# ПОТРЕБИТЕЛСКА ЕФЕКТИВНОСТ НА ИНФОРМАЦИОННИТЕ СИСТЕМИ В ИНТЕРНЕТ: МЕТОДОЛОГИЧНИ ПРОБЛЕМИ ПРИ ПОТРЕБИТЕЛСКИ ОРИЕНТИРАНОТО ПРОЕКТИРАНЕ

Ваня Лазарова*

## 1. ЕРГОНОМИЧНОСТ, ПОЛЗВАЕМОСТ И ПОТРЕБИТЕЛСКА ЕФЕКТИВНОСТ НА ИНФОРМАЦИОННИТЕ СИСТЕМИ

### 1.1. Проблемът за потребителската ефективност на информационните системи

През последните десетина години се изграждат многобройни информационни системи в Интернет, предназначени за използване от крайни потребители – неспециалисти по информационни технологии. Такива информационни системи се създават в много предметни области и чрез тях се извършват различни видове услуги. Такива системи са справочно-информационните сайтове, сайтовете за онлайн търговия, сайтовете на държавните институции за административно обслужване на граждани, онлайн библиотеките и многобройните сайтове за онлайн обучение.

Спрямо крайните потребители на тези системи разработчиците не могат да поставят изисквания за обучение, каквато е масовата практиката при информационните системи, използвани от служители на дадена фирма. Дори и сайтът да съдържа подробни указания, никой не може да задължи потребителя да ги прочете. Много от потребителите не четат указанията, дадени в помощните раздели на сайтовете, а разчитат, че сами могат да се ориентират какво и как да направят. Когато потребителят е затруднен при извършването на необходимите действия, в голям процент от случаите се отказва да ползва даден сайт. Ако сайтът е комерсиален, един потребител на даден онлайн магазин, например, ще закупи стока от този магазин само ако (а) успее достатъчно бързо и лесно да открие, дали се предлага стоката, която търси, и (б) успее също така бързо и лесно да поръча и плати стоката. В противен случай потребителят най-вероятно ще потърси конкурентен сайт, предлагащ същите стоки и услуги, но по разбираем начин.

Проблемът за потребителската ефективност добива особена актуалност у нас във връзка с въвеждането на административни услуги за гражданите в рамките на приетата от Министерски съвет (2002) "Стратегия за електронно правителство".

Сайтовете за административни услуги са предназначени за ползване от десетки или стотици хиляди потребители. В тези случаи пряк потърпевш е потребителят, защото той е принуден да търси посредник, за да получи желаната услуга. Потърпевша е и самата институция, чието задължение е да извършва услугата. Ако примерно въведеният в действие от началото на тази година сайт за подаване на данъчните декларации в електронен вид се окаже неудобен и трудно разбираем за голяма

* Ваня Лазарова е доктор по икономика, главен асистент в катедра „Информационни и комуникационни технологии" на УНСС; тел.: 81-95-593, e-mail: lazarova@gbg.bg

част от потребителите, то губещи ще са първо данъкоплатците, които ще трябва отново да ползват посредничеството на счетоводител за попълване на декларациите. Но губеща ще е и данъчната администрация, защото ако процентът на декларациите, подавани по електронен път, е нисък, тогава ще е нужно да се запази и целият досега действащ персонал, обработващ декларациите, подавани на хартиен носител.

И в двата случая – и при комерсиалните и при публичните сайтове – обвинения в "компютърна неграмотност" на потребителите са безполезни. Отговорността за това в каква степента крайните потребителите успяват да ползват дадена информационна система е изцяло на разработчиците.

Оценката на информационна система от гледна точка на крайните потребители се означава на английски с термина *usability* (*използваемост)*, но в специализираната литература на български език, за да се разграничи по-ясно новото, специфично за информационните технологии понятие, като термин се използва неологизма *"ползваемост"* (Мария Николова, 2001).

Разграничават се три основни аспекта на ползваемостта (ISO 9241-11, 1998):

- потребителска ефективност (effectiveness),
- потребителска производителност (efficiency) и
- потребителска удовлетвореност (satisfaction).

Потребителската ефективност има централно място при определяне на ползваемостта. Високата продуктивност и естетическата стойност на една система безспорно допринасят за повишаване на общата оценка на ползваемостта й. Но при съществено разминаване на това, което системата може да прави, и това, което потребителят очаква от системата, т.е. при ниска потребителска ефективност, и най-перфектният от естетическа гледна точка дизайн не може да направи тази система ползваема.

За постигане на висока потребителска ефективност на информационните системи (и висока ползваемост като цяло) се изисква нов подход при проектирането на потребителския интерфейс. Този подход, при който в центъра на анализа се поставят целите, желанията, познанията и ограниченията на крайните потребители, в англоезичната литература се означава с термина *user-centered design* (потребителски центрирано проектиране); на български се използва терминът "*потребителски-ориентирано проектиране"*, или еквивалентният термин "*проектиране, ориентирано към потребителя"*.

### 1.2. Предмет и цели на изследването

Предмет на изследването е *потребителската ефективност* при информационните системи в Интернет, предназначени за работа с крайни потребители.

Значимостта на проблема за потребителската ефективност нараства изключително много в резултат на бурното развитие на онлайн информационните системи, насочени към стотици хиляди крайни потребители (неспециалисти по информационни технологии), в областите на системите за електронната търговия (e-commerce), образователните системи (e-learning) и системите за административно обслужване на гражданите (e-government).

Анализът е насочен към решаването на няколко проблема:

- Да се изясни понятието за *потребителската ефективност* в контекста на по-общото понятие за *ползваемост* на информационните системи (раздел 1).
- Да се анализират и обобщят теоретичните принципи на *потребителски ориентираното проектиране* като специфичен подход за постигане висока потребителска ефективност (раздел 2).
- Да се изследват факторите, свойства и характеристиките на потребителския интерфейс, които имат съществено влияние за повишаване на потребителската ефективност; да се формулират определени препоръчителни свойства на графичния интерфейс и полезни практики при потребителски ориентираното проектиране (раздел 3).
- Да се разкрият възможностите за използване на унифицирания графичен език за моделиране UML при потребителски ориентираното проектиране на информационни системи (раздел 4).

*За терминологията:* Една допълнителна задача, за чието решаване се опитваме да спомогнем с тази студия, е изграждането на адекватна терминология на български език в областта на проектирането, ориентирано към потребителя. Две причини пораждат терминологичния проблем. Първата причина е, че това е относително нова област и няма единна терминология, възприета от специалистите по проектиране на информационни системи. Втората причина за терминологичния проблем е начинът на превеждане на английските термини. Английският език е основният език на научните документи в тази област (както и в информатиката като цяло). Думите, използвани в английската терминология, могат да бъдат превеждани по различен начин на български. Но ние имаме нужда от определена конвенция за *еднозначно* съпоставяне на използваните български и английски термини.

По отношение на първия аспект на терминологичния проблем ние даваме предимство на терминологията, използвана в стандартите на Международната стандартизационна организация ISO (International Standard Organization). Терминологията на ISO има две предимства – тя е добре позната на специалистите в областта и дава едно базово непротиворечиво (макар и неизчерпателно) множество от термини. Разбира се, терминологията предлагана от ISO не бива да се разглежда като някакво външно наложено ограничение и не е пречка за развитието на терминологията в областта на изследването.

По отношение на превода на английската терминология ние сме се съобразявали с наличните преводи в българската литература. Както отбелязахме по-горе, за двата основни английски термина “usability” и “*user-centered design”* тук няма да използваме техния буквален превод, а двата приети в нашата литература съответстващи термина: *ползваемост* и *потребителски ориентирано проектиране*. По отношение на другите термини, при първа употреба на даден български термин, в скоби посочваме съответстващия английския термин. По този начин целим да се избегне възможно недоразумение. Надяваме се, че в близко бъдеще терминологията на български език в областта на потребителски ориентираното проектиране ще бъде достатъчно утвърдена, за да не се налагат чести препратки към английските термини.

### 1.3. Софтуерна ергономия и потребителска ефективност

Ергономията на информационните системи, наричана също ергономия на взаимодействието човек-компютър, включва два големи раздела: хардуерна ергономия и софтуерна ергономия.

*Хардуерната ергономия* обхваща влиянието на физическите свойства на компютърните устройства, използвани при работа с информационната система. Много аспекти на хардуерната ергономия на информационните системи се регулират от международната организация по стандартите ISO. Например стандартът ISO 9355-2 (1999) формулира ергономични принципи за проектирането и използването на дисплеи, стандартът ISO 13406-2 (2001) формулира изисквания специално към плоски екрани (LCD); а ISO 9241-400 (2007) формулира ергономични принципи за проектирането и използването на входни устройства – клавиатури, мишки, устройства за управление на курсора при лаптопи (трак-бол и трак-пад), джой-стикове, чувствителни към допир екрани.

*Софтуерната ергономия* изследва ергономичните аспекти от програмната реализация на информационната система. Софтуерната ергономия обхваща всички ергономични аспекти на разработването и използването на софтуера на информационните системи, включително ергономията на програмисткия труд.

*Ползваемостта (usability)* е дял от софтуерната ергономия, в който се изследва взаимодействието между информационните системи и *крайните потребители*. Ползваемостта на дадена информационна система е показател за ефективността на системата от гледна точка на целите, знанията и уменията на потребителите (Лазарова, Ваня, Д. Петров, 2007). Ползваемостта обхваща три класа характеристики на информационните системи, фокусирани върху три различни типа отношения към крайните потребители на информационната система:

- *Потребителска ефективност (effectiveness).* При този аспект се акцентира на това, доколко функционалният обхват и начинът на работа със системата съответстват на нуждите, целите, уменията на потребителите.
- *Потребителска производителност (efficiency).* При този аспект се акцентира на количеството полезни действия, които потребителят може да извърши за определено време.
- *Потребителска удовлетвореност (satisfaction*). Това е субективната емоционална и естетическа оценка на потребителя за работата на системата и неговата лична удовлетвореност/неудовлетвореност от начина на работа и получените резултати.

*Потребителската ефективност*, която е предмет на настоящото изследване, също се разглежда като формираща се от логически обособени, но практически тясно свързани характеристики. Няма общоприета "номенклатура" на характеристиките, формиращи потребителската ефективност. В това изследване използваме десет характеристики, аспекти на потребителската ефективност:

- Навигация
- Архитектурна и визуална яснота
- Полезност

– Контрол на потребителя
– Език и съдържание
– Онлайн помощ и ръководство за потребителя
– Системна и потребителска обратна връзка
– Достъп до мрежата
– Съгласуваност
– Съобщения за грешки и корекции

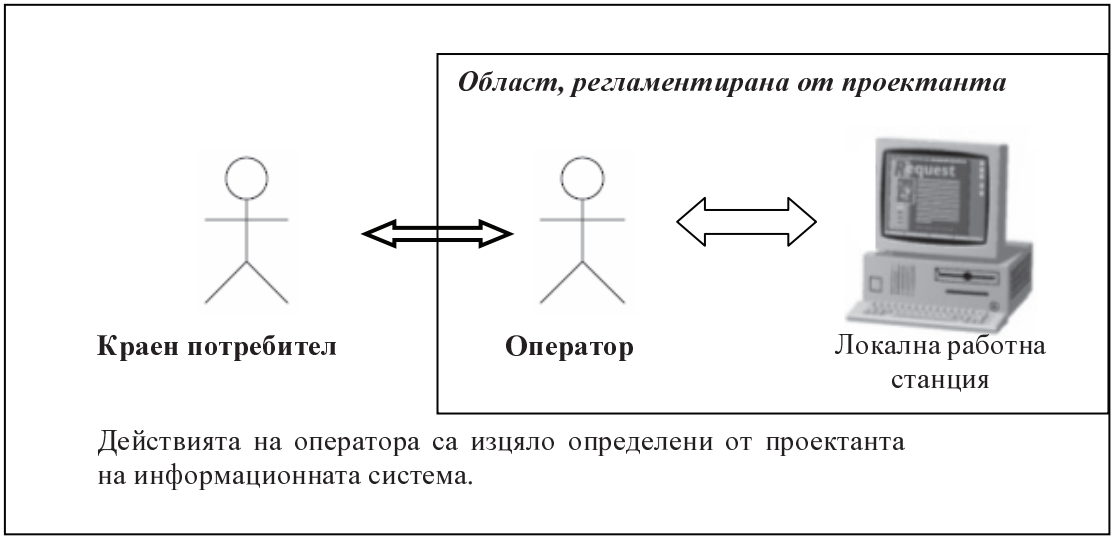
Смисълът на всяка от тези десет характеристики е разяснен в раздел 4. Сега ще се спрем само на една от тези характеристики – полезността.

