<Software System for Store Chain>

Речник

Версия <1.0>

История на промените

|  |  |  |  |
| --- | --- | --- | --- |
| **Дата** | **Версия** | **Описание** | **Автор** |
| 03.12.2013 | 1.0 | Създаване на документа | Т. Колева |
|  |  |  |  |

Съдържание

1. Въведение 4

1.1 Цел 4

1.2 Какво обхваща този речник 4

1.3 Препратки 4

1.4 Преглед 4

2. Акроними/абревиатури 4

3. Термини 6

4. UML стереотипи 12

4.1 Structure diagrams (Структурни диаграми) 12

4.2 Behavior diagrams (Диаграми на поведение на системата) 13

Речник

# Въведение

Този раздел запознава читателя със съдържанието на речника на използваните термини, съкращения и абревиатури в документацията на „Информационна система за верига от магазини”.

## Цел

Целта на речника е:

* да запознае възложителя на проекта с използваните термини, акроними и абревиатури
* да поясни значенията на тези термини, акроними и абревиатури
* да осигури по-лесно/ясно общуване между възложителя и изпълнителя на проекта

## Какво обхваща този речник

Този речник обхваща всички използвани термини, акроними и определения в документацията на проекта „Информационна система за верига от магазини”.

## Препратки

Този речник дефинира абревиатури и термини използвани в следните документи:

* Quality Assurance Plan
* Software Development Plan
* Software Requirements Specification
* Deliverables-List
* Project-Schedule

## Преглед

Този речник има следната организация:

* *Въведение*, което запознава читателя с неговата цел.
* *Акроними/абревиатури*, тук се поясняват значенията на използваните съкращения в документацията на проекта.
* *Термини,* тук се поясняват значенията на термините използвани в документацията на проекта.

# Акроними/абревиатури

* **ГВРС -** Гарантирани вземания на работниците и служителите.
* **ДОД** - Данък върху общия доход.
* **ДОО** - Държавно обществено осигуряване.
* **ЕГН -** Единен граждански номер.
* **МОД -** Минимален осигурителен доход.
* **НАП** – Национална агенция по приходите.
* **ТЗПБ** - Трудова злополука и професионални болести.
* **BA (Business Analyzer)** – Бизнес анализатор: човек, който анализира съществуващите или идеални организации и дизайна на системи, включващи предприятия, ведомства и различни организации. Бизнес анализаторът оценява бизнес моделите и тяхната интеграция с технологиите.
* **c (completed)** – Завършен (замразен) – Това е вид състояние на даден документ в даден момент от изпълнението на проект според **RUP** методологията.
* **C1...C4 (Construction) –** Изграждане на проекта.
* **Design Patterns** - Шаблони за дизайн. Тази концепция предлага стандартни решения за архитектурни и концептуални проблеми в компютърното програмиране.
* **E1, E2, E3 (Ellaboration) –** Детайлизиране на проекта**.**
* **e (evolved)** – В процес на разработка – Това е вид състояние на даден документ в даден момент от изпълнението на проект според **RUP** методологията.
* **I1 (Inception 1) –** Планиране. Начална фаза. Първо предаване на документи.
* **Maven –** система за управление на софтуерни проекти.
* **MVC (Model-View-Controller) -** е архитектурен шаблон за дизайн в програмирането, основан на разделянето на бизнес логиката от графичния интерфейс и данните в дадено приложение.
* **Java EE (Java Platform, Enterprise Edition) –** Набор от спецификации. Платформата предоставя API среда за изпълнение, разработването и управлението на бизнес софтуер , включително мрежови и уеб услуги, както и на други мрежови приложения .
* **Java Database Connectivity (JDBC)** - интерфейс за приложно програмиране (API) за езика за програмиране Java, който определя как един клиент може достъп до базата данни.
* **JIRA** - Уеб-базирана система за проследяване на грешки (bug tracking), проблеми (issue tracking) и управление на разработката на софтуерни проекти. Управлението на технологичния процес (workflow) прави JIRA подходяща за управление и подобряване на процеси.
  + **PM (Project Manager)** - Проект мениджърът е професионалист в областта на управлението на проекти. Ръководителите на проекти могат да имат отговорностите да планират, изпълняват и закриват всеки проект, обикновено свързан с най-различни сфери на дейност, в случая разработка на софтуер.
  + **POS** – система, която се грижи за въвеждане на удобство при работа на касиерите, ускоряване на процеса на обслужване и подобряване на отчетността. Осигурява функции, които помагат при работата на продавачите.
  + **r (revised)** – Ревизиран (актуализиран след приемане) – Това е вид състояние на даден документ в даден момент от изпълнението на проект според **RUP** методологията.
* **RUP (Rational Unified Process)** – Итеративна методика на процеса на разработване на софтуер създадена от Rational Software Corporation, подотдел на **IBM (International Business Machines Corporation** - американска мултинационална корпорация за компютърни технологии и консултации**)** от 2003 г. насам. **RUP** не е един точно предписан процес, а по-скоро адаптивна методика, предназначена да бъде приспособена от разработващите организации и софтуерните екипи на проекти, които трябва да изберат кои са елементите на методиката, които са подходящи за техните нужди.
* **s (stable)** – Приет (утвърден) – Това е вид състояние на даден документ в даден момент от изпълнението на проект според **RUP** методологията.
* **SSSC (Software System for Store chain) -** Информационна система за верига от магазини.
* **Test-driven development (TDD) -** Разработката чрез тестове, е метод за разработка на софтуер, при който се спазва следният работен цикъл: първо се пишат тестови варианти (test cases), които да покрият изискванията за новия изходен код, а чак след това се пише програмния код така, че да покрива тези тестове.
* **T1 (Transition) –** Окончателно предаване на проекта.

