1. Цели освоения дисциплины

Цель курса – Целью освоения дисциплины является изучение основных принципов использования формальных методов в программной инженерии, в том числе, изучение основных математических моделей и методов их анализа и синтеза с целью получения навыков анализа и проектирования программного обеспечения с использованием формальных методов.

Задачами данного курса являются:

- освоение студентами базовых современных достижений в области использования формальных методов в программной инженерии;
- формирование практических навыков анализа и проектирования программного обеспечения на основе формальных методов.

2. Место дисциплины в структуре образовательной программы

Изучение данной дисциплины базируется на знаниях, полученных студентами при освоении учебных дисциплин:

- «Дискретная математика»,
- «Программирование»,
- «Информатика, математическая логика и теория алгоритмов»,
- «Построение и анализ алгоритмов»,
- «Архитектура вычислительных систем»,

1 Тематический план учебной дисциплины

№	Название раздела	Всего часов	Аудиторные часы			Самостоя-тельная работа
			Лекции	Лабораторные работы	Практические занятия	
1.	Введение. Использование формальных методов при анализе и проектировании программного обеспечения	20	2	0	0	18
2.	Модели с конечным числом переходов: конечные автоматы и полуавтоматы, расширенные и временные автоматы, композиция автоматных моделей, сети Петри	95	10	0	22	63
3.	Тестирование и верификация программного обеспечения на	180	24	12	24	108

	основе формальных моделей					
4.	Проектирование программного обеспечения на основе формальных моделей	95	12	0	22	63
Итого:		**380**	**48**	**12**	**68**	**252**

3. Формы контроля знаний студентов

Тип контроля	Форма контроля	1 год				Параметры
		1 модуль	2 модуль	3 модуль	4 модуль	
Текущий		1-6 неделя	1-6 неделя	1-6 неделя	1-6 неделя	
	Домашнее задание		*			Сдача не позднее, чем за 15 дней до экзамена
	Контрольная работа			*		Письменная работа 60 минут
Итоговый	Экзамен		*		*	Устный экзамен

4. Критерии оценки знаний, навыков

В рамках курса слушателям предлагается выполнить домашнее задание, контрольную работу, сдать два итоговых экзамена. Оценки за контрольную работу, за домашнее задание и за каждый экзамен выставляются по 10-ти балльной шкале.

5. Порядок формирования оценок по дисциплине

Оценка за первый семестр (итоговая оценка $О_{итог.1}$) складывается из накопленной оценки 1 семестра ($О_{накопл.1}$) и оценки за устный экзамен в конце 2-го модуля ($О_{экз.1}$):

$$О_{накопл.1} = О_{дом.зад.} ;$$

$$О_{итог.1} = 0.5 \cdot О_{накопл.1} + 0.5 \cdot О_{экз.1} ;$$

Оценка за второй семестр (итоговая оценка $О_{итог.2}$) складывается из накопленной оценки ($О_{накопл.2}$) и оценки за устный экзамен в конце 4-го модуля ($О_{экз.2}$), в то время как $О_{накопл.2}$ складывается из $О_{итог.1}$ и оценки за контрольную работу ($О_{контр.}$)

$О_{накопл.2} = 0.7 \cdot О_{итог.1} + 0.3 \cdot О_{контр.}$;

$О_{итог.2} = 0.5 \cdot О_{накопл.2} + 0.5 \cdot О_{экз.2}$;

Оценка за 2-й семестр ($О_{итог.2)}$ является результирующей.

Способ округления оценок – арифметический.

6. Содержание дисциплины

Тема 1. Введение. Формальные методы в программной инженерии. История появления формальных методов и оценки эффективности их использования. Необходимость их использования.