Оценката за *полезност* (usefulness) на една информационна система е оценка за това, какво потребителят може да направи чрез системата, в съответствие със своите познания, умения и желания. Полезността е ядрото на потребителската ефективност. Проблемът за потребителската ефективност най-често произлиза от несъответствието между:

– *полезността* на една система (от гледна точка на потребителя), и
– *функционалния обхват* на системата – пълната съвкупност от услуги, които системата може да извършва, и/или задачи, които може да решава.

Несъответствието се проявява в това, че потребителите не използват част от функциите на системата или защото не са им нужни, или защото не умеят да ги ползват. Нека разгледаме въпроса как се стига до тази нежелана ситуация.

При информационните системи, обслужвани от оператори, по правило няма разминаване между функционален обхват и полезност. При проектирането на информационна система, предназначена да бъде обслужвана от оператор (фиг. 1), проектантът решава как работата, която трябва да се свърши, да бъде разпределена между компютъра и оператора. Операторът е лице, чието служебно задължение е да работи с дадена система. Операторът е специално обучен – чрез курсове и чрез изучаване на документация – да обработва всички задачи от функционалния обхват на системата. Примерно човек, който работи като оператор-фактурист в една търговска фирма, по правило знае всички операции, които могат да се извършат с модула за фактуриране на своята работна станция. Операторът може да изготви всеки предвиден вид фактури; той познава и умее да използва клавиши за директен избор на номенклатури от стоки, позволяващи по-бързото изготвяне на една фактура. При системите, обслужвани от оператор, резултатът се предоставя обикновено на друго лице, наричано *краен потребител*. В разглеждания пример крайни потребители са купувачите, които получават фактурите, изготвяни от оператора.

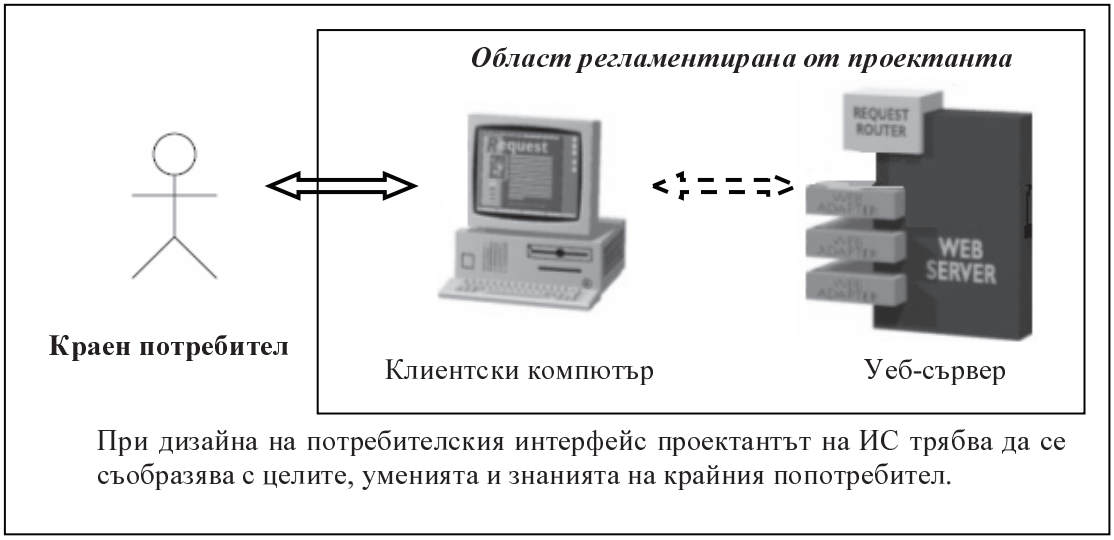



**Фигура 1. Информационна система, обслужвана от оператор.**

Несъответствия между *функционален обхват* и *потребителска полезност* възникват, когато информационната системата се ползва *пряко* от крайните потребители. Кои са причините, водещи до тези несъответствия? Защо, от една страна, в информационните системи се включват функции, от които крайните потребители не се интересуват? И защо, от друга страна, много от крайните потребителите не желаят или не могат да изпълнят някои от задачите, заложени във функционалния обхват на дадена информационна система?

Нека продължим посочения по-горе пример и предположим, че търговската фирма създава собствен сайт за електронна търговия и в този сайт е включен модул за попълване на данни за фактурата. Модулът за фактуриране лесно може да се препрограмира в уеб-приложение и фактурирането да стане достъпно за потребителите в Интернет.

На пръв поглед изглежда разумно,модулът за фактуриране в сайта да има същия дизайн и същата функционалност, както локалната версия на програмата, използвана от оператор-фактурист. Разликата е в това, че сега данните за фактурата трябва да се попълват пряко от крайния потребител – купувача. Предполага се, че купувачът отлично знае съдържанието на документа. От него се изисква *само* да въведе данните в една екранна бланка. Това се счита за *елементарна работа*, която досега е вършена от оператор със среден или нисък образователен ценз. Но разработчиците на уеб-приложението ще бъдат силно изненадани, когато узнаят че мнозинството от техните клиенти считат предложения модул за фактуриране за “прекалено сложен” и “неразбираем”, че според клиентите сайтът “не следва нормалната логика за избор на стоки и фактуриране” и “изисква излишна информация”. В такава ситуация в миналото разработчиците често обясняваха отрицателните оценки за системата с *ниската компютърна грамотност* на крайните потребители.




**Фигура 2. Информационна система, използвана пряко от крайните потребители.**

През последните години, в условията на все по-широко използване на Интернет и при силната конкуренция в електронната търговия, настъпиха съществени промени в теорията и практиката на проектирането на информационни системи за крайните потребители. Мнозинството специалисти по информационни технологии днес приемат, че разработването на информационни системи, предназначени за пряко използване от крайните потребители, изисква специален подход, различен от подхода, прилаган при системите, обслужвани от оператори. Наложи се по безспорен начин възгледът, че при разминаване на функционалния обхват на системата, от една страна, и възможностите и потребностите на крайните потребители, от друга страна, грешката не трябва да се търси в крайните потребители, а в разработчиците. Разработчиците са длъжни да изучават и да се съобразяват с целите, уменията и знанията на крайните потребители.

## 2. ПОТРЕБИТЕЛСКИ ОРИЕНТИРАНИЯТ ПОДХОД ПРИ ПРОЕКТИРАНЕ НА ИНФОРМАЦИОННИ СИСТЕМИ

### 2.1. Основни идеи на потребителски ориентирания подход

Идеята за потребителски ориентираното проектиране добива популярност с книгата на Norman и Draper (1986) "*Потребителски центрираният системен дизайн: Нови перспективи на взаимодействието човек-компютър*"

През периода 1986-1993 г. развитието на потребителски ориентираното проектиране остава ограничено в няколко изследователски центъра и няма сериозно влияние върху софтуерната индустрия. След 1993 година започна изключително бързо

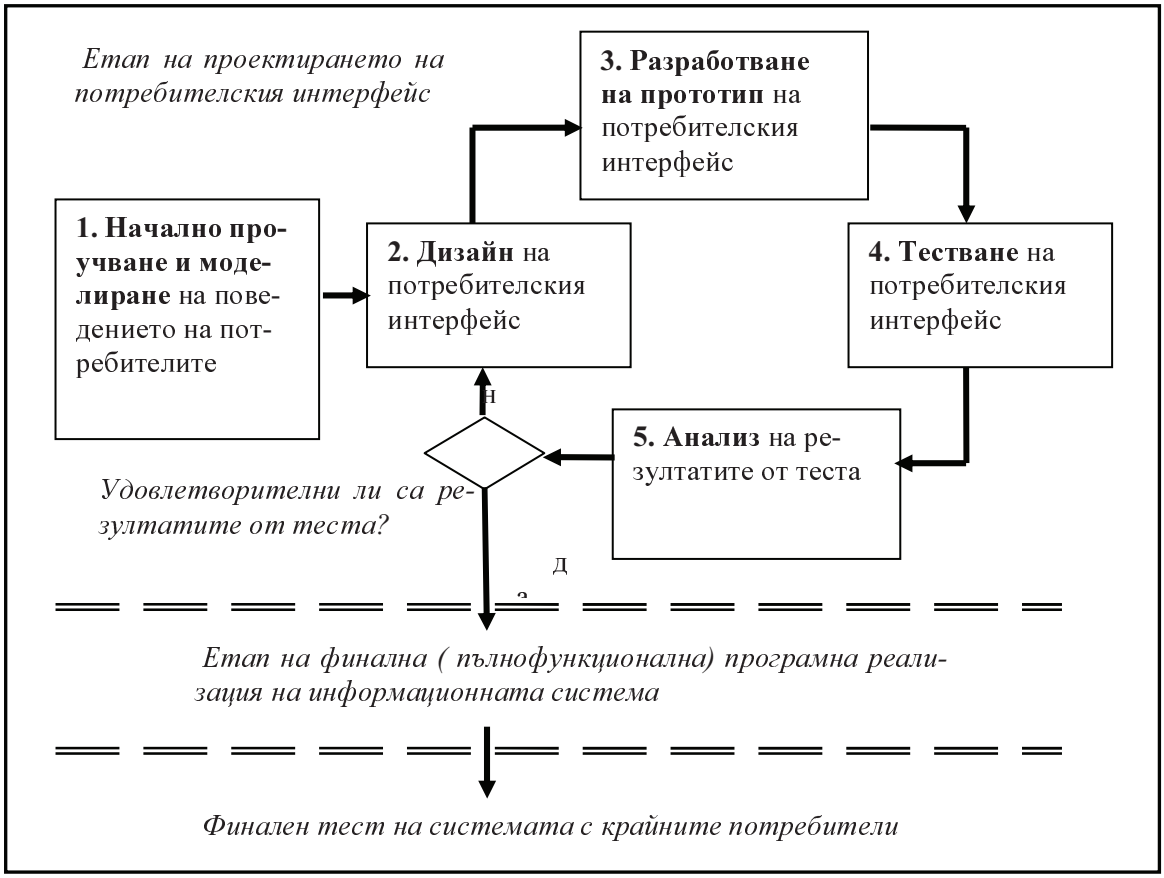
развитие на Интернет и по-специално на уеб-технологиите – използването на езика HTML, протокола HTTP. През 1998 година е приет стандарт за ползваемостта ISO 9241-11 (1998). През 1999 е приет стандарта ISO 13407 (1999) за потребителски ориентираното проектиране. (В този документ се ползва синонимният термин *"human*-centred design" - *човеко*центрирано проектиране. Този термин получи разпространение само в Русия, където методът се нарича "человекоориентированное проектирование").

