.

Grades will be assigned on the basis of the content of the project, the presentation and the report.

5. Approximate Lecture Schedule

An approximate lecture schedule follows. There will be several guest lectures by experts on some of the topics.

Lecture Date	Lecture Topic	Reference Material
8/28/2008	Unified Approach to Verification and Validation	Lecture Notes
9/2/2008	Designing for Verification and Validation	Lecture Notes and web references
9/4/2008	Property Specification : Temporal Logics, Floyd/Hoare Logics, JML Pre-conditions, Post-conditions, invariants, etc	Lecture Notes and Web references: http://cnx.org/content/m12317/latest/ http://ieeexplore.ieee.org/iel5/6783/18169/00841031.pdf ftp://ftp.cs.iastate.edu/pub/leavens/JML/jmldbc.pdf
9/9/2008	Property Specification : Temporal Logics, Floyd/Hoare Logics, JML Pre-conditions, Post-conditions, invariants, etc	Lecture Notes and Web references: http://cnx.org/content/m12317/latest/ http://ieeexplore.ieee.org/iel5/6783/18169/00841031.pdf ftp://ftp.cs.iastate.edu/pub/leavens/JML/jmldbc.pdf
9/11/2008	Models, Abstractions and Compositionality	PY – Chapters 2, 3 and 5, lecture notes
9/16/2008	Static Analysis – Type checking, data flow	Lecture Notes, PY – Chapters 5, 6 and 13. and web references

	analysis, control flow analysis	
9/18/2008	Testing – Transition from informal to structured testing	PY Chapters 5 and 6, lecture notes and web references
9/23/2008	Fundamentals of Model Checking	Lecture Notes, PY – Chapters 5 and 8, web references.
9/25/2008	Symbolic Execution and its applications	Lecture Notes , PY – Chapter 7 and web references
9/30/2008	Translation/Abstraction based Unification of Static Analysis, Testing, Model Checking and Runtime Monitoring	Lecture Notes and web references Lecture Notes, PY –Chapter 19 and web references: http://portal.acm.org/citation.cfm?id=760066
10/02/2008	Testing in Depth	PY – Chapters 10,11,12,13,14,15
10/07/2008	Fundamentals of Proof Methods	Lecture Notes, PY – Chapters 5 and 8, web references.
10/9/2008	Automated Proof Systems - Demonstrations	(Systems to be chosen) Key, etc.
10/14/2008	Proof Systems – Software Case Study	Lecture Notes and web references
10/16/2008	Mid-Term Exam – CS378 Section	
10/22/2008	Proof Methods Revisited	Lecture Notes and web references
10/24/2008	Automated Formal Proof Methods	Guest Lecture
10/29/2008	Runtime Monitoring	
10/31/2008	Specification and Verification of Non-functional Properties – Performance and Security	Lecture notes and web materials
11/04/2008	Process Algebras and Process Calculi	Lecture Notes and web references
11/06/2008	Guest Lecture	
11/11/2008	Guest Lecture	
11/13/2008	Project Presentations	
11/18/2008	Project Presentations	
11/20/2008	Project Presentations	
11/25/2008	Project Presentations	
12/2/2008	Project Presentations	
12/4/2008	Project Presentations	