

ЛЕКЦИИ ПО ИНФОРМАТИКА

Аврам Ескенази

Нели Манева

# СОФТУЕРНИ ТЕХНОЛОГИИ

Второ преработено и допълнено издание

КЛМН, София, 2006

# Съдържание

Увод	11
I Модели на софтуера	15
1 Основни понятия	17
1.1 Програми и програмни продукти	17
1.1.1 Хронология	17
1.1.2 Проблеми	18
1.1.3 Определения	19
1.2 Характеристики на софтуера	19
1.2.1 Необходими ресурси	19
1.2.2 Абстрактност на софтуера	20
1.2.3 Уникалност на софтуерното производство	21
1.2.4 Мултидисциплинарност на разработването на софтуер	21
1.2.5 Специфични проблеми на надеждността	22
1.2.6 Рискове	23
1.2.7 Софтуерът – средство, а не цел	23
1.2.8 Мащаби на производителността	24
1.3 Софтуерни технологии	24
1.3.1 Терминология	24
1.3.2 Определения	25
1.3.3 Цели	26
1.3.4 Наука и практика	27
2 Моделиране и модели на жизнения цикъл на ПП	31
2.1 Моделиране на жизнения цикъл	31
2.1.1 Защо се нуждаем от модел на жизнения цикъл	31
2.1.2 Какви свойства има моделът	31

2.1.3	Какви могат да бъдат следствията от един успешен модел . . . . .	32
2.2	Класификация на моделите на ЖЦ на ПП . . . . .	33
2.3	Типове модели на ЖЦ на ПП . . . . .	34
2.3.1	Стандартни хронологични модели . . . . .	35
2.3.2	Функционален модел . . . . .	35
2.3.3	Разклонен модел на Фокс . . . . .	36
2.3.4	Тримерен модел на Питърс-Трип . . . . .	37
2.3.5	Частични модели . . . . .	38
2.3.6	Каскаден модел . . . . .	40
2.3.7	Прототипен модел . . . . .	41
2.3.8	Еволюционен модел . . . . .	42
2.3.9	Спирален модел . . . . .	43
<b>3</b>	<b>Модел на Гънтър</b>	<b>45</b>
3.1	Обща характеристика . . . . .	45
3.2	Фази . . . . .	45
3.2.1	Изследване . . . . .	45
3.2.2	Анализ на осъществимостта . . . . .	46
3.2.3	Проектиране . . . . .	47
3.2.4	Програмиране . . . . .	47
3.2.5	Оценка . . . . .	48
3.2.6	Използване . . . . .	48
3.3	Функции . . . . .	48
3.3.1	Планиране . . . . .	49
3.3.2	Разработване . . . . .	49
3.3.3	Обслужване . . . . .	49
3.3.4	Документиране . . . . .	50
3.3.5	Изпитания . . . . .	50
3.3.6	Поддържане . . . . .	51
3.3.7	Съпровождане . . . . .	51
<b>II</b>	<b>Технологии за разработване на софтуер</b>	<b>53</b>
<b>4</b>	<b>Структурни технологии</b>	<b>55</b>
4.1	Въведение . . . . .	55
4.2	Определяне на софтуерната система . . . . .	56
4.2.1	Определяне на изискванията . . . . .	56
4.2.2	Аналитичен модел . . . . .	58