Фокусирането върху потребителя е голяма стъпка напред в сравнение с технологично ориентираното проектиране, при което потребителят се третира в голяма степен като оператор. При потребителски ориентираното проектиране, дизайнерът на системата се стреми да подчини функционалността на системата на целите на потребителя.

### 2.2. Фази на проектирането на потребителския интерфейс

Процесът на разработване на потребителския интерфейс включва две етапа:
- етап на проектиране, който включва и частична програмна реализация, доколкото е необходима за създаването на прототип на интерфейса;
- етап на пълнофункционална програмна реализация на потребителския интерфейс.

Проектирането се разглежда като *итеративен* процес, който започва с предварително изследване на потребителите, последвано от няколко цикъла на изграждане на прототип на потребителския интерфейс, тестване и съответни промени в дизайна. Разграничават се следните фази:
1. Начално проучване и моделиране на поведението на потребителите.
2. Дизайн на потребителския интерфейс.
3. Разработване на прототип на потребителския интерфейс.
4. Тестване на потребителския интерфейс.
5. Анализ на резултатите от теста.

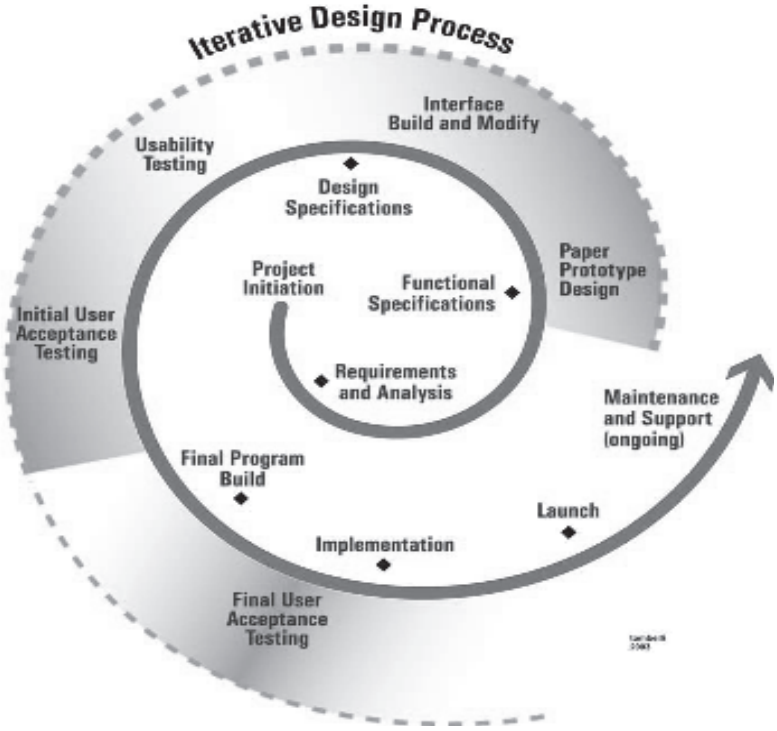



**Фигура 3 (а). Фази на потребителски ориентираното проектиране**

Когато оценката на резултата от потребителския тест е удовлетворителна, се преминава към следващите етапи:

- финална (пълнофункционална) програмна реализация
- финален тест на системата с крайните потребители.

На фиг. 3(а) е показана логическата последователност на фазите. На фиг. 3(б) са показани същите фази, но се илюстрира *спираловидният* характер на процеса на разработване на потребителския интерфейс; всяка следваща итерация води до развитие на системата.




**Фигура 3 (б). Потребителски ориентираното проектиране е *спираловиден процес.* (michigan.gov, 2003)**

**Проучване и моделиране на поведението на потребителите**

Преди да се изгради система, трябва подробно да се проучат потребителите й. Използват се както традиционните методи за допитвания до потребителите чрез интервюта и анкети, така и методи, основани на активност от страна на потребителите. От втората група методи все по-голямо значение придобиват тематичните форуми в Интернет.

При експерименталните изследвания се наблюдава протичането на диалога „потребител – система" (т.е. последователността от действия на потребителя). Регистрират се както последователността на действията, така и времето, отделяно за всяко действие. Експерименталните изследвания чрез пряко физическо наблюдение на поведението на потребителите (заснемане с камера) са трудоемки и скъпи. Почесто се прилагат методи за софтуерна регистрация на диалога на потребителите със системата.

Тези изследвания могат да дадат многостранна информация, която обаче трябва да бъде обобщена, осмислена и разбрана. Понякога е възможно сред потока от всякакви данни, с които изобилства Интернет, да се пропусне ключова и ценна информация. Като краен резултат само някои от изводите и заключенията, базирани на част от събраната информация, са съотносими към дизайна и ефективния потреби-

телски интерфейс. Днес потребителите в Интернет свободно съобщават мнението си по всеки въпрос, а разработчиците на сайтове, информационни системи, отделни страници непрекъснато ги подканят да дават мнения и оценки. И потребителите правят това много охотно в повечето случаи.

Главният въпрос при изследването на потребителите е да се посочи върху какво трябва да се фокусира вниманието на разработчиците и дизайнерите и какво трябва да се игнорира.

Потребителски ориентираният подход представя потребителите в набори от потребителски профили с разпознаваеми стереотипи на поведение. Потребителските профили са популярна форма на представяне на резултатите от изследването на потребителите. Профилът включа реалистично и правдоподобно конструирано описание на единичния потребител. Обикновено профилът обединява в едно цяло много детайли като среда, характер, история и опит, манталитет и навици. Това дава възможност потребителят да бъде "разбран" и "почувстван".

**Дизайн на потребителския интерфейс**

Дизайнът е основната дейност на проектанта. На тази фаза се разработват (модифицират при следващите итерации) трите вида модели на потребителския интерфейс, които ще разгледаме по-нататък.

**Разработване на прототип на потребителския интерфейс**

Разработването на прототип на потребителския интерфейс е задължително условие за прилагане на потребителски ориентираното проектиране. Единственият ефективен начин да получим оценка от потребителя *преди* окончателното завършване на системата, е да му предоставим действаш прототип.

Технологиите за разработване на прототип излизат извън обхвата на настоящата студия.

**Тестване на потребителския интерфейс**

Тестването на ползваемостта дава в повечето случаи неочаквани резултати за дизайнерите. Причината е в разминаването на представата на дизайнерите и разработчиците за сайта с това, което потребителите действително правят. Потребителите пренебрегват инструкциите; ако нещо ги затруднява, най-често продължават нататък, опитвайки се чрез налучкване или на база на предишни знания да свършат работата си. Докато обратно – разработчиците и дизайнерите си представят как потребителят внимателно изчита всичко на сайта преди да пристъпи към каквато и да било работа.

Тестването се извършва обикновено върху неголяма група от потребители – 3 до 8 души, тъй като в случая не е нужно да се търси масовост. Тестването за ползваемост не е социологическо проучване за посещаемостта или известността на сайта.

На потребителите се поставят за изпълнение определени задачи за работа със системата и се наблюдава тяхното поведение, измерва се времето, за което успяват да се справят. В други изследвания потребителите се запознават със системата и след това попълват тест, в който отразяват своето мнение.

**Анализ на резултатите от теста**

Анализът на резултатите показва доколко очакванията на разработчиците на системата съвпадат с очакванията на потребителите. При сериозно разминаване трябва да се преосмисли цялостната стратегия и целите на системата, тъй като в случая е ясно, че не потребителите трябва да се адаптират към информационната система. Ако те не успяват да се справят с нея, трябва внимателно да се анализира процесът на тестване, да се намерят слабите места и процесът да се върне итеративно към дизайна на системата.

## 2.3. Три типа модели на потребителския интерфейс

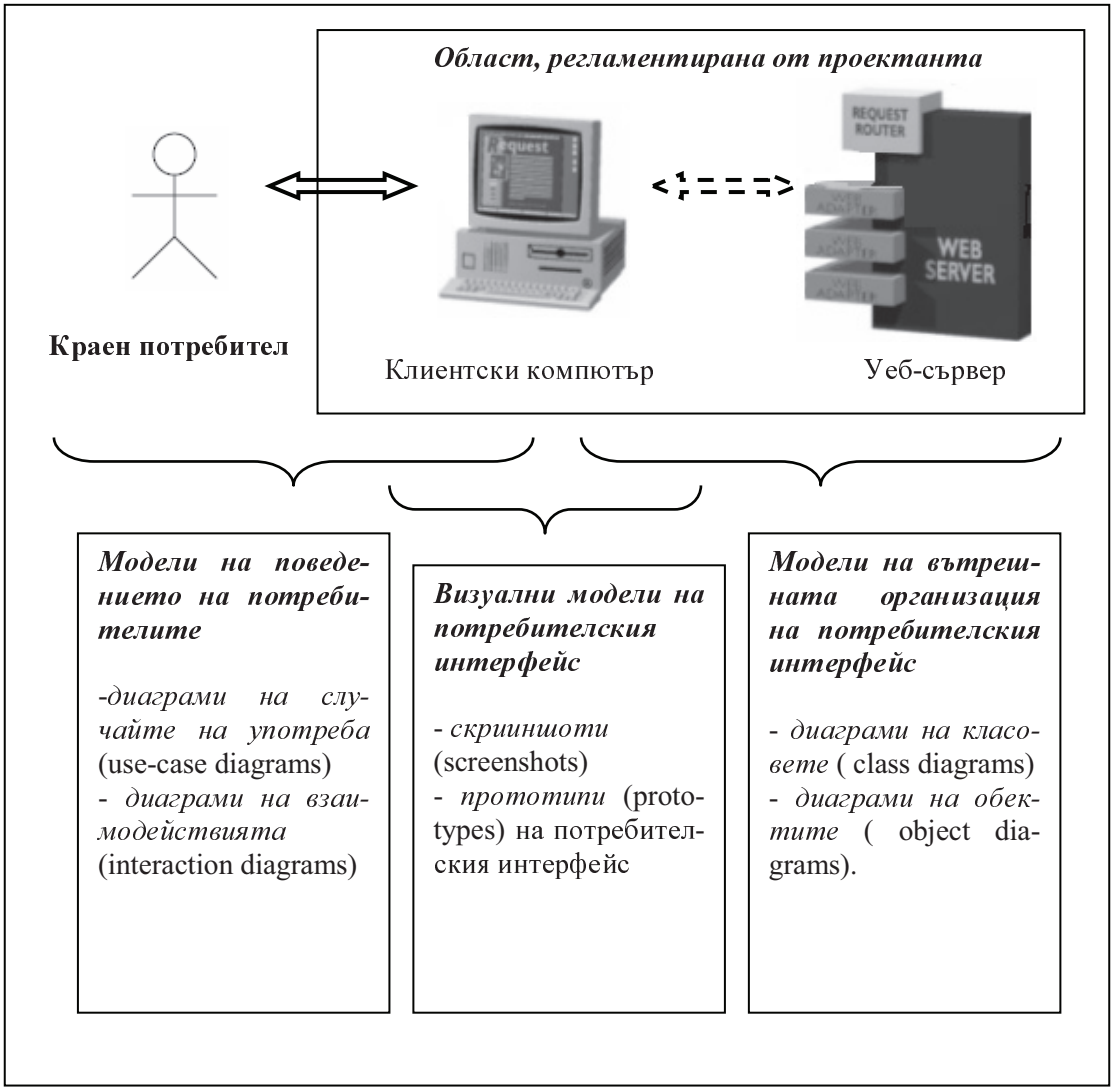
В процеса на проектирането се използват различни типове модели на потребителския интерфейс. Моделите включват както условни графични представяния (диаграми), така и текстови описания.