# Термини

* **Customer audits** – външен клиент, който използва системата.
* **Database** – Това е организирана съвкупност от данни. Данните обикновено са организирани така, че да могат да моделират най-различни аспекти от реалността, по начин, който позволява да съществуват процесите, изискващи тази информация. Терминът база данни се отнася за самите данни и техните структури от данни.
* **Eclipse** – Представлява среда за разработка на софтуер, състояща се от базова работна среда и разширяема плъг-ин система за персонализация на средата. Написана е главно на **Java** и се използва за разработката на приложения на **Java**.
* **External system** – Представлява външна система, която ще комуникира с нашата система.
* **Interface** – Представлява средствата за комуникация между потребител и компютър чрез периферни устройства, като монитор, клавиатура, принтер, скенер и други, или връзката с Интернет посредством интернет протокола (**IP**), или всякакъв друг вид комуникация свързана с компютър или подобна система.
* **Microsoft (Microsoft Corporation) –** Американска мултинационална софтуерна компания, която разработва, произвежда, лицензира и поддържа широка гама от продукти и услуги свързани с компютрите. **Microsoft** е най-големият производител на софтуер, измерен според приходите, и една от най-ценните компании в света.
* **Microsoft Office** – Представлява офис пакет на десктоп приложения, сървъри и услуги за операционни системи Microsoft Windows и OS X, въведени от Microsoft.
* **unit тестване** – Метод, при който отделни части на сорс кода, група от един или повече компютърни програмни модули заедно със свързаните с тях контролни данни, оперативни процедури и други, се тестват за да се определи дали са подходящи или годни за използване.

**А**

* **автоматично архивиране** - Архивирането на информация или данни от базата данни,което се извършва автоматично през точно определен период от време или на точно определена дата (и/или час).
* **архивиране на информация/данни** – Архивирането на информация или данни от базата данни през определен период от време или поради определени обстоятелства.

**Б**

* **бъгове** – Грешки.

**В**

* **валидация -** Да се установи дали едно обстоятелство има сила, дали е действително.
* **вендор** – Продавач.
* **верификация** – Верификацията на софтуер е процес, чиято цел е да се гарантира, че софтуерът отговаря напълно на очакваните изисквания. Има два основни подхода за верификация: динамична верификация, известна още като тестване или експериментиране (много добър подход за намиране на грешки и бъгове) и статична верификация, известна още като анализ (полезен подход за доказване на коректността на дадена програма, въпреки, че може да доведе до неверни положителни резултати).
* **Визия (Vision)** – Документ, в който са събрани, анализирани и определени нуждите и характеристиките от високо ниво на една система. Документа се фокусира върху възможностите за поръчителите, потребителите, към които се стремят, както и защо тези нужди съществуват.
* **визуализация** – Визуално представяне на информация, данни или други.
* **внедряване** – Въвеждане на дадена система в употреба.
* **външни системи** – Системи, които нямат никакви физически връзки с информационната система на университета, системи които функционират самостоятелно и могат да бъдат интегрирани със информационната система на университета.

**Г**

* **генериране** – Автоматично създаване на определени обекти или детайли на дадена програма или софтуер.

**Д**

* **диаграма** - Представлява двуизмерно геометрично символно представяне на информация, в зависимост от различните техники за визуализация.
* **дизайн модел** – Представлява обектен модел, който описва реализацията на потребителските случаи и служи за извеждане на Модела на имплементацията и неговия програмен код. Дизайн моделът е важен входен артефакт за дейностите по имплементация и тестване.