Тема 2. Модели с конечным числом переходов: конечные автоматы и полуавтоматы, расширенные и временные автоматы, композиция автоматных моделей, сети Петри. Обсуждение использования моделей с бесконечным числом состояний и переходов. Основные задачи анализа данных моделей, их связь с анализом качества прикладного и системного программного обеспечения. Сложность задач анализа моделей с конечным числом переходов. Идентификация состояний в автоматных моделях. Построение проверяющих тестов с гарантированной полнотой на основе классических и неклассических автоматных моделей. Полнота проверяющих тестов. Многокомпонентные программные системы. Композиция автоматных моделей. Задачи анализа композиции; осцилляции и тупики. Сети Петри.

Тема 3. Тестирование и верификация программного обеспечения на основе формальных моделей. Использование верификаторов SPIN и JPF. Тестирование на соответствие спецификации и совместимости. Отношении соответствия (конформности). Тестирование программных реализаций на соответствие спецификации с использованием автоматных моделей. Синтез проверяющих тестов для программных реализаций на основе расширенных и временных автоматов. Тестирование автоматных композиций на совместимость, проверка наличия тупиков и зацикливаний.

Тема 4. Проектирование программного обеспечения с использованием формальных моделей и методов. Автоматное программирование. Автоматическое и полуавтоматическое проектирование программного обеспечения по автоматному описанию, например, UML. Достоинства и недостатки автоматической кодогенерации.

7. Оценочные средства для текущего контроля и аттестации студента

a. Тематика заданий текущего контроля

1. Лабораторная работа. Тестирование программных реализаций на основе автоматных моделей. Домашнее задание: отчет по лабораторной работе.
2. Контрольная работа. Тестирование на основе неклассических автоматных моделей.

b. Примеры контрольных вопросов для экзамена

1. Что такое конечный автомат? Какова семантика входных и выходных символов конечного автомата? Каким образом представляется последовательностная функция посредством конечного автомата?
2. Какие существуют модели неисправности на основе конечного автомата? Какие Вам известны методы построения тестов с гарантированной полнотой по конечному автомату?
3. Что такое тестирование на соответствие спецификации?
4. Что такое тестирование многокомпонентного программного обеспечения на совместимость?
5. Тестирование протокольных реализаций: к каких случаях удобно использовать активное и пассивное тестирование?
6. Верификаторы SPIN и JPF. Какова их роль в современной верификации программного обеспечения?
7. Какие существуют этапы проектирования программного обеспечения?
8. Что такое автоматное программирование?
9. Каковы достоинства и недостатки автоматической генерации программного кода?

8. Учебно-методическое и информационное обеспечение дисциплины

a. Основная литература

1. C. Kaner, R. L. Fiedler. Foundations of Software Testing. Context-Driven Press, 2013.
2. Крупский В. Н. Математическая логика и теория алгоритмов: [учебное пособие для бакалавров, обучающихся по направлениям подготовки "Информатика и вычислительная техника", "Информационные системы"] / В. Н. Крупский, В. Е. Плиско. - Москва : Академия, 2013. – 415 с.
3. Villa, T., Yevtushenko, N., Brayton, R., Mishchenko, A., Petrenko, A., Sangiovanni-Vincentelli. 2012. The Unknown Component Problem: Theory and Applications. Springer, 2012, 313 p.
4. Евтушенко Н.В. Недетерминированные автоматы: анализ и синтез: учебное пособие, ч.1 / Н. В. Евтушенко, и др.. Томск: Том. гос. ун-т, 2006 - 2013.
5. C. Kaner, R. L. Fiedler. Foundations of Software Testing. Context-Driven Press, 2013.
6. Н.И. Поликарпова, А.А. Шалыто. Автоматное программирование. Санкт-Петербург, 2008, 147 с.

b. Дополнительная литература

1. Гилл А. Введение в теорию конечных автоматов / А. Гилл; под ред. П.П. Пархоменко. М.: Наука, Физматлит, 1966, 272 с.
2. 3. Майерс Г. Д. Искусство тестирования программ / Г. Майерс; Пер. с англ. под ред. Б. А. Позина. - М. : Финансы и статистика, 1982. - 176 с.: ил.
3. Олифер В. Г. Компьютерные сети. Принципы, технологии, протоколы : [учебное пособие для вузов по направлению 552800 "Информатика и вычислительная техника" и по специальностям 220100 "Вычислительные машины, комплексы, системы и сети",