Можем да разграничим следните типове модели на потребителския интерфейс:

- модели на поведението на потребителите,
- визуални модели на потребителския интерфейс,
- модели на вътрешната структура на потребителския интерфейс.

*Моделите на поведението на потребителите* представят чрез диаграми и текстови описания действията, които потребителят извършва при една или друга ситуация при взаимодействието с информационната система. В тези модели се включват и действия, които не са пряко насочени към системата, но са логически или технологично свързани с работата на потребителя със системата. Тези модели са включени в рамките на общия модел на системата, а често и в по-обхватни информационни инфраструктури (Кисимов, Валентин, Р. Николов. 2007).

Можем да съпоставим трите типа модели с три гледни точки върху процеса на взаимодействие на крайния потребител със системата (фиг. 4)




**Фигура 4. Съпоставка на трите модела**

*Прототипите* са "действащи" програмни модели, които симулират достатъчно точно потребителски интерфейс на бъдещата система. В раздел 4 ще разгледаме определени характеристики на потребителския интерфейс, които в хода на разработката се оценяват от крайните потребители въз основа на прототипи. Само в редки случаи, на начални фази на проектирането, на потребителите се представят за оценка с*крийншоти* (screenshots) на потребителския интерфейс, т.е "снимки" на екранно изображение, представящи определени моментни състояния от потребителския интерфейс.

Другите два типа модели – моделите на поведението на потребителите и моделите на вътрешната организация на потребителския интерфейс – ще разгледаме в раздел 5. На фигура 4 са посочени някои видове UML диаграми, използвани при тях.

При *моделите на поведението на потребителите* се използват предимно:
- *диаграмите на случаите на употреба* (use-case diagrams), и
- четири типа *диаграми на взаимодействията* (interaction diagrams):
  - *Диаграма на комуникации*
  - *Диаграма на последователността*
  - *Времева диаграма*
  - *Диаграма за преглед на взаимодействията*

При *моделите на вътрешната организация на потребителския интерфейс* често използвани UML диаграми са:
- *диаграми на класовете* (class diagrams), и
- *диаграми на обектите* (object diagrams).

## 3. ПОКАЗАТЕЛИ ЗА ПОТРЕБИТЕЛСКАТА ЕФЕКТИВНОСТ

Потребителите могат да оценят – при взаимодействие с прототип или при реална работа – дали интерфейсът на системата е добър, или не. Но те не могат да кажат *как* трябва да работи системата, какви свойства трябва да има интерфейсът. Дизайнерските решения са изцяло задача на проектантите (Мурджева, Александрина и др., 2007). Един от основните методологични въпроси в потребителски ориентирания подход е да се определят онези характеристики на потребителския интерфейс, които допринасят за повишаване на качеството му.

В настоящия анализ ще ползваме десет *показатели за ползваемостта* (Usability Guidelines), формулирани от Information Services and Technology (IS&T) към Масачузетският технологичен институт.
- **Навигация** – местоположение в сайта, достъпност на основните части от главната страница, търсене в сайта
- **Архитектурна и визуална яснота –** структуриране на секции и страници, карта на сайта
- **Полезност** – разнообразни и достъпни функции, отговарящи на предметната област
- **Контрол на потребителя** – точки за вход и изход, прекъсване на работа, връщане
- **Език и съдържание** – термини, отговарящи на предметната област, ясни хипервръзки, кратко съдържание
- **Онлайн помощ и ръководство за потребителя** – достъпна и навременна помощ
- **Системна и потребителска обратна връзка** – адреси и телефони за връзка с администратора или собственика на сайта
- **Достъп до мрежата** – актуални web стандарти и конфигурации

– **Съгласуваност** – логическа връзка между елементите на сайта
– **Съобщения за грешки и корекции** – разпознаване на потребителските реакции, ясен изход от всяка ситуация

### 3.1. Навигация

Когато потребителят активира дадена информационна система в Интернет – електронен магазин, борса, аукцион или какъвто и да било друг сайт, целта му е да открие нещо, да намери информация, да поръча стока, да предложи нещо за продажба или просто да разгледа какво има там. Във всички случаи обаче той трябва да може лесно да се ориентира. Навигацията трябва точно да показва на потребителя къде се намира. Йерархичната структура на сайта трябва да е ясна и видима, защото по нея потребителят се ориентира какво съдържа сайтът. Навигацията също така трябва да посочва къде е началото и откъде трябва да се започне търсенето. Потребителят трябва да е сигурен в кой сайт е – т.е. логото на фирмата трябва да присъства като задължителен елемент от навигацията и да бъде ясно разпознаваемо.

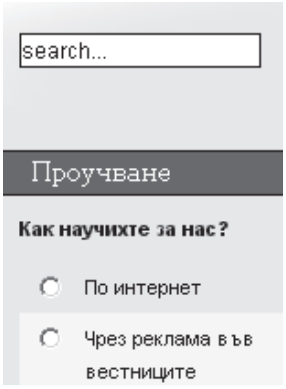
Стратегиите при изграждане на навигацията са две:
– да се използва Search;
– да се браузва, като се преминава през йерархия от страници, докато се открие необходимото.

Потребителите ползват и двете стратегии според особеностите на характера и навиците си. Някои не искат да разглеждат и се насочват директно към търсене на това, което ги интересува. Други обратно, първо разглеждат "наоколо", за да видят какво се предлага, и после търсят това, за което са отворили сайта.

Използването на Search има няколко тесни места от гледна точка на потребителски ориентираното проектиране. Търсенето в никакъв случай не трябва да затруднява потребителя. Той не трябва да обмисля по какъв начин е конструирана машината за търсене в сайта и как да формулира запитването си. Голяма част от проектантите и програмистите на сайтовете са разбрали това и вече почти не се срещат сайтове, при които да е нужно да специфицираш въпроса си и да използваш само тези ключови думи, които машината за търсене очаква (фиг.5)


**Фигура 5. Ясен и недвусмислен бутон за търсене и поле за въвеждане на въпроса**

Ако обаче структурата е неясна, например има поле за въвеждане на въпрос, но няма бутон за търсене, потребителят не може веднага да се ориентира ясно и еднозначно, започва да чете ненужната за него информация или попада в някакво проучване, което (ако не е желано) може да го накара за излезе от сайта (фиг. 6)




**Фигура 6. Поле за въвеждане на данни, но липса на бутон за търсене**

Търсенето може да се ограничи чрез добавяне на допълнителни опции, но само в случаите, когато трябва да се помогне на потребителя да получи от търсенето това, което му е нужно. В една онлайн борса за употребявани автомобили, например, много полезно е потребителят да може да въведе точната марка автомобил, който търси, кубатурата му, цвета или годината на производство.

Навигацията през страниците се подчинява на някои общоприети правила:

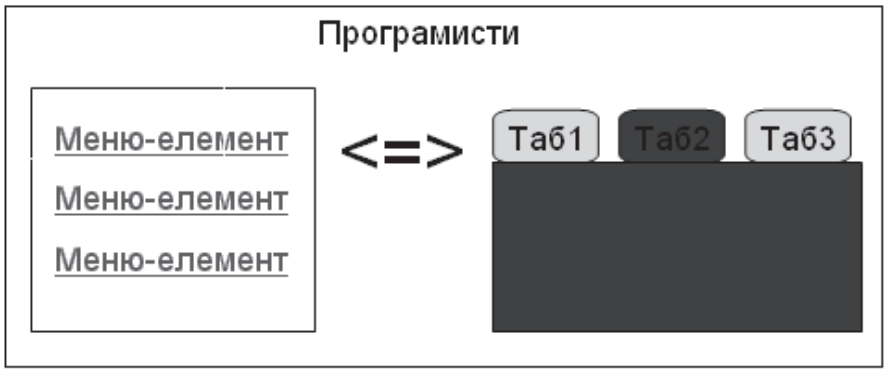
– На всяка страница от сайта навигацията трябва да е на едно и също място и да изглежда по един и същи начин, така че потребителят да е сигурен, че винаги може да намери началото или да смени страницата с друга.
– Всички елементи на навигацията – лого, хипервръзка към началната страница, секции на сайта, подсекции, страници, области на страницата, елементи на страницата, инструменти, трябва да са ясно разграничими.

### *Нови инструменти на потребителския интерфейс за изграждане на навигация*

Табовете (tabs, разделители) са сравнително нов инструмент на интерфейса, чрез който се разделя съдържанието на сайта на секции. Много големи сайтове (и не толкова големи) започнаха да използват табовете като навигационно решение. Причините за тази бърза популярност на табовете са, че те са очевидни, създават усещане за тримерност на сайта и изглеждат много добре (фиг. 7).

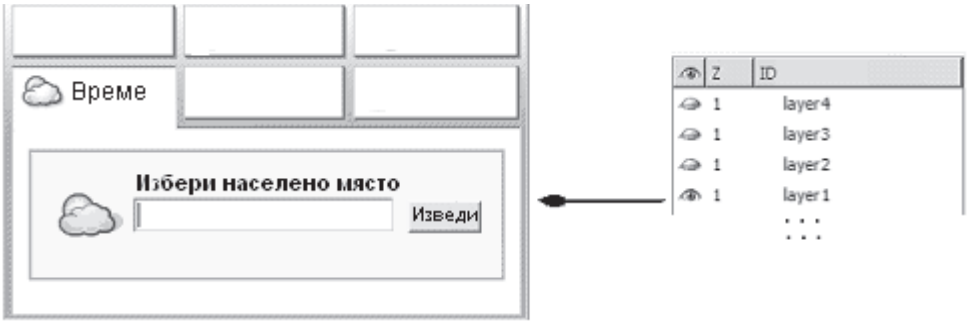
**Фиг. 7. Табовете като инструмент на навигацията**

От гледна точка на програмната реализация на сайта табовете са аналогични на менютата. За програмиста е все едно дали ще програмира меню или таб – тези елементи се различават само графично (фиг. 8).




**Фигура 8. Еквивалентност между менюта и табове за програмистите**

За потребителя обаче между тези две неща няма нищо общо... И не само това. Напоследък, поради голямата борба за все повече пространство, се появиха табове, съчетани с пластове (layers), Големите сайтове (като yahoo.com) първи ги използваха. Придвижвайки се с курсора по табовете съдържанието им се отваря без потребителят да е щракнал (кликнал) върху него. Това се осъществява чрез използване на пластове, които физически се намират на едно и също място върху страницата, но стават видими едва когато се посочи хипервръзката, текста или елементът, който ги активира (фиг. 9).




**Фигура 9. Видим пласт ( layer) 1, останалите заемат същото пространство, но са невидими**

Потребителите ги възприеха много бързо, без да се колебаят за предназначението им, защото вече бяха свикнали с употребата на табовете. За програмистите на сайта обаче те са нещо съвсем различно, тъй като се използва изцяло нова технология – наслагващи се едни върху друг невидими пластове, а не старата технология на менютата.

Всички тези разсъждения сочат, че понякога при потребителски ориентираното проектиране новаторските елементи могат изобщо да не се забележат и възприемат като нещо ново от потребителите и обратно, добре познатата стара технология, ако бъде представена с нов графичен дизайн, може да доведе до революция във възприятията на потребителите.

### 3.2. Архитектурна и визуална яснота

Навигацията на сайта е свързана с неговата архитектура и визуална яснота.

