|  |
| --- |
| CodePro AnalytiX – анализ и разбиране на големи софтуерни продукти на Java |
|
| *Проект по „Софтуерни технологии“, 2015 г.* |
| Факултет математика и информатика, Софийски университет |

|  |
| --- |
| Изготвили: |
| *80866*, *Стефан Василев Василев*, *thesopa@abv.bg (имейл)*  *Компютърни науки, втори курс, втори поток, група 8* |
| *фак.номер*, *три имена на студента*, *електронен адрес (имейл)*  *специалност, курс, поток, група* |
| *фак.номер*, *три имена на студента*, *електронен адрес (имейл)*  *специалност, курс, поток, група* |
| *фак.номер*, *три имена на студента*, *електронен адрес (имейл)*  *специалност, курс, поток, група* |

Ръководител: доц. Димитър Биров

2015 г.

Съдържание

[1 Увод 3](#_Toc418974213)

[1.1 Тема на проекта 3](#_Toc418974214)

[1.2 Цели на проекта 3](#_Toc418974215)

[1.3 Резултати 4](#_Toc418974216)

[2 Описание 4](#_Toc418974217)

[2.1 Изисквания към софтуера и инсталиране 5](#_Toc418974218)

[2.1.1 Инсталация 5](#_Toc418974219)

[2.2 Раздел „Анализ и дизайн“ 7](#_Toc418974220)

[2.3 Раздел „Реализация“ 7](#_Toc418974221)

[3 Форматиране на таблици и фигури 7](#_Toc418974222)

[3.1 Форматиране и подреждане на фигури 8](#_Toc418974223)

[3.1.1 Под под раздел 9](#_Toc418974224)

[3.1.2 Под под раздел 9](#_Toc418974225)

[3.2 Форматиране на програмен код 10](#_Toc418974226)

[3.2.1 Програмен текст като изображение 10](#_Toc418974227)

[3.2.2 Програмен код като текст 10](#_Toc418974228)

[3.3 Предаване на проекта 11](#_Toc418974229)

[3.3.1 Под под раздел 11](#_Toc418974230)

[3.3.2 Под под раздел 11](#_Toc418974231)

[3.4 Защита на проекта 12](#_Toc418974232)

[Библиография 12](#_Toc418974233)

# Увод

С напредъка на софтуерните технологии се появяват все повече и повече големи продукти. Те сами по себе си съдържат изключително много файлове, пълни с програмен код и зависимости, които е трудно да бъдат съобразени от обикновения програмист. Ето защо на помощ идват инструменти за автоматизирана проверка на коректността на продукта.

## Тема на проекта

В този проект ще разгледаме конкретния инструмент CodePro AnalytiX. С помощта на CodePro AnalytiX можем лесно да управляваме софтуерните си продукти, тъй като той предлага набор от средства за верификация на програмния ни код, както и коректността на зависимостите между файловете ни. По този начин се спестява изключително много време и усилия на софтуерния разработчик. Ето защо е удачно инструмента да се използва за особено големи проекти, в които има ясно установени срокове за изпълнение на конкретни задачи. Освен това автоматизацията на процеса на проверка на програмния код предотвратява възможността за допускане на тривиални грешки, като объркани имена на файлове, пакети и други, които често човек може да обърка при невнимание.

## Цели на проекта

На първо място целта е да се научим да работим в екип, всеки от нас да поема отговорност за собствената си задача, но да не забравя да се консултира за това и с останалите членове на екипа. Освен това повечето от нас са стажанти в момента и всеки ден се сблъскваме с големи проекти и голямо количество програмен код. Ето защо никога не е излишно да се запознаем с още един добър инструмент за анализ на код и управление на големи проекти. Сложните алгоритми, които стоят зад един инструмент за анализиране на код също представляват голям интерес и могат да ни бъдат полезни в бъдещи проекти. И на последно място в крайна сметка е добре да обогатяваме общата си култура в сферата на готовите софтуерни продукти и плъгини, които могат да ни бъдат в ползва когато сами правим своите проекти. С усилията, които положихме да се запознаем с CodePro AnalytiX научихме и доста странични и полезни неща.

