

Software Verification and Validation (VIMMD052)

Exam Topics in 2017

1.	<ul style="list-style-type: none">• The notion of verification and validation. Overview of the typical verification and validation activities during software development.• Efficient verification of complex systems by symbolic model checking.	<ul style="list-style-type: none">• L01a• L08a
2.	<ul style="list-style-type: none">• Basic formalisms for modelling behaviour: Kripke Structure, Kripke Transition System, Labelled Transition System, Timed Automata.• Formal relations for refinement checking: “may preorder” and “must preorder”, their relationship with testing.	<ul style="list-style-type: none">• L04b• L17b
3.	<ul style="list-style-type: none">• Verification of the software requirement specification: criteria and techniques.• Verification of invariant properties by bounded model checking.	<ul style="list-style-type: none">• L02• L08b
4.	<ul style="list-style-type: none">• Verification of the software architecture design: criteria and techniques.• Formalization and checking of requirements using HML and linear temporal logics (LTL).	<ul style="list-style-type: none">• L03• L05
5.	<ul style="list-style-type: none">• Verification of the detailed design: criteria and techniques. Categorization of the typical techniques of formal verification.• Model based test case generation by model checking and bounded model checking.	<ul style="list-style-type: none">• L04a• L17b
6.	<ul style="list-style-type: none">• The role of development standards in the verification and validation of critical systems.• Software model checking: The counterexample guided abstraction refinement (CEGAR) approach with predicate abstraction.	<ul style="list-style-type: none">• L01c• L13
7.	<ul style="list-style-type: none">• Verification of program source code: criteria and techniques.• Model checking of time dependent behaviour: basic modelling formalism (timed automata) and timed temporal logic.	<ul style="list-style-type: none">• L12• L10
8.	<ul style="list-style-type: none">• Specification based testing of software modules: test design techniques.• Correctness criteria and basic strategies for proving program correctness.	<ul style="list-style-type: none">• L15a• L14

9.	<ul style="list-style-type: none"> • Structure based testing of software modules: test coverage criteria. • Formal relations for checking behavioural equivalence: Strong bisimulation and weak bisimulation (observational equivalence). 	<ul style="list-style-type: none"> • L15b • L11b
10.	<ul style="list-style-type: none"> • Model based test case generation techniques: graph based algorithms. • Model checking of stochastic properties: basic modelling formalism and temporal logic (Continuous Stochastic Logic). 	<ul style="list-style-type: none"> • L17a • L09
11.	<ul style="list-style-type: none"> • Software integration testing techniques. • Formalization and checking of requirements using branching time temporal logics (CTL* and CTL). 	<ul style="list-style-type: none"> • L18 • L06
12.	<ul style="list-style-type: none"> • Verification during software maintenance: criteria and techniques. • Source code based test input generation by symbolic execution. 	<ul style="list-style-type: none"> • L19 • L16