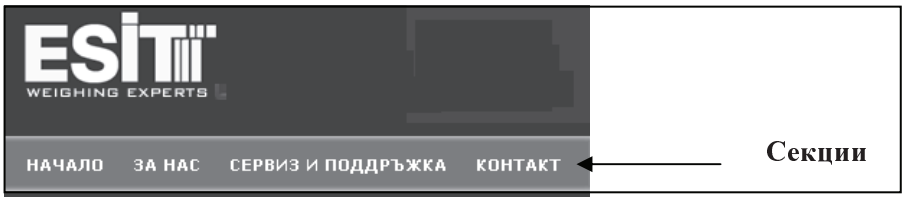
*Знакът (логото)* на фирмата, чийто сайт разглежда потребителят, е много важен елемент, тъй като той еднозначно идентифицира сайта. Докато потребителят го вижда, той е сигурен къде се намира. Логото трябва да е най-високо разположеният елемент в структурата. То може да бъде изнесено отгоре, да прекъсва ограждането на страницата с рамка или да е вътре в страницата (фиг. 10) – но във всички случаи да е ясно забележимо и на място, където потребителят очаква да го види. Напоследък много добра практика, с която потребителите вече са свикнали, е самото лого да бъде и хипервръзка към началната страница. Това, разбира се, не премахва нуждата от бутон Home, просто е още едно застраховане, че потребителят няма да се изгуби из структурата на сайта при навигацията и винаги може да се върне в началото.


**Фигура 10. Място на логото в сайта**

Много интересна и ориентирана към потребителя е тенденцията напоследък някои браузъри да изобразяват логото на фирмата пред URL адреса: 

. Докато потребителят е в този сайт, вижда логото му, поставено в началото на адреса. Ако го напусне, изчезва и логото.

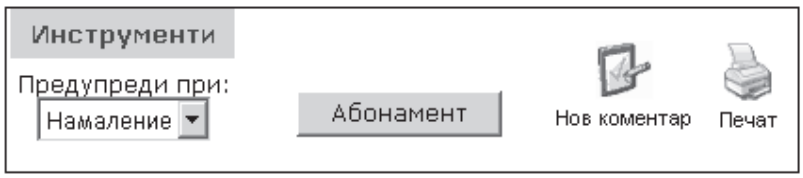
*Секциите на сайта* са хипервръзките към главните му подразделения (фиг. 11). Те представляват горното ниво на йерархичната структура на сайта.




**Фигура 11. Секции на сайт**

Секциите трябва да се различават от всички други средства за навигация и хипервръзки в сайта. При избор те трябва да се променят (цветово, графично), за да се ориентира потребителят коя секция е избрал. Някои секции могат да имат и подсекции – например в случая, посочен на фиг. 11, – секцията “Сервиз и поддръжка” може да се раздели на две подсекции.

*Инструментите на сайта* са хипервръзки към важни елементи на сайта, които не са част от йерархичното съдържание – такива инструменти са например количката за пазаруване, картата на сайта, помощната информация, форумите и пр. Тези инструменти са различни за различните сайтове, но общото между тях е, че не трябва да изпъкват толкова, колкото секциите. Добра практика, която налагат някои сайтове напоследък, е инструментите да са обособени в отделна група (фиг. 12).




**Фигура 12. Инструменти на сайт**

*Страницата* е част от цялостната структура на сайта, която потребителят получава, когато щракне върху някоя хипервръзка. Страниците са също така основна част от йерархията. Задължителните елементи, които трябва да съдържа една страница, от гледна точка на потребителски ориентираното проектиране, са:

– Заглавие (име) на страницата – не е достатъчно да се подчертае името на страницата в навигацията. Заглавието трябва да се откроява ясно, за да се ориентира дори потребител, който е попаднал на тази страница чрез препратка от друг сайт, а не чрез навигация от началото на сайта.
– Заглавието трябва да е разположено там, където потребителят очаква, и така, че уникалното съдържание на страницата да бъде под него.
– Името трябва да е с ясно отчетлив шрифт и да съвпада с думите от хипервръзката или бутона, който е натиснат за да се отвори страницата. Може да има вариации и малко разминаване, но контекстът трябва да е ясен и да не озадачава потребителя (фиг. 13) .


**Фигура 13. Името на страницата Get Started On Facebook е малка вариация на хипервръзката Getting Started (facebook.com)**

– Страниците трябва да са разделени на ясно обособени области. Това позволява на потребителите бързо да преценят към кои области да се насочат и кои не ги интересуват. На посочения пример (фиг. 13) ясно се вижда, че уникалното съдържание на страницата е разположено под името й и не предизвиква никакви

съмнения относно предназначението на тази страница в цялостната структура на сайта. Ако потребителят желае да се регистрира, то тогава обръща специално внимание на заграденото със сиво поле за регистрация. Ако това не го интересува или вече е регистриран, преминава на друга страница.

### 3.3. Полезност

Функциите, реализирани в даден сайт, имат определена полезност от гледна точка на потребителите. Първото условие, за да бъде оценена полезността, е функциите да са много ясно обозначени. Още при влизане в сайта потребителят трябва да може да се ориентира това сайт за търговия ли е, за услуги и справки, за развлечение и игри, или за нещо друго.

От гледна точка на потребителски ориентирания подход при определяне на функционалността, тя трябва да съответства възможно най-точно на представите, нагласата и очакванията на крайните потребители, за които е предназначен разработвания сайт. Ако сайтът е предназначен например за обяви за работа, то всяка от функциите трябва да е назована по лесно разбираем и обичаен за крайните потребители начин: “свободни работни места”, “предлагам работа”, “търся да наема”. Една и съща функция трябва да се обозначава с едно име в целия сайт и то да насочва много точно потребителя. Всякакви двусмислици, неясни думи или жаргони само биха объркали и отблъснали потребителя.

### 3.4. Контрол на потребителя

Потребителски ориентираният подход при изграждането на сайт изисква сайтът да дава възможност на потребителя във всеки един момент да се чувства, че може да контролира ситуацията. Това означава, че може да намери подходяща входна точка за начало на работа в сайта, може да прекъсне дадена операция, може да затвори страницата и да се върне към предишната или да напусне сайта. Всички тези действия трябва да имат точно означение и най-добре е то да съвпада с общоприетите представи, наложили се в Интернет. Входната точка за регистрация в страницата да е *Регистрация* или *Вписване* (Sign In – за англоезичен интерфейс на сайта), за изход и затваряне на страница е най-добре да се използва *Изход* и *Отписване* ( Exit и Sign Out – за англоезичен интерфейс на сайта). Когато страниците не изискват регистрация за четенето им, да се посочва ясно статусът на потребителя – като анонимен посетител или гост. Ако потребителят се регистрира със собствено име и парола, то тогава в следващите страници е добре да се появява неговото име с поздрав или обръщение. Това прави добро впечатление на потребителите и те с удоволствие ще се върнат отново в този сайт, който се обръща към тях на малко име. Това е един пример как минимални усилия, направени от дизайнерите и програмистите на сайта, се отплащат многократно.

Друг елемент от контрола върху сайта, който напоследък също се предоставя на потребителя, е възможността той сам да избира цветовото решение, големината на шрифта и да настройва страницата спрямо своя собствен монитор (фиг. 14). Тези елементи се обединяват в отделен блок – контрол или *page options*, който не е нуж-

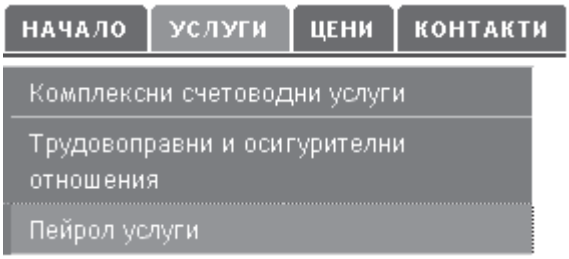
но да изпъква много. Ако потребителят не харесва сайта в цветовата тоналност по подразбиране, той може да промени цветовата гама.


**Фигура 14. Лента за контрол на потребителя върху елементи от страницата**

### 3.5. Език и съдържание

Потребителски ориентираният подход изисква сайтът да се изражда с кратки параграфи, ясен и точен език. Използваните термини трябва да са ясни и навсякъде в сайта да се използват с едно и също значение. Описанията на целите и задачите на системата трябва да са стегнати и кратки, излишните думи създават "шум" и пречат на потребителите да се съсредоточат върху задачата си. Ако са необходими повече обяснения, те могат да бъдат обособени в отделна страница, където ще бъдат прочетени само от тези потребители, които действително желаят да сторят това, чрез бутон "още по темата" или "повече информация".

Особено внимание трябва да се обръща на езика при информационните системи, които извършват услуги като: съвети за данъчно облагане, социално осигуряване, обучение, безопасност на стоки и пр. Трябва да се избягват термините, които нямат общоприето значение за потребителите, като например "Пейрол услуги" (фиг. 15). В такива случаи потребителят трябва да търси допълнителна помощ, за да се справи със задачата си, но по-вероятно е да потърси друг сайт, където ще пише просто "Изчисляване на заплащането на труда".




**Фиг. 15. Пример за използване на неясни термини в сайт**

Съкращенията също трябва да се използват много внимателно, тъй като те са много специфични за конкретната предметна област и не всеки потребител може да знае техните значения. Например в сайт, който служи на потребителите като ориентир за намиране на най-подходящите за тях финансови кредити (фиг. 16), съкращението ГПР предизвиква недоумение, ако не е пояснено.

| Банка | ГПР | Изисквания за дохода | Обезпечение | Наименование на кредита |
|---|---|---|---|---|

**Фигура 16. Използване на съкращения**

Потребителят е принуден допълнително да търси помощна информация, за да разбере, че става въпрос за "реалните разходи по кредита на годишна база", което обаче пък изобщо не отговаря на съкращението. В този случай съкращението ГПР не може да се избегне, но би трябвало да се поясни веднага под самата таблица.

### 3.6. Онлайн помощ и ръководство за потребителя

Задължително условие при потребителски ориентираното проектиране е към всяка информационна система да има подходяща онлайн помощ и подходящо ръководство за потребителя. Тук важи правилото за ненатрапчивост. Потребителят трябва да намери помощ или да прочете ръководството, само когато счете това за необходимо. Така или иначе той няма да го направи предварително, преди да е възникнал проблем, с който не може да се справи.

В някои случаи обаче не е нужно да се пестят помощни обяснения. Едва ли някой потребител би разбрал значението на бутона  , ако не е изписано отстрани какво означава. А още по-добре е да се използват общоприети бутони с ясни обозначения върху тях, за да не се нуждае потребителят от помощ при разгадаването им.

### 3.7. Системна и потребителска обратна връзка

Потребителски ориентираното проектиране изисква потребителите да могат да изпратят запитване по електронната поща, когато е необходимо, и също така да получат обратен отговор навреме.

Необходимо е така също да има екран за потвърждение, че личните му данни са получени, при попълване на формуляри, бланки и пр. В никакъв случай не трябва да се оставя потребителят в недоумение дали адресът му, номерът на кредитната му карта или телефонът му са попаднали точно там, където трябва. В много случаи потребителите използват първо фалшиви данни, за да се уверят, че сайтът работи коректно с личната им информация, че получават веднага обратна връзка и няма да се подведат.

Особено важно за потребителите е на сайта да има посочена дата на *последно обновяване* (Last update). По този начин те ще избегнат стара и неактуална информация. Когато актуалността е особено важна – в случаите, когато сайтът е предназначен за търговия (цените се менят много динамично) или услуги (където промяната на законовата уредба може да направи услугата неактуална или непълна), то в страниците трябва да е заложено автоматично обновяване на съдържанието през определен период от време.