4.3	Проектиране . . . . .	62
4.3.1	Основни понятия . . . . .	62
4.3.2	Методи и средства за проектиране . . . . .	62
4.3.3	Етапи на проектиране . . . . .	63
4.3.4	Методи за описване на проекти . . . . .	64
4.3.5	Организационни аспекти на проектирането . . . . .	66
4.3.6	Оценяване на качеството на проекти . . . . .	66
4.4	Програмиране . . . . .	67
4.5	Интегриране и тестване . . . . .	68
4.5.1	Модулно (поелементно) тестване . . . . .	68
4.5.2	Интеграционно тестване . . . . .	69
<b>5</b>	<b>Обектно-ориентиран подход за разработване на софтуер</b>	<b>71</b>
5.1	Основни понятия . . . . .	71
5.2	Обектно-ориентиран анализ и проектиране . . . . .	72
5.3	ОО програмиране и тестване . . . . .	74
5.4	Управление на ОО разработване . . . . .	76
5.4.1	Управление на процеса на разработване . . . . .	76
5.4.2	Управление на проекта и продукта . . . . .	76
5.4.3	Управление на персонала . . . . .	79
5.5	Въвеждане на обектно-ориентирания подход . . . . .	79
<b>6</b>	<b>Съвременни техники за разработване на софтуер</b>	<b>83</b>
6.1	Ускорено разработване . . . . .	83
6.1.1	Гъвкави (agile) методи . . . . .	83
6.1.2	Разработване с прототипиране . . . . .	84
6.1.3	Разработване с участието на потребителя . . . . .	87
6.2	Мултиплициране („Re-use“) . . . . .	89
6.2.1	Същност на re-use подхода . . . . .	89
6.2.2	Разработване чрез компоненти . . . . .	93
6.3	Разработване на надежден софтуер . . . . .	94
6.3.1	Разработване на софтуер с минимизиране на дефектите в него . . . . .	94
6.3.2	Софтуер с приемливо ниво на грешки . . . . .	95
6.3.3	Защитно програмиране . . . . .	97
<b>7</b>	<b>Екстремно програмиране</b>	<b>99</b>
7.1	Необходимост . . . . .	99
7.2	Корени . . . . .	99
7.3	Същност . . . . .	100

7.4	12 практики . . . . .	102
7.4.1	Планиране – Planning Game . . . . .	102
7.4.2	Малки рилийзи – Small releases . . . . .	102
7.4.3	Метафора – Metaphor . . . . .	102
7.4.4	Просто проектиране – Simple design . . . . .	102
7.4.5	Тестване – Testing . . . . .	102
7.4.6	Рефакторинг – Refactoring . . . . .	102
7.4.7	Програмиране по двойки – Pair programming . . . . .	103
7.4.8	Колективна собственост – Collective ownership . . . . .	103
7.4.9	Непрекъснатата интеграция – Continuous integration . . . . .	103
7.4.10	40-часова седмица – 40-hour week . . . . .	103
7.4.11	Клиентът при разработчика – On-site customer . . . . .	103
7.4.12	Стандарти за кодиране – Coding standards . . . . .	103
7.5	Същност . . . . .	103
7.5.1	Планиране – Planning Game . . . . .	103
7.5.2	Малки рилийзи – Small releases . . . . .	104
7.5.3	Метафора – Metaphor . . . . .	105
7.5.4	Просто проектиране – Simple design . . . . .	105
7.5.5	Тестване – Testing . . . . .	106
7.5.6	Рефакторинг – Refactoring . . . . .	106
7.5.7	Програмиране по двойки – Pair programming . . . . .	106
7.5.8	Колективна собственост – Collective ownership . . . . .	107
7.5.9	Непрекъснатата интеграция – Continuous integration . . . . .	107
7.5.10	40-часова седмица – 40-hour week . . . . .	107
7.5.11	Клиентът при разработчика – On-site customer . . . . .	108
7.5.12	Стандарти за кодиране – Coding standards . . . . .	108
7.6	Ролята на ръководителя . . . . .	109
7.6.1	Мениджър (manager) . . . . .	109
7.6.2	Водач (coach) . . . . .	110
7.6.3	Проследяващ (tracker) . . . . .	111
7.7	Ролята на програмиста . . . . .	112
7.8	Съвсем практически аспекти . . . . .	113

### III Измерване на софтуера

117

#### 8 Измерване в софтуерното производство

119

8.1	Основни понятия . . . . .	119
8.2	Методологични проблеми на измерването . . . . .	122