## Резултати

С присъствие на презентацията на този проект, зрителят може да:

* Се запознае с плюсовете, които ни дава възможността да анализираме кода си автоматично.
* Се убеди, че инструментите за анализ на код са изключително полезни за големи софтуерни продукти.
* Се научи да инсталира инструменти за анализ на код, както и други различни плъгини в редактора Eclipse.
* Се запознае по-подробно с инструмента CodePro AnalytiX. С неговата функционалност, възможности, кога и как е удачно да бъде използван от софтуерния разработчик.
* Види конкретно приложение на CodePro AnalytiX.

Главната методология за съставяне на проекта беше проучване в интернет за материали, видео, указания за работа с продукта. Също така четене, тестване и разбиране на документацията на CodePro AnalytiX.

# Описание

Като всеки софтуерен продукт и CodePro AnalytiX се нуждае от определена конфигурация на компютъра, както и от редица други зависимости и особености, които трябва да изпълнява нашата машина, за да може да бъде инсталиран.

## Изисквания към софтуера и инсталиране

На първо място компютърът ни трябва да има инсталирана операционна система. Тя може да бъде Windows или Linux.

### Инсталация

#### Стъпка 1 – Сваляне на редактора Eclipse

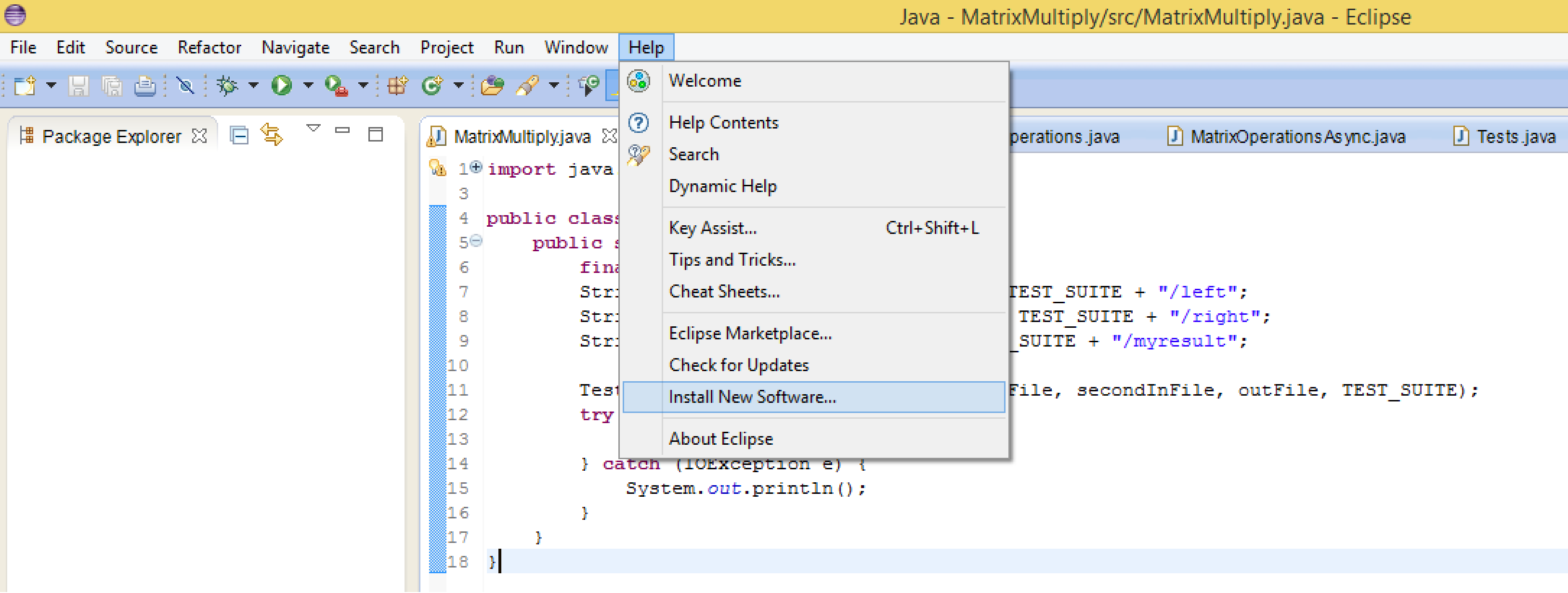
След като има операционна система потребителят трябва да изтегли версия на Eclipse за Java разработчици (препоръчително най-новата) от официалния сайт:

http://www.eclipse.org/downloads/

#### Стъпка 2 – Инсталиране на CodePro AnalytiX

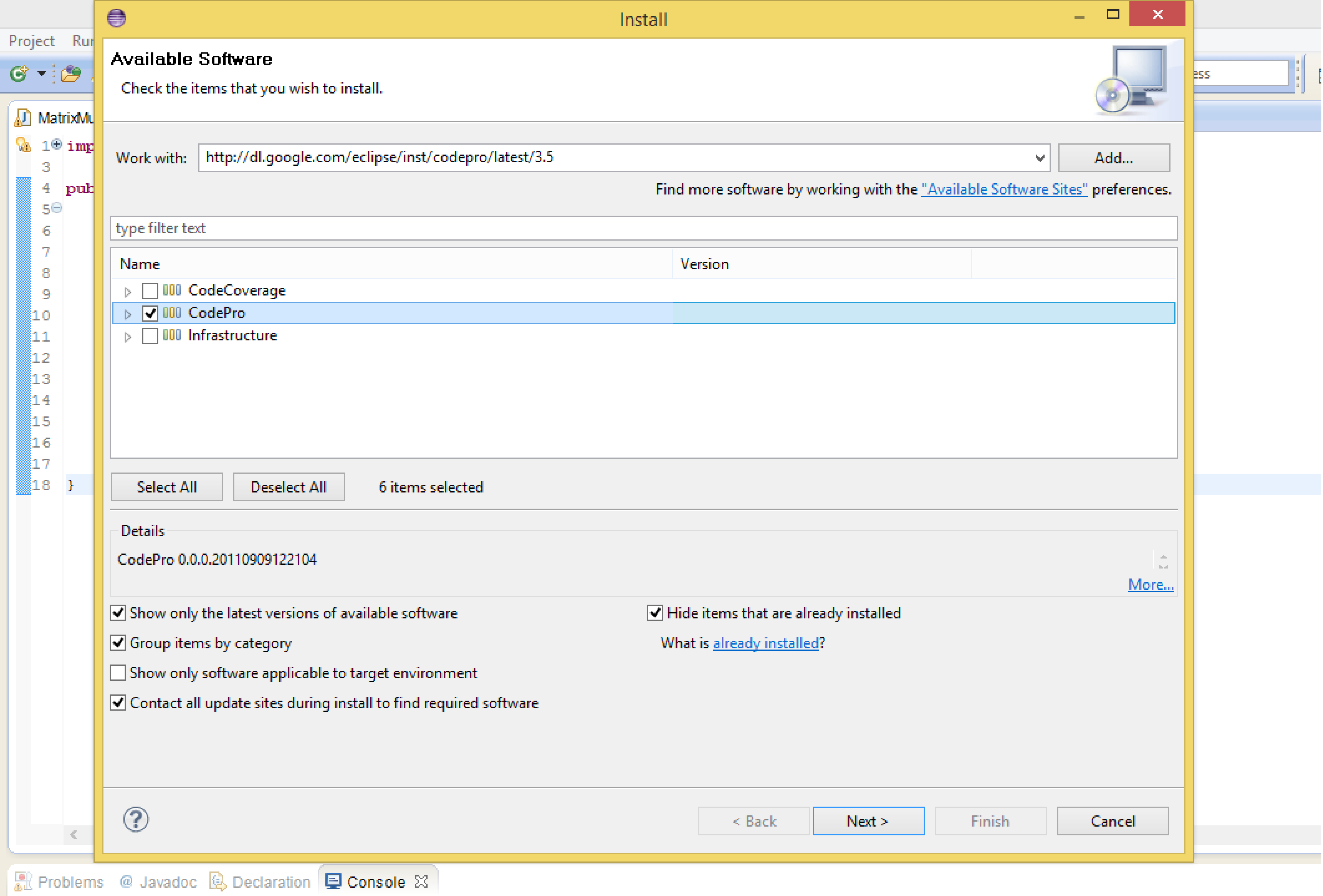
След инсталация на редактора Eclipse вече сме готови да инсталираме CodePro AnalytiX. Тъй като платформата на Eclipse работи главно с плъгини, самия редактор предоставя лесен начин за инсталиране на нови плъгини.