### 3.8. Достъп до мрежата

При проектирането на информационни системи за Интернет трябва да се отчитат най-разпространените в момента версии на браузъри, операционни системи и технически характеристики на използваните компютри. В най-добрия случай сайтът трябва да може да се преобразува за работа с различни хардуерни, софтуерни платформи и браузъри, без да губи от ефективността и функционалността си. Това, раз-

бира се, не винаги е възможно. Достатъчно е да отговаря на изискванията на най-разпространените в момента програмни средства и компютърни архитектури и периодично да се актуализира, когато те се променят. Данни за това кои са в момента най-използваните браузъри от потребителите, с какъв размер и разрешаваща способност са най-често използваните монитори, могат да бъдат намерени в Интернет. Тази информация трябва да се използва, за да се настрои подразбиращата се версия на сайта спрямо масово използваните технически и програмни средства.

### 3.9. Съгласуваност

За да се използва коректно даден сайт, той трябва да е така проектиран, че потребителите да разчитат на разпознаването и съгласуването между отделните действия, а не на запаметяването къде се намират. Информацията и действията в секциите трябва да засягат един и същ обект. Ако има реклами, хипервръзки към външни системи, те трябва да са ясно обозначени, като най-добре е да бъдат оформени в отделни блокове.

Съгласуваността изисква между отделните части на информационната система да съществува връзка. Ако например една страница съдържа информация за "Строителни услуги", то потребителят би очаквал да намери информация не само за списъка с услуги, но и за цените им и за фирмите, които ги извършват. При липса на такава обвързаност потребителят започва да преглежда последователно опциите на главното меню в търсене на страниците с полезната за него информация. Това принудително разучаване на сайта обаче най-често води до отказ от ползването му и търсене на нужната информация (ползване на нужната услуга) на друго място в мрежата.

### 3.10. Съобщения за грешки и корекции

Информационната система, изградена на потребителски ориентирания подход, трябва да разпознава голямо разнообразие от действия на потребителя, при което не се налага често да се появяват съобщения за грешки. Съобщенията и предупрежденията за грешки могат да бъдат разделени на три групи (Денчев, Емил и др., 2007): предупредителни съобщения; съобщения за прогрес на операция (при по-продължителни операции); съобщения за приключване на операция; съобщения за предупреждения за грешка.

Съобщенията за грешки са специфични за всяка система, но при тяхното формулиране се препоръчва:

- Системата да дава кратки инструкции за възможните действия на потребителя, включително и формата на данните при въвеждане.
- Съобщенията за грешки да са видими (не скрити), на понятен език, без технически параметри.
- Съобщенията за грешки да описват действията за премахване на проблема и да осигуряват ясен изход от ситуацията.
- В случай на нужда съобщенията за грешки трябва да осигуряват информация за контакт с разработчика.

# 4. ПРИЛОЖЕНИЕ НА UML ПРИ ПОТРЕБИТЕЛСКИ ОРИЕНТИРАНОТО ПРОЕКТИРАНЕ НА ИНФОРМАЦИОННИ СИСТЕМИ

## 4.1. Обща характеристика на UML

През 80-те и 90-те години на 20-ти век, заедно с развитието на *обектно ориентираните езици за програмиране* (като C++), започват да се развиват и ***графични системи*** (наричани "моделни нотации" или "графични езици") за ***обектно ориентирано моделиране***.

Едни от най-популярните методи за обектно ориентирано моделиране от този период са:

- *Object-modeling technique (OMT),* разработен от Румбау (Rumbaught),
- *Booch's method,* разработен от Грейди Буук (Booch)
- *Object-oriented software engineering (OOSE)* – метод, разработен от Айвър Джейкъбсън (Jacobson)

През 1996 *Rational Software Corporation* стига до заключението, че наличието на много езици за моделиране забавя развитието на обектно ориентираната технология; затова възлага на Грейди Буук, Айвър Джейкъбсън и Джим Румбау да създадат една обща система от типове диаграми. Проектът включва не само изобретените от тях нотации, но и разработки на други автори. Например, включен е методът ***OOram*** (Object Oriented Role Analysis Method – Метод на обектно ориентирания ролеви анализ) разработен през 1996 г. от Риинскуг (Reenskaug), от Университета в Осло, Норвегия.

През 2005 г. UML (версия 1.4.2) е приет за стандарт на международната организация за стандарти ISO 19501 (2005) Information technology - Open Distributed Processing - Unified Modeling Language (UML).

UML (*Unified Modeling Language* - Унифициран език за моделиране) е графичен език за специфициране, конструиране и документиране на информационни системи. UML позволява *еднородно* (*унифицирано*) моделиране на информационни системи, в два аспекта:

- UML е препоръчителен стандарт в софтуерната индустрия, възприет от много водещи фирми. Това позволява системните модели, разработвани от различни екипи, да бъдат описани с еднакви графични средства и респективно – разбираеми за голям кръг от специалисти извън разработващия екип.
- UML е независим от конкретните програмни езици и това създава възможност да се моделират по единен начин системи, които са много различни от гледна точка тяхната програмна реализация.

UML дефинира правилата за изграждане на различни *типове диаграми* (графични модели), които служат за графично представяне на различни аспекти на информационните системи.

По отношение на *детайлността на моделирането* се разграничават три нива:

- ***Моделиране на високо ниво*** (*High level modeling*). На това ниво се представя системата като цяло и нейните основни модули. При създаване на нова система обобщеният модел се използва като начална скица на системата. Ето защо това ниво се нарича също *ниво на скица* (sketch).

– ***Детайлно моделиране*** (*Detail modeling*). На това ниво се разработват модели, които показват детайли от структурата и функционирането на отделни подсистеми и модули.
– ***Моделиране на ниво на програмиран код*** – На това ниво се разработват диаграми на класовете (class diagrams). Една диаграма на клас представя в графичен вид характеристиките на един клас в обектно ориентиран език за програмиране (свойства, методи, отношения на наследяване).

Основната цел на UML е да се предостави една добре развита система от средства за моделиране, която да не е зависима от конкретен език за програмиране. UML моделите, създавани за една система от даден екип, са разбираеми за софтуерни специалисти (проектанти и програмисти) от други екипи.

За всеки тип UML диаграми са дефинирани нотация и мета-модел.

*Нотацията* (notation) са графичните елементи, който се виждат в моделите; това е графичният синтаксис на езика за моделиране. Например нотацията за диаграма на клас дефинира как се представят елементи и концепции като клас, асоциация и множественост.

*Мета-модел* (meta-model) в UML се нарича формалното дефиниране или неформалното описание на основните понятия използвани в UML диаграмите. Официалната спецификация на езика UML съдържа формални дефиниции на основните понятия при всеки тип диаграми. Но в повечето практически ръководства начинът на използване на графичните нотации и тяхната интерпретация се обясняват чрез интуитивни, неформални описания и примери.

Основна област на приложение на UML е проектирането на нови системи. UML позволява при големи проекти, с участието на много специалисти, всички части от проекта да са разработени по еднакъв начин. Освен това UML позволява лесно да се обменя опит между различни екипи и да се използват най-добрите практики.

Друга сфера на приложение на UML са случаите, когато се налага сложни системи да се анализират и поддържат от специалисти, които не са участвали в разработката. Целта е, въз основа на програмния код, наличната документация (но не във вид на UML диаграми) и фактическото функциониране на системата, да се създаде модел на системата със средствата на UML. Този вид дейност се нарича *реверсивно софтуерно инженерство* (reverse software engineering).

Използването на UML на ниво на програмен код (диаграми на класовете) е съществен фактор в развитието на съвременните средства за обектно ориентирано програмиране. Разработват се специални програмни средства, които могат автоматично да трансформират клас-диаграмите в програмен код на определен обектно ориентиран език за програмиране (примерно C#, C++, Java). Тези програмни средства се наричат **CASE** (Computer-Aided Software Engineering) инструменти. Разработчиците чертаят UML диаграми на класовете, които се компилират директно в изпълним код. Очевидно, тази употреба на UML поставя много високи изисквания към CASE инструментите.

### 4.2. Приложим ли е UML при потребителски ориентираното проектиране?

Днес UML се използва предимно като инструмент на *моделиране на бизнес процесите*, т.е. *на средното ниво* на една информационна система с трислойна архитектура, каквито са мнозинството от съвременните информационни системи. (Цанева,

Моника и др., 2007; Велев, Димитър, и др., 2007, Стефанова, Камелия и др., 2007). Но от факта на успешното приложение на UML за моделиране на бизнес процесите е погрешно да се направи изводът, че UML е приложим *единствено за тази цел.*

В следващите параграфи на този раздел ще покажем, че UML *е достатъчно мощен и абстрактен език за моделиране, който може успешно да се прилага и при проектиране на нивото на потребителския интерфейс.*

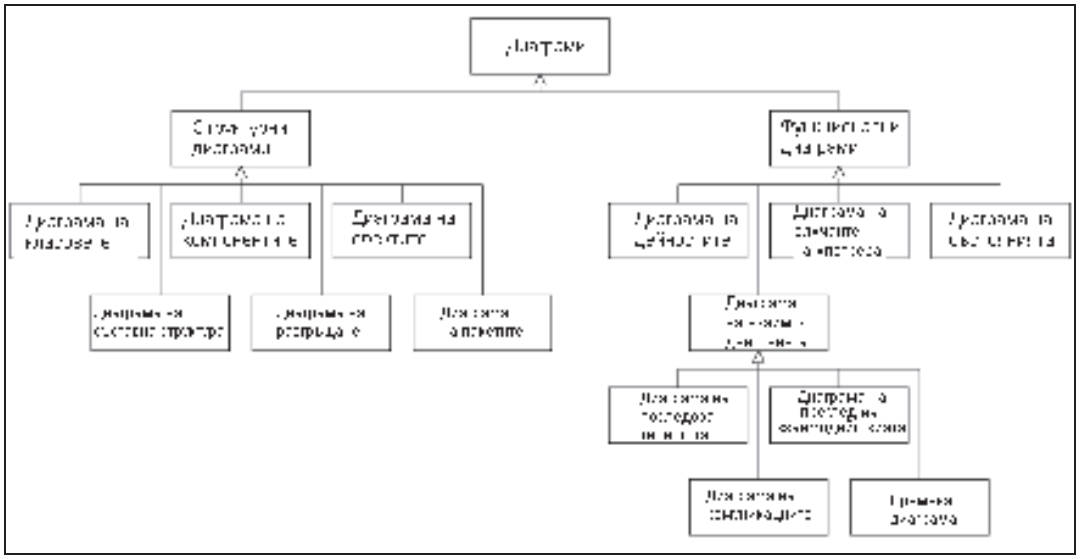
### 4.3. Класификация на моделите и типовете UML диаграми

Общият модел на една информационна система се разглежда като изграден от два взаимно допълващи се модела:

a) ***Структурен модел*** (*Structural model),* който показва структурата на системата и подсистемите, използвайки обекти, атрибути, операции и връзки.

b) ***Функционален модел*** (*Functional model*), наричан още *динамичен модел* (*Dynamic model),* който показва функционалността на системата и измененията на системата във времето.

Всяка UML диаграма е *частично графично представяне* на един от двата системни модела. В UML 1.0 от 1997 г. са дефинирани 10 типа диаграми. В UML 2.0 от 2004 г. са добавени още три типа диаграми. Класификацията на типовете диаграми е представена в графичен вид на фиг. 17.