<b>9</b>	<b>Софтуерни метрики</b>	<b>127</b>
9.1	Въведение	127
9.2	Класификация на софтуерните метрики	127
9.3	Примери за софтуерни метрики	130
9.4	Обектно-ориентирани метрики	139
<b>10</b>	<b>Оценяване на качеството на софтуера</b>	<b>143</b>
10.1	Общи понятия	143
10.2	Модели на качеството на софтуера	144
10.2.1	Модел на Боем	144
10.2.2	Типичен йерархичен модел	145
10.2.3	Класификационен модел	153
<b>11</b>	<b>Модел на Боем за цената на разработване на софтуер</b>	<b>159</b>
11.1	Необходимост и цели	159
11.2	Критерии	160
11.2.1	Определеност	160
11.2.2	Точност	160
11.2.3	Обективност	161
11.2.4	Детайлност	161
11.2.5	Устойчивост	161
11.2.6	Област на приложение	162
11.2.7	Конструктивност	162
11.2.8	Простота на прилагане	162
11.2.9	Предсказуемост	163
11.2.10	Икономичност	163
11.3	Моделът на Boehm COSOMO	163
11.3.1	Цели и основни идеи	163
11.3.2	Същност на модела	164
11.3.3	Пример и следствия	164
11.3.4	Усъвършенстване на модела	165
11.3.5	Критика на „редове първичен код“	167
<b>12</b>	<b>Други модели за цената на разработване на софтуер</b>	<b>171</b>
12.1	Метод на функционалните точки	171
12.1.1	История и мотиви	171
12.1.2	Същност на модела	172
12.1.3	Процедура по пресмятането	173
12.1.4	Използване на модела	175
12.2	Други модели	177

12.2.1 Doty . . . . .	177
12.2.2 SPQR . . . . .	177
12.2.3 ESTIMACS . . . . .	177
12.2.4 BANG . . . . .	178
12.3 Оценяване при обектна ориентираност . . . . .	178
<b>IV Организация на софтуерното производство</b>	<b>181</b>
<b>13 Откриване и поправяне на дефекти</b>	<b>183</b>
13.1 Основни понятия . . . . .	183
13.2 Основни дейности за откриване и отстраняване на грешки	184
13.3 V-модел за разработване и тестване на софтуер . . . . .	187
13.4 Автоматизиране на дейностите настройване и тестване . .	188
13.5 Осъществяване на тестването . . . . .	191
13.6 Метрики за тестването . . . . .	192
<b>14 Дейности, осигуряващи разработването на софтуера</b>	<b>195</b>
14.1 Съпровождане . . . . .	195
14.1.1 Същност на съпровождането . . . . .	195
14.1.2 Осъществяване на съпровождането . . . . .	196
14.1.3 Управление на внасянето на изменения . . . . .	196
14.1.4 Разходи и цена на съпровождането . . . . .	197
14.1.5 Автоматизирани средства, подпомагащи съпровожданието . . . . .	198
14.2 Управление на софтуерните конфигурации . . . . .	199
14.3 Документиране . . . . .	202
14.4 Автоматизиране . . . . .	204
14.4.1 Автоматизация чрез индивидуални средства . . . . .	205
14.4.2 Автоматизация чрез интегрирани среди . . . . .	207
<b>15 Осигуряване на качеството на софтуера</b>	<b>217</b>
15.1 Проблемът за управление на качеството . . . . .	217
15.2 Компоненти на програмата за осигуряване на качеството .	218
15.2.1 Фактори . . . . .	218
15.2.2 Прегледи . . . . .	218
15.2.3 Оценяване . . . . .	219
15.2.4 Типове оценки . . . . .	221
15.2.5 Управление на конфигурацията . . . . .	222
15.2.6 Отчитане на грешките . . . . .	222