След като стартираме Eclipse, можем лесно да се навигираме до падащото меню Help -> Install New Software…



**Фиг.1** *Инсталиране стъпка 1*

В резултат на това се отваря прозорец, в който трябва да въведем линк, от който Eclipse ще може да свали CodePro AnalytiX. Тъй като има значение коя версия на редактора сме изтеглили, трябва да отидем на официалния сайт на CodePro и да вземем линк за сваляне на плъгина, съответстващ на нашата версия на Eclipse. След като въведем адреса за сваляне в прозореца имаме няколко опции. Чекваме опцията CodePro:



**Фиг.2** *Инсталиране, стъпка 2*

След това кликваме два пъти Next, прочитаме и се съгласяваме с условията за използване на плъгина и избираме Finish. Изчакваме известно време, след което рестартираме Eclipse (инсталацията сама ще ни помоли да го направим), за да могат промените да влязат в действие.

## Раздел „Анализ и дизайн“

Следващ раздел в проекта е анализа и дизайна. Съгласно описаните в предния раздел сценарии и начини на изпозлван ена софтуера е добре да бъдат описани съотвтстващият им дизайн.

В този раздел се очакват д абъдат дадени и съответните UML диаграми, които спомагат за да се разбере идеите, които са реализирани от екипа в зададения им за разработка по проекта софтуер.

## Раздел „Реализация“

Ако изрично не е посочено в заданието ви реализацията на проекта не е задължителна. Разбира се ако е направено носи бонус за екипа!

По-важно е да бъдат описани какви са избраните/предложени технологии з ареализация. Важно е д аопишете мотива за избор на тези технологии.

Очаква се и описание на основните модули които съставят изпълнимите единици по проекта и ак те са разпределени върху съответния хардуер. За целта можете д аизпозлвате отново съответните диаграми на UML.

# Форматиране на таблици и фигури

Ако използвате таблици по-долу е даден вида и формата на примерна таблица. Таблиците се именуват със центриран текст под тях, който съдържа кратко описание на предназначението на таблицата.

Пример:

| Изисквания | Описание |
| --- | --- |
| **Работоспособност** |  |
| **Сигурност** |  |
| **Използваемост** |  |
| **Тестируемост** |  |
| **Еволюция** |  |
| **Адаптация и самоадаптация** |  |

**Таб. 1**. *Таблицата представлява пример за оформяне на таблица*

Номерацията на таблиците се прави последователно в нарастващ ред на номерата като се започне от 1.

## Форматиране и подреждане на фигури

Използването на фигури, диаграми и схеми изисква те да бъдат номерирани и именувани по следния начин:



**Фиг. 1**. *Фигура, демонстрираща номерацията и формата*

Номерацията на фигурите се прави последователно в нарастващ ред на номерата като се започне от 1.