220400 "Программное обеспечение вычислительной техники и автоматизированных систем"] / В. Г. Олифер, Н. А. Олифер. - 5-е изд. - Санкт-Петербург [и др.] : Питер, 2016. - 991 с.: ил., табл.- (Учебник для вузов) - (Стандарт третьего поколения).
4. Д. Месарош. Шаблоны тестирования xUnit. М.: Вильямс, 2008.
5. Г. Майерс. Искусство тестирования программ. М.: Финансы и статистика, 1982.
6. A. P. Mathur. Foundations of Software Testing. Copymat Services, 2006.
7. Gerald J. Holzman The Spin Model Checker: Primer and Reference Manual – Addison-Wesley, 2004.
8. Nica, S. On the Use of Constraints in Program Mutations and its Applicability to Testing, PhD thesis, 2013, Graz Technical University.
9. Petrenko, A., Boroday S., Groz, R.: Confirming Configurations in EFSM Testing. IEEE Trans. Software Eng., 2004, 30(1), pp. 29-42.
10. El-Fakih, K., Salameh, T., Yevtushenko, N. On Code Coverage of Extended FSM Based Test Suites: An Initial Assessment. LNCS, 2014, 8763, pp. 198-204.
11. Proccedings of Intern Conf. on Software Testing, ICST, 2008 - 2015, and Testing Systems and Software, ICTSS, 1991 – 2015.
12. A. Ermakov, N. Yevtushenko. Increasing the fault coverage of tests derived against Extended Finite State Machines. System informatics, 2016, 7, pp. 23-32.
7. Kushik, N., Lopez, J., Cavalli, A., Yevtushenko, N. Optimizing Protocol Passive Testing through ‘Gedanken’ Experiments with Finite State Machines. In Proc. of the Intern conference on Sofware Quality and Reliability, QSR’2016 .
8. J. Tretmans Model based testing with labeled transition systems // Formal Methods and Testing. — 2008, pp. 1–38.
9. Dirk Beyer, Thomas A. Henzinger, Ranjit Jhala, and Rupak Majumdar. The Software Model Checker Blast. *International Journal on Software Tools for Technology Transfer (STTT)*, 9(5-6), 2007, pp. 505-525.
10. Thomas A. Henzinger, Ranjit Jhala, Rupak Majumdar, and Gregoire Sutre. Software Verification with Blast. In *Proceedings of the 10th SPIN Workshop on Model Checking Software (SPIN 2003)*, LNCS 2648, Springer-Verlag, 2003, pp. 235—239.

c. Ресурсы информационно-телекоммуникационной сети Интернет

1. А.К.Петренко, А.В.Хорошилов, Е.В.Корныхин. Лекция 8. Тестирование на основе формальных моделей, 2012. – URL: http://sp.cmc.msu.ru/courses/fmsp/2012/slides/lecture8.pdf (дата обращения: 03.10.2016).
2. Верификация программного обеспечения : курс лекций: [Электронный ресурс] / С.В. Синицын, Н.Ю. Налютин. – Московский инженерно-физический институт (государственный университет). – ФГАУ ГНИИ ИТТ «Информика» . – М., 2005 – 2016. – URL: http://window.edu.ru/window/library/pdf2txt?p_id=18858&p_page=1.
3. JUnit 4, documentation. [Электронный ресурс] // JUnit Home Page. – URL: http://junit.org/junit4/ (дата обращения: 10.04.2016)
4. Java Path Finder. Documentation / The National Aeronautics and Space Administration (NASA). Home page.– URL: http://JavaPathFinder. sourceforge.net/