**Е**

* **екселски файл (еxcel file)** – Файл, в който информацията се записва най-често във формата на таблици (табличен вид).

**И**

* **имплементация** – Отнася се за процеса след покупката на даден продукт, при който се ръководи потребителя от момента на поръчката на софтуер или хардуер до момента на неговото използване. Този процес включва анализ на изискванията, обхвата, персонализацията, интеграцията на системи, потребителските правила и условия, обучението на потребителите и доставката.
* **история на ревизиите** – Описва направените промени във времето върху даден документ, като описва какви са промените, датата, на която е направена всяка промяна, както и човекът, който е направил промяната и понякога причината или причините за тази промяна.
* **итерация** – Процес на повтаряне на определен процес с цел достигане на желаната цел или резултат. Всяко повторение на процеса се нарича итерация и резултатите от една итерация обикновено се използват като отправна точка за следващата итерация.

**К**

* **код дизайнер** - Разработчик с голям опит, който участва в изготвянето на дизайна на отделните модули.

**Л**

* **логване –** Влизане, отчитане.

**Н**

* **нефункционални изисквания** – Нефункционалните изисквания са изисквания, които определят критериите за работата на системата от хардуерна гледна точка.

**О**

* **одит –** инвентаризация.
* **онлайн среща** – Среща на членовете на екипа посредством Интернет връзка и програми или приложения като Skype.

**П**

* **парола** – Последователност от букви и/или цифри, използвани (обикновено заедно с потребителско име) за вход в дадена система.
* **Поръчител/възложител** – Човек, който възлага на някого определена задача или проект за изпълнение.
* **потребителски графичен интерфейс** – Това е вид потребителски интерфейс, който служи за комуникация на потребителите с електронни устройства или компютри, използвайки повече изображения, отколкото текстови команди.
* **потребителски изисквания** – Това са изискванията, които потребителя очаква от софтуера, които трябва да бъдат изпълнени при изграждането на софтуерния проект.
* **потребителски профил** – Профил, който се създава в дадена система, в който се съдържа лична и друга информация свързана с конкретния потребител.
* **права за достъп** – Тези права определят до какви ресурси и/или информация има достъп определено лице и какви действия може да извършва с тях.
* **програмен език** – Това е изкуствено създаден език, предназначен за комуникация на инструкциите и машината, и по-точно компютъра. Езиците за програмиране могат да се използват за създаване на програми, които контролират работата на машината и / или да изпълняват алгоритми много прецизно.
* **програмен интерфейс (API)** – Протокол предназначен да служи като интерфейс за комуникация между различни софтуерни компоненти. **API** може да се разглежда като библиотека, която може да включва спецификации за рутинност, структури от данни, обектни класове и променливи.
* **програмист** – Човек, който пише или създава компютърен софтуер. Програмист може да означава специалист в дадена област на компютърното програмиране или пък общото значение на програмист, който пише код за различни видове софтуер.
* **Програмна среда** – Изпълнява основните задачи на компютърния език. Всеки компютърен език изпълнява различни форми на средата без значение дали езикът е компилатор, интерпретатор или някакъв друг тип, или пък се изпълнява чрез приложния програмен интерфейс (**API**).
* **протокол за среща** – Протокол, в който се описват действията извършени по време на срещите на членовете на един екип и евентуалните задачи, които са раздадени на членовете, както и резултати от свършени задачи.

**Р**

* **регистриран потребител –** Човек, който използва приложение, програма или уеб сайт за да предостави своите идентификационни данни, с което да докаже своята самоличност.
* **редактиране** – Редакция или промяна на документ ,файл или информация.
* **ръководител на проект** – Ръководителят или мениджърът на проект, е професионалист в областта на управлението на проекти. Ръководителите на проекти могат да имат отговорностите да планират, изпълняват и закриват всеки проект, обикновено свързан с най-различни сфери на дейност, в случая разработка на софтуер.