15.2.7	Анализ на тенденциите . . . . .	224
15.2.8	Проследимост . . . . .	225
15.2.9	Планиране на ПОКС . . . . .	225
15.2.10	Социални фактори . . . . .	225
<b>16</b>	<b>Зрялост на софтуерните процеси</b>	<b>227</b>
16.1	Методологията SEI CMM . . . . .	227
16.1.1	Същност и предназначение . . . . .	227
16.1.2	Структура . . . . .	228
16.1.3	Нива на зрялост . . . . .	228
16.1.4	Ключови области на обработка . . . . .	231
16.1.5	Цели . . . . .	232
16.2	Методологията BOOTSTRAP . . . . .	233
16.2.1	Същност, произход, преназначение . . . . .	233
16.2.2	База на BOOTSTRAP . . . . .	234
16.3	Стандарти за качеството на софтуера . . . . .	236
16.3.1	Национални и други стандарти с ограничено действие	236
16.3.2	Международни стандарти за софтуера . . . . .	237
16.3.3	Стандартите от серията ISO 9000 . . . . .	238
<b>17</b>	<b>Организационно управление</b>	<b>241</b>
17.1	Въведение . . . . .	241
17.2	Основни понятия . . . . .	241
17.3	Управление на софтуерни проекти . . . . .	242
17.3.1	Методология на управлението на проекти . . . . .	242
17.3.2	Управленческа структура . . . . .	244
17.4	Планиране . . . . .	245
17.5	Анализ и управление на риска . . . . .	247
17.6	Съставяне на графици . . . . .	248
17.7	Проследяване на проекта . . . . .	251
17.8	Примерен план на софтуерен проект . . . . .	254
17.9	Препоръчителни мениджърски практики . . . . .	255
17.10	Правила за софтуерни проекти . . . . .	256
<b>18</b>	<b>Човешкият фактор в софтуерното производство</b>	<b>261</b>
18.1	Наемане на добри софтуерни специалисти . . . . .	262
18.2	Определяне на структурата и състава на работните групи	264
18.3	Управление на взаимоотношенията в работната група . . . . .	266
18.4	Организиране на комуникациите в групата . . . . .	269
18.5	Организиране на сбирки и заседания . . . . .	270



18.6	Разработването на софтуера – teamwork . . . . .	271
18.7	Ергономика . . . . .	271
18.8	PM-CMM модел за управление на човешките ресурси . . .	272
18.9	Професионализъм и етично поведение . . . . .	273

## **V Бизнес и софтуер 277**

<b>19</b>	<b>Организационни структури за създаване на софтуера</b>	<b>279</b>
19.1	Въведение . . . . .	279
19.1.1	Същност на проблема . . . . .	279
19.1.2	Определения . . . . .	279
19.1.3	Класификация на институциите (структурите) . . .	280
19.2	Категории институции – базови характеристики . . . . .	280
19.2.1	Университети (научни институти) . . . . .	280
19.2.2	Фирми . . . . .	282
19.2.3	Консорциуми . . . . .	282
19.2.4	Движението за отворен код . . . . .	283
19.3	Институции и тяхната зависимост от различни влияния . .	285
19.3.1	Университети (научни институти) . . . . .	285
19.3.2	Фирми . . . . .	285
19.3.3	Консорциуми . . . . .	286
19.3.4	Движението за отворен код . . . . .	286
19.4	Характеристики на софтуера според категориите институции . . . . .	286
<b>20</b>	<b>Софтуерният бизнес – характерни особености</b>	<b>289</b>
20.1	Уникалност на софтуерния бизнес . . . . .	289
20.2	Регионални особености на софтуерния бизнес . . . . .	290
20.2.1	Европа . . . . .	291
20.2.2	Япония . . . . .	291
20.2.3	САЩ . . . . .	292
20.3	Стратегии на софтуерните фирми . . . . .	293
20.3.1	Продукти или услуги . . . . .	293
20.3.2	Целевият пазар . . . . .	295
20.3.3	Хоризонтално или вертикално сегментиране на пазара . . . . .	296
20.3.4	Постоянният поток от приходи . . . . .	297
20.3.5	Насоченост на фирмата – лидер, последовател или допълващ . . . . .	299