Фигурите се центрират в текстта. Около тях не с епоставя текст. Текстта предшества или следва фигурата.

### Под под раздел

Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф

#### Под под под раздел

Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф

#### Под под под раздел

Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф

### Под под раздел

Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф

#### Под под под раздел

Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф

#### Под под под раздел

Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф

## Форматиране на програмен код

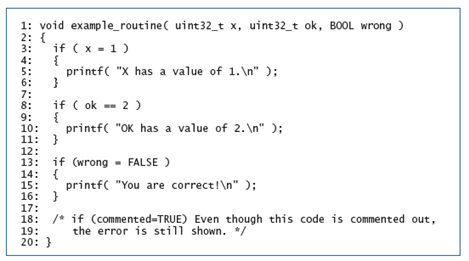
В есето се налага понякога да се добави програмен код или друг вид програмен текст – скриптов език, XML, HTML, CSS и др.

Начините на форматиране са следните:

* Поставяне на програмния код/текст като изображение
* Поставяне на програмния код/текст като текст съ сспециализиран формат

### Програмен текст като изображение

В този случай програмният код се форматира каакто фигура с изключение на названието. Пример:



**Листинг. 1**. *Грешки в оператор if по отношение на оператор за присвояване и операцията за равенство*

### Програмен код като текст

Програмния код се форматира със стила .... А под него се поставя отново означение с Листинг.

Пример:

if (x==y) {

x = x + y;

}

**Листинг. 2**. *Грешки в оператор if по отношение на оператор за присвояване и операцията за равенство*

## Предаване на проекта

Проекта се предава в електронен вид след щателен оглед за спазване на форматирането на текстта съгласно изикванията на този темплейт по електронната поща. След проверка и коментар от страна на проверяващия проекта се коригира, допълва или се разработват допълнително негови отделни елементи. След което се изпраща отново по електронната поща в екетронен формат на Microsoft Word.

### Под под раздел

Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф

#### Под под под раздел

Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф

#### Под под под раздел

Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф

### Под под раздел

Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф

#### Под под под раздел

Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф

#### Под под под раздел

Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф Това е текст формиран в цялостен параграф

## Защита на проекта

Проектът се защитава чрез презентация. Презентацията се приготвя предварително чрез използване на Microsoft PowerPoint или друг подходящ софтуер за създаване н презентации.

За представяне на проекта чрез подготвената презентация се отделят 10 минути в които всеки един от екипа прави представяне на частта от проекта, която е разработил.

По време на презентацията се говори по слайдовете. НЕ се позволява да се чете от допълнителни материали – листове, телефон, таблет и др.

# Библиография

Запова на нова страница.

Библиографията се формира по следния начин – цитиран в начупени скоби [ ] се поставя следния текст:

[Asi02] Nadim Asif, “Architecture Recovery”, In the Proc. of International Conference of Information and Knowledge Engineering (IKE’02), lasVegas, 2002.

[Gar00] D. Garlan, “Software Architecture: A Roadmap,” Proc. Int’l Conf. Software Eng., pp. 91-101, 2000.

Ако автора е един се вземат първите три букви н фамилията след това се слагат двете последни цифри от годината на излизане на публикацията

Ако авторите са двам или трима се взимат първите им букви на фамилиите и се поставят последните две цифри от годината на излизане на публикацията

[LB85] M. Lehman and L. Belady, Program Evolution: Processes of Software Change. London Academic Press, 1985.

[JRL00] M. Jazayeri, A, Ran, F. van der Linden, Software Architecture for Product Families Principles and Practice, Addison Wesley, 2000.

[GAO09] D. Garlan, R. Allen, and J. Ockerbloom, “Architectural Mismatch: Why Reuse Is So Hard,” IEEE Software, vol. 26, no. 4, pp. 66-69, Nov. 2009.