**Фигура 17. Класификация на типовете диаграми**

#### 4.3.1. Структурен модел и структурни диаграми

Диаграмите, които представят аспекти на структурния модел, се наричат *структурни диаграми*. Към структурните диаграми се отнасят 6 типа, посочени в таблица 1.

Първите три типа диаграми представят *логическата структура на системата* в термините на обектно ориентираното проектиране и програмиране:

- Диаграми на класовете ***(Class diagrams***)
- Обектни диаграми ***(Object diagrams***)
- Диаграми на съставна структура ***(Composite structure diagrams***)

Следващите три типа диаграми представят общата *структура на информационната система:*

- Диаграми на пакетите ***(Package diagrams***)
- Компонентни диаграми ***(Component diagrams***)
- Диаграми на разгръщане на системата ***(Deployment diagrams***)

Последните три типа диаграми нямат приложение при проектирането на потребителския интерфейс и затова няма да ги разглеждаме.

**Таблица 1.**

Типове структурни диаграми в UML 1 и UML 2.

| Типове диаграми | Предназначение | Произход |
|---|---|---|
| Диаграма на класовете (*Class diagram* ) | Описва свойства и методи на класовете | UML 1 |
| Диаграма на обектите (*Object diagram* ) | Представя обектите (инстанции на класове) в една система в даден момент | UML 1 |
| Диаграма на съставна структура (*Composite structure diagram*) | Изобразява вътрешната структура на класовете | Ново в UML 2 |
| *Диаграми на общата структура на информационната система* (нямат приложение при проектирането на потребителския интерфейс) | | |
| Диаграма на пакетите (*Package diagram*) | Йерархична структура по време на компилиране | UML 1 |
| Компонентна диаграма (*Component diagram*) | Структура и връзки на компоненти | UML 1 |
| Диаграма на разгръщане (*Deployment diagram*) | Показва физическата схема на една система | UML 1 |

### 4.3.2. Функционален модел и функционални диаграми

Диаграмите, които представят аспекти на функционалния модел, се наричат *функционални диаграми* или *поведенски (behavioral) диаграми*. Към функционалните диаграми се отнасят следните 7 типа (табл. 2):

- Диаграми на дейностите ***(Activity diagrams***)
- Диаграми на случаите на употреба ***(Use case diagrams***)
- Диаграми на състоянията ***(State machine diagram***)

И четири типа *диаграми на взаимодействията:*

- Диаграми на последователностите ***(Sequence diagrams***)
- Диаграми на комуникациите ***(Communication diagrams***)
- Диаграми за преглед на взаимодействията ***(Interaction overview diagrams***)
- Времеви диаграми ***(Timing diagrams***)

**Таблица 2.**

Типове функционални диаграми в UML 1 и UML 2

| **Типове диаграми** | **Предназначение** | **Произход** |
|---|---|---|
| Диаграма на случаите на употреба (*Use case diagram*) | Как потребителите взаимодействат със системата | UML 1 |
| Диаграма на дейностите (*Activity diagram*) | Процедурно и паралелно поведение | UML 1 |
| Диаграма на състоянията (*State Transition diagram*) | През какви състояния преминава даден обект.<br>Друго название State Machine diagram | UML 1 |
| *Диаграми на взаимодействията* | | |
| Комуникационна диаграма (*Communication diagram*) | Взаимодействие между обекти; ударение върху връзките<br>Предишно название Collaboration diagram (диаграма за сътрудничеството) | UML 1 |
| Диаграма на последователностите (*Sequence diagram*) | Взаимодействие между обекти; ударение върху последователностите | UML 1 |
| Времева диаграма (*Timing diagram*) | Взаимодействие между обекти; ударение върху синхронизацията | Нова в UML2 |
| Диаграма за преглед на взаимодействие (*Interaction overview diagram*) | Комбинация от диаграми на последователностите и диаграми за дейностите | Нова в UML 2 |

## 4.4. UML диаграми, приложими за описание на действията на потребителите

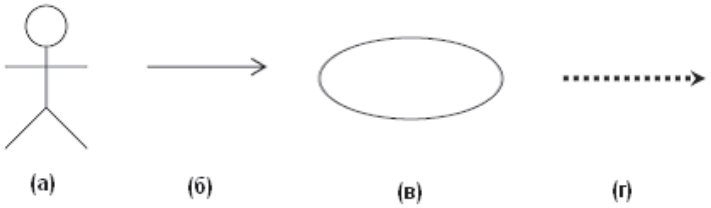
### 4.4.1. Диаграма на случаите на употреба

Диаграма на случаите на употреба се използва за специфициране на функционалните изисквания към разработваната информационна система, от гледна точка на различните типове потребители и външни клиенти. Състои се от актьори (actor), случаи на употреба (use cases) и връзките между тях (фиг. 18).

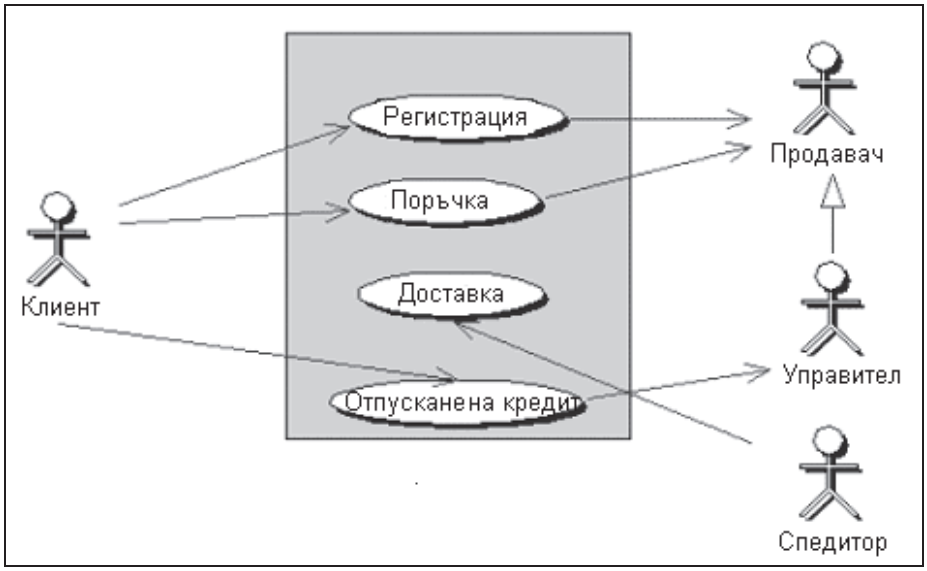
Актьор е някой или нещо извън системата, което взаимодейства със случаите на употреба, написани за тази система.

Сценарий (scenario) е серия от стъпки, описващи взаимодействието между потребител и система. Случай на употреба представлява набор от сценарии, свързани помежду си от обща потребителска цел.

Use case (communication) relationship показва връзката между актьори и случаи на употреба. Фигура 19 показва елементите на диаграмите на случаи на употреба.




**Фигура 18. Нотация на диаграмите на случаите на употреба: (а) Актьор; (б) Отношение между актьор и случай на употреба; (в) Случай на употреба; (г) отношение между случаи на употреба – отношение на включване или разширение.**




**Фигура 19. Пример за диаграма на случаите на употреба**

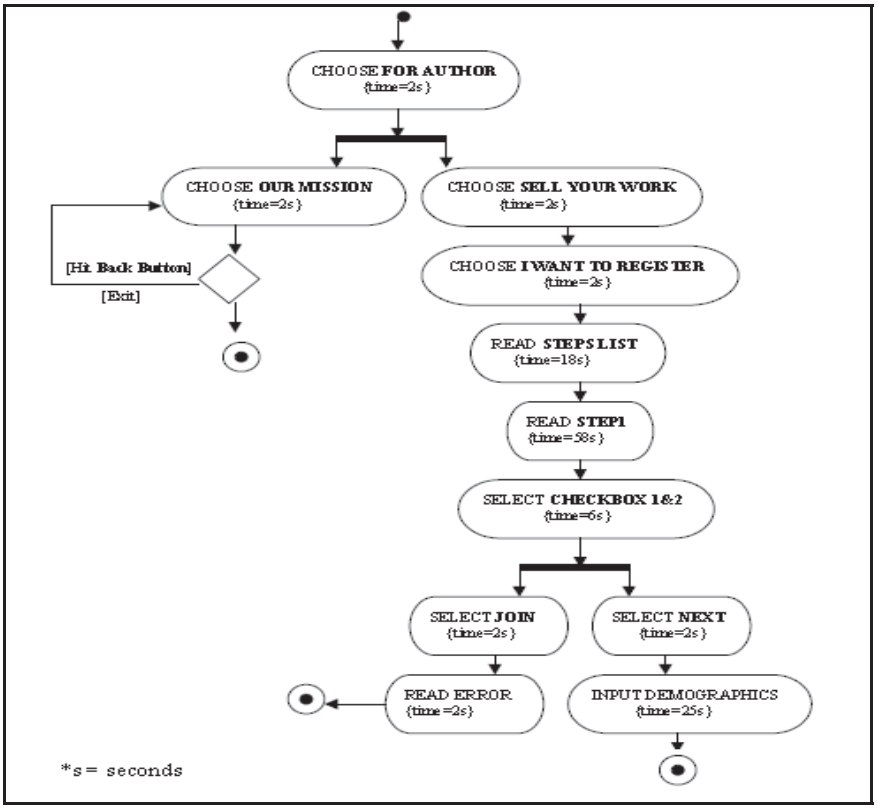
### 4.4.2. Диаграма на дейностите

Диаграмите за дейност изобразяват някакъв относително сложен комплекс от дейности чрез графично представяне на последователност от действия. Този тип диаграми е наследник на ***алгоритмичните блок-схеми*** (***flowchart***).

Диаграмата има един символ за начало и един символ за край. Действията се изобразяват като правоъгълници, а преходите от едно действие към друго – със стрелки. Условното разклонение, както в блок-схемите, се означава с ромб. Символът "вилица" (*fork*) се използва за изобразяване на паралелно разклоняване на процесите. Удебелената черта, но с няколко входящи стрелки и една изходяща, означава обединяване на процеси в един общ процес (*join*). Нотацията, използвана за диаграмите за дейност, е показана на фиг. 20.


**Фигура 20. Нотация на диаграмите на дейност**

В рамките на блока за действие може да се посочва и допълнителна информация. В следващата диаграма например е посочено средното време, за което потребителят извършва отделните стъпки при работа със системата.



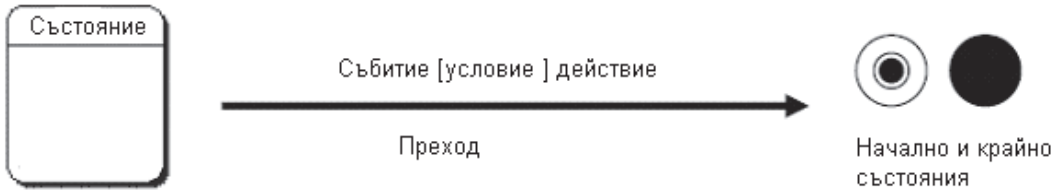

**Фигура 21. Пример за нотация на диаграмите на дейност с отбелязано времетраене (Dewalt, Craig ,1999, p. 29)**

Диаграмата за дейност може да се раздели на *ивици (swimlane*), които съдържат действия, осъществявани от различни изпълнители (лица или програмни модули).