20.3.6	Характер на взаимоотношенията с другите субекти	299
<b>21</b>	<b>Еволюция на малката софтуерна фирма</b>	<b>303</b>
21.1	Основа на изследването	303
21.2	Типове ноу-хау на стартиращата софтуерна фирма	304
21.3	Опростена схема на развитие на малката софтуерна фирма	305
21.4	Един по-диференциран модел на развитие	306
21.5	Еволюция на конфигурация С1	308
21.6	Еволюция на конфигурация С5	309
21.7	Извод	310
<b>22</b>	<b>Правни проблеми на софтуера</b>	<b>311</b>
22.1	Необходимост от специализирана правна култура	311
22.2	Закопи за интелектуалната собственост	312
22.3	Правна защита на софтуера	313
22.4	Защита на софтуера чрез авторското право	314
<b>23</b>	<b>Маркетинг на софтуера</b>	<b>319</b>
23.1	Общи съображения	319
23.2	Определения за маркетинг	320
23.3	Основна маркетингова концепция	320
23.4	Маркетингова стратегия	321
23.4.1	Целеви пазар	321
23.4.2	Маркетингов микс	324
23.5	Пазарен жизнен цикъл	327
23.5.1	Пазарен жизнен цикъл на разработването на продукта	327
23.5.2	Пазарен жизнен цикъл на готовия продукт	328
23.6	Позициониране и марка	330
23.7	Моделиране на софтуерния пазар	331



## Увод

Настоящата книга е преработено издание на същото заглавие, което публикувахме през 2001 година в издателство „Анубис“. Доколкото основната ѝ цел беше да се ползва като учебник от студентите във висшите учебни заведения, очакванията ни се сбъднаха и тиражът беше изчерпан за около две години. Предмет на настоящата книга е това, което на английски език се нарича **Software Engineering** вече над три десетилетия. Преводът на български не може да бъде буквален, най-малкото защото „инженерство“ и „Engineering“ смислово не се покриват. Немското название – **Software Entwicklung** или пък френското – **Génie Logiciel** могат да ни насочат, но не и да ни дадат крайното решение. Задачата се усложнява и от разнообразието от значения, които различните автори влагат в термина. Теоретикът е склонен да остава около езиците за спецификация, верификацията и свързаните с тях формализми, практикът – да търси годни за прилагане модели на жизнения цикъл, ефективни методи за осигуряване на качество, за правилно планиране и организиране на процеса на разработка, ръководителят стига в интересите си до проблемите на софтуерния маркетинг и методите за оценяване на необходимите ресурси. Тези и много други съображения са ни довели (след 2-3 по-несполучливи опита) до българския термин **Софтуерни технологии**.

Първият у нас курс по Софтуерни технологии беше подготвен и четен от Аврам Ескенази и Владимир Занев в продължение на три години, започвайки от 1984/85 учебна година, за студентите от IV и V курс на Факултета по математика и информатика на Софийския университет. От 1987/88 учебна година след известна преработка той стана задължителен за студентите в това учебно заведение и в четенето на лекциите и упражненията се включиха Нели Манева и Валя Петрова. С известни модификации този курс продължава и до днес да се чете там. Междувременно, най-напред в Икономическия университет – Варна (с лектор Георги Зеленков), а по-късно в Пловдивския университет, International

University, Нов български университет и на други места започнаха и продължават лекции по този предмет, макар и не винаги под същото име, а понякога и в рамките на повече от един курс.

Невъзможно е една книга да обхване целия предмет, още по-малко възможно е той да се прочете в рамките на един курс. Като пример ще посочим, че в някои университети в САЩ (Тексаски университет, Университетът в Сиатъл и др.) обучението по софтуерни технологии продължава две години в рамките на 6-8 задължителни и още толкова избираеми курса, практикум и 2-3 курсови проекта. Отскоро във Факултета по математика и информатика на Софийския университет започна подобна магистърска програма по Софтуерни технологии. Съответна на тази констатация е и целта на авторите – макар да е предназначена да бъде учебник за студентите по предмета Софтуерни технологии, настоящата книга би могла да бъде за тях и за всички практикуващи софтуерни специалисти и ръководители въведение в основополагащите направления на тази изключително динамична и с нарастващо за практиката значение дисциплина.

Главите на тази книга са разработени както следва:

Аврам Ескенази: 1-3, 7, 10-12, 15-16, 19-23.

Нели Манева: 4-6, 8-9, 13-14, 17-18.