Ако авторите са повече от трима се взимат първите букви от фамилните имена на първите двама автори, след това се поставя + и след това последните две цифриот годината на издаване н апубликацията

[KM+06] J. Knodel, D. Muthig, M. Naab, and M. Lindvall, “Static Evaluation of Software Architectures,” Proc. Conf. Software Maintenance and Reeng., pp. 279-294, 2006.

[GP+09a] Joshua Garcia , Daniel Popescu , George Edwards , Nenad Medvidovic, Identifying Architectural Bad Smells, Proceedings of the 2009 European Conference on Software Maintenance and Reengineering, p.255-258, March 24-27, 2009 [doi>10.1109/CSMR.2009.59]

[GP+09b] Joshua Garcia, Daniel Popescu, George Edwards and Nenad Medvidovic. Toward a Catalogue of Architectural Bad Smells. Proceedings of the Fifth International Conference on the Quality of Software Architectures (QoSA 2009), East Stroudsburg, Pennsylvania, USA, June 2009.

Ако две публикации с един и същи автори в една и съща година има няколко две три и повече след цифрите на годината се добавя последователно a,b,c и т.н. както по-горе

Примери:

[Hol01] R. Holt, “Sofware Architecture as a Shared Mental Model,” Proc. ASERC Workshop Software Architecture, 2001.

[MEG03] N. Medvidovic, A. Egyed, and P. Gruenbacher, “Stemming Architectural Erosion by Architectural Discovery and Recovery,” Proc. Int’l Workshop from Software Requirements to Architectures, 2003.

[MMP00] N. R. Mehta, N. Medvidovic, and S. Phadke. Towards a taxonomy of software connectors. In Proc. of the 22nd Int. Conf. on Software Engineering, 2000.

[PW92] D.E. Perry and A.L. Wolf, “Foundations for the Study of Software Architecture,” ACM SIGSOFT Software Eng. Notes, vol. 17, no. 4, pp. 40-52, 1992.

[Riv04] C. Riva, “View-Based Software Architecture Reconstruction,” PhD thesis, Technical Univ. of Vienna, 2004.

[TMD08] R. Taylor, N. Medvidovic, and E. Dashofy. Software Architecture: Foundations, Theory, and Practice. John Wiley & Sons, 2008.

[WPM10] David Woollard, Chris A. Mattmann, Daniel Popescu, and Nenad Medvidovic. KADRE: Domain-specific architectural recovery for scientific software systems, 25th IEEE/ACM International Conference on Automated Software Engineering, 2010.

[MK96] N.C. Mendonc¸a and J. Kramer, “Requirements for an Effective Architecture Recovery Framework,” Proc. Int’l Software Architecture Workshop, pp. 101-105, 1996.

[OSV02] L. O’Brien, C. Stoermer, and C. Verhoef, “Software Architecture Reconstruction: Practice Needs and Current Approaches,” Technical Report CMU/SEI-2002-TR-024, Carnegie Mellon Univ., 2002.

[DP09] Stephane Ducasse, Damien Pollet, "Software Architecture Reconstruction: A Process-Oriented Taxonomy," IEEE Transactions on Software Engineering, vol. 35, no. 4, pp. 573-591, July/August, 2009

[Deu02] Arie van Deursen, Software architecture recovery and modeling: [WCRE 2001 discussion forum report], ACM SIGAPP Applied Computing Review, v.10 n.1, p.4-7, Spring 2002 [doi>10.1145/568235.568236]

[RA07] Ghulam Rasool, Nadim Asif, Software Architecture Recovery, World Academy of Science, Engineering and Technology 4 2007

Цитиране на ресурси от интернет страници става с помощта на съкратено наименование до 5 символа

[ECLPS] https://www.eclipse.org/ (последно използвана на 04.2014)

Текстта на линка запазете в черен цвят – изтрийте по възможност хиперлинка на връзката. В кръглите скоби поставете датата на последен достъп на връзката от ваша страна. Връзката след време може и да не е валидна или преместена.