**С**

* **системен администратор** - Системния администратор е лице, наето да поддържа и експлоатира компютърна система и/или мрежата.
* **софтуерен архитект** – Компютърен програмист, който прави избора на дизайн от високо ниво и взема решения какви технологични стандарти, включително и софтуерни стандарти за кодиране, инструменти и платформи, ще се използват за създаването на даден софтуер или друг подобен продукт.
* **софтуер за разработка на база данни** – Специализиран софтуер за създаване и управляване на бази от данни, включващ специални команди и функции специфични единствено за работа с бази данни.
* **софтуер за тестване** Софтуер, позволяващ провеждането на различни тестове върху системата или части от нея.
* **софтуерна архитектура** – Обозначава структурата от високо ниво на една софтуерна система. Тя може да се дефинира като набор от структури, необходими за софтуерната система, които включват софтуерни елементи, връзките между тях, както и свойствата на елементи и техните отношения. Терминът софтуерна архитектура също така означава набор от практики, използвани да избор, определяне или създаване на дизайна на софтуерната архитектура.
* **софтуерни изисквания** – Установява потребностите на поръчителя, които трябва да бъдат реализирани от софтуера. Изискванията представляват състояния или способности, които трябва да се изпълнят или които системата или даден системен компонент трябва да притежава, за да удовлетвори даден договор, стандарт, спецификация или друг официално наложен документ.
* **спецификация** – Спецификациите са математически базирани методи, чиято цел е да подпомогнат имплементирането на системи или софтуер. Използват се за описание на системата, за анализиране на поведението й, и да се насочи към нейния дизайн, проверявайки важни характеристики с помощта на мощни и ефективни инструменти.
* **състояние (State)** – Определен вид състояние на документ или обект в даден момент от време.

**Т**

* **тестер** - Тестерът е отговорен за основните дейности по тестване на системата и верифициране на нужното й качество.
* **тест мениджър** – Мениджър на екипа по тестване на системата.
* **тестов модел** - Тестовият модел обяснява какво и как ще бъде тествано. Той представлява разрез на моделите на бизнес процесите, на дизайна и имплементацията, с който са описани самите тестове и тези аспекти на тестваните артефакти, които са от значение за тестовия процес. Тестовият модел включва набор от тестови случаи, тестови процедури и тестови скриптове, както и описание на очакваните резултати от тестовете и на взаимовръзките помежду им. Трябва да съдържа също таблици на съответствието бизнес процеси – системни потребителски случаи – компоненти на системата - тестови случаи и тестови сценарии.

**Ф**

* **фаза Детайлизиране (Elaboration)** – Изграждат се бизнес модела, чрез анализ на потребителските изисквания, диаграми на бизнес процесите, състоянията и потребителските случаи. Избира се софтуерна архитектура и също така се изграждат и тестовия модел, моделът на данните и дизайн модела.
* **фаза Изграждане (Construction)** – В тази фаза основната цел е изграждането на системата. В началото се изграждат общите/базовите функции на системата. След това се изграждат функционалности, отговарящи за конкретните изисквания. Тази фаза приключва с пълно тестване на потребителските изисквания.
* **фаза Планиране (Inception)** – В тази фазата се изграждат визията и планът за изграждане на проекта. Определя се и организационната схема, състава и отговорностите на членовете на екипа. В тази фаза се събират основните функционални и нефункционални изисквания.
* **фаза Предаване (Transition)** – В тази фаза се позволява на бенефициента да извърши тестване за приемането на системата, както и да се извърши прехода към новата система. За целта е необходимо и провеждането на съответното обучения на персонала.
* **функционален поглед** – Цялостен поглед над системата от гледна точна на нейната функционалности.
* **функционален тест** – Това е тест, който се извършва от екипа по качеството и е тест тип „тестване на черна кутия”, който основава своите тестови случаи на спецификациите на компонента по време на тестването. Функционалностите се тестват като им се подават входни данни и се изследват изходните данни, като вътрешната структура на програмата рядко се разглежда.
* **функционални изисквания** – Тези изискване дефинират функциите на софтуерната система или нейните компоненти. Функционалните изисквания могат да са някакви изчисления, технически детайли, различни видове обработки и работа с данни или други специфични функционалности, които определят както точно трябва да прави дадената система.

# UML стереотипи

## Structure diagrams (Структурни диаграми)

Структурните диаграми подчертават нещата, които трябва да са налице в системата която се моделира. Тъй като структурните диаграми представят структурата на системата, те се използват широко при документирането на софтуерната архитектура на софтуерните системи.

* **Class diagram (клас диаграма)** - описва структурата на системата, като представя нейните класове, техните атрибути и връзките между класовете.
* **Object diagram (обектна диаграма)** – представя пълен или частичен изглед на структурата на моделираната система в определен момент от време.

## Behavior diagrams (Диаграми на поведение на системата)

Тези диаграми подчертават какво трябва да се случва в моделираната система. Тъй като тези диаграми илюстрират поведението на системата, те се използват широко за описание на функционалността на софтуерната система.

* **Activity diagram (диаграма на действията)** – описва бизнес и оперативните потоци на работа на компонентите в системата. Диаграмата на действията представя цялостния поток на работа на системата.
* **State diagram (диаграма на състоянията)** – описва състояната на системата и прехода между тях.
* **Use Case diagram (диаграма на потребителски случаи)** – описва функционалността предоставена от системата по отношение на участниците, техните цели , представени чрез потребителски случаи и всякакви зависимости между тези случаи.