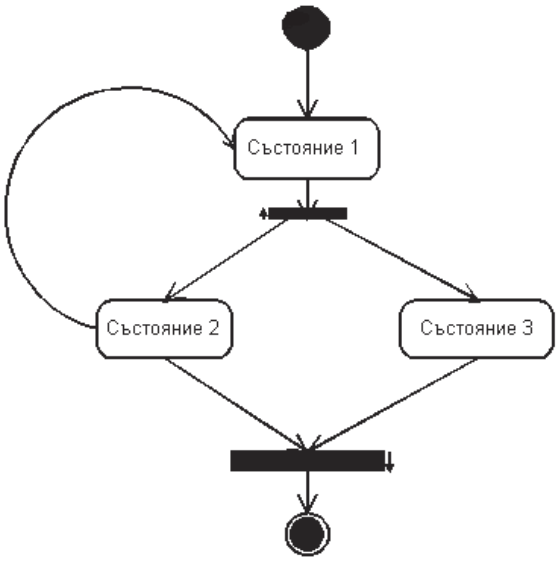
### 4.4.3. Диаграма на състоянията

Диаграмата на състоянията представя последователността от състояния, през които преминава даден обект. Това е една силно разширена версия на нотацията за *диаграма на състоянията и преходите на крайни автомати*, използвана в математическата теорията на автоматите.

Най-често диаграмите се използват за представяне на последователните състояния, през които преминава обект от даден програмен клас. ***Състоянията*** (states) на обекта се представят като правоъгълници. ***Преходите*** (transition) от едно състояние към друго се представят чрез стрелки. Обектът има едно начално състояние и няколко крайни състояния. ***Тригерите*** (trigger events) са събития, които инициират даден преход (респективно, ако събитието не възникне, преходът е невъзможен).

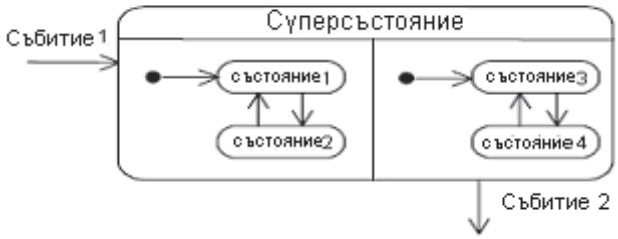



**Фигура 22. Нотация за диаграмите на състоянията**




**Фигура 23. Пример за диаграма на състоянията**

С този тип диаграми могат да се описват и абстрактни обекти с голяма степен на обобщение, примерно състоянията на даден модул. Няколко състояния могат да бъдат обединени в едно ***суперсъстояние*** (superstate).




**Фигура 24. Пример за *диаграма на състоянията* със суперсъстояние**

### 4.4.4. Диаграми на взаимодействията

При *диаграмите на взаимодействията* се предполага, че дейността на системата може да се разглежда като колективна дейност на множество от относително обособени *участници* (actors). Като "участници" могат да се дефинират програмни компоненти. Това може да са големи подсистеми, но може да са отделни инстанции на даден клас. Понякога като "участници" се дефинират и лица, взаимодействащи със системата (потребители, администратори). Диаграмите на взаимодействията представят графично взаимодействията между различните участници. "Взаимодействията" най-често са обмяна на съобщения: заявки към даден участник да извърши дадено действие и съобщения за завършване на дадено действие от участник.

UML включва 4 типа диаграми на взаимодействията:

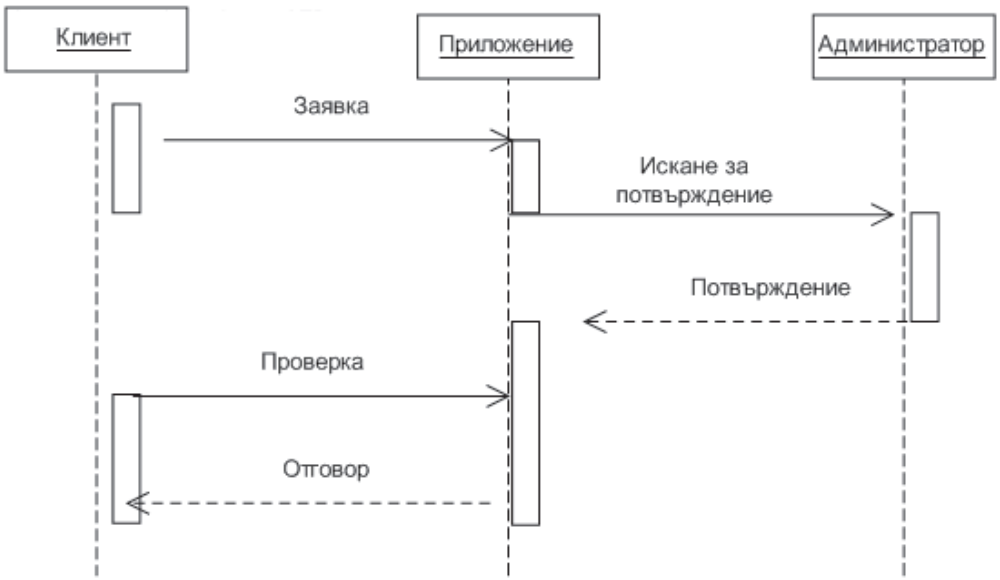
- *Диаграма на комуникации* (communication diagrams)
- *Диаграма на последователността* (Sequence diagram)
- *Времева диаграма* (Timing diagram) [UML2]
- *Диаграма за преглед на взаимодействията* (Interaction overview diagram) [UML2]

### 4.4.5. Диаграма на комуникациите

Диаграмата на комуникациите (communication diagrams) позволява свободно разполагане на правоъгълници, символизиращи участници, чертане на стрелки за визуализиране начинът на свързване на участниците и използва номериране за последователността на съобщенията. В UML 1.x тези диаграми се наричат *диаграми на сътрудничеството (collaboration diagrams).*

### 4.4.5. Диаграма на последователността

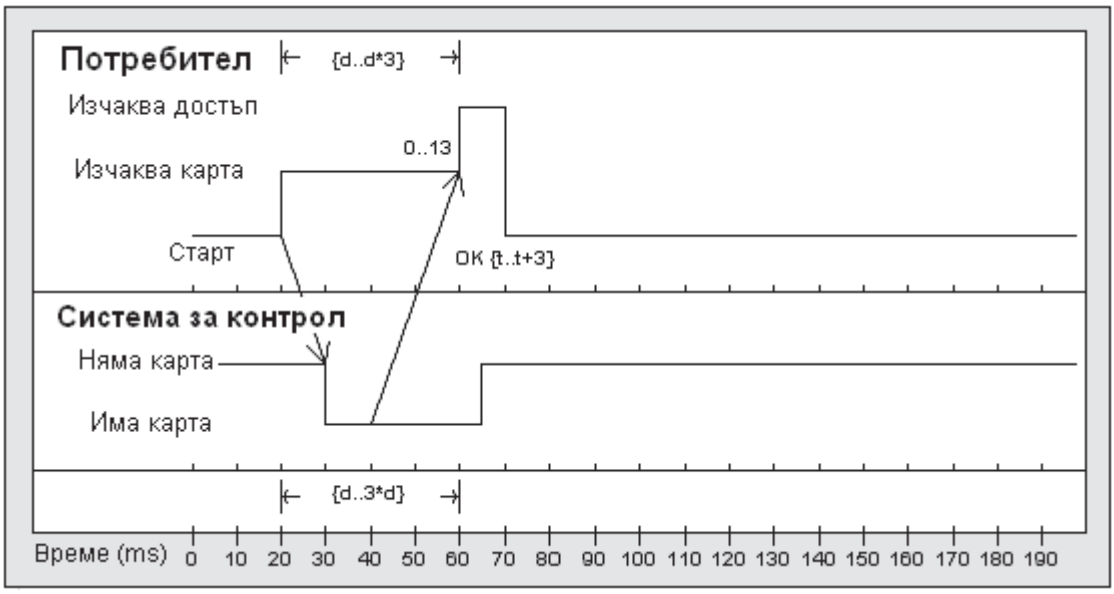
Диаграмата на последователността (Sequence diagram) показва взаимодействията между обектите, подредени във времева последователност. Оста на времето е отгоре надолу.




**Фигура 25. Пример за диаграма на последователността**

### 4.4.6. Времева диаграма

Времевата диаграма (Timing diagram) е *специален вид диаграма на последователност*. Времевата диаграма показва промените в състояние или условие на един или повече обекти през даден период от време. Оста за графично изобразяване на времето е разположена *отляво надясно* и е оразмерена с времеви интервали (за разлика от диаграмите на последователност, където оста на времето е отгоре надолу и показва само последователността на събитията). Състоянията или условията се изобразяват като времева линия, отговаряща на съобщените събития.



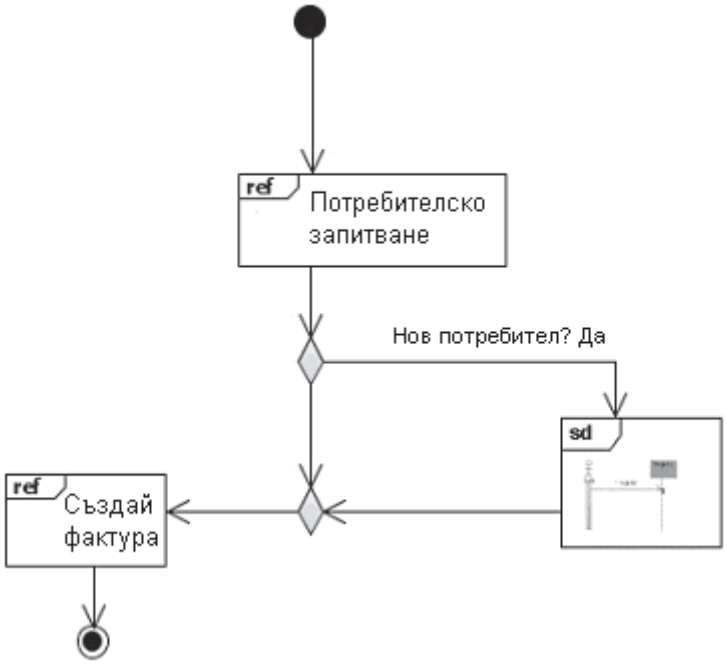

**Фигура 26. Пример за времева диаграма**

Времеви диаграми се използват при проектиране на софтуер, при който е важно да се спазват точно времевите интервали между събитията; това е предимно софтуер за вграждане в технически устройства (embedded software) или за системи за управление на производствени процеси в реално време (real-time systems).

### 4.4.7.Диаграма за преглед на взаимодействията (Interaction overview diagram)

Диаграмата за преглед на взаимодействията дава възможност за представяне на връзки между диаграми на взаимодействията от трите типа: диаграма на комуникациите, диаграма на последователността, времева диаграма.

Методът за конструиране на диаграмата е подобен на този на *диаграмата на дейностите;* използват се същите моделиращи елементи – стрелки за преход, ромб за условен преход и правоъгълници. Всеки от правоъгълниците представя една диаграма на взаимодействията. Диаграмата или е нарисувана в правоъгълника, или в правоъгълника е показано името на диаграмата, а в горния ляв ъгъл има препратка [**ref**], чрез която в нов прозорец се показва съответната диаграма.




**Фигура 27. Пример на диаграма за преглед на взаимодействията**

### 4.4.8. Предимства и недостатъци

При проектиране на потребителския интерфейс от структурните диаграми най-често се използват:

- диаграми на пакетите (*Package diagrams*)
- диаграмите, които представят *физическата структура на системата* – компонентната диаграма (*Component diagram*) и диаграмите за разгръщане на системата (*Deployment diagram*)

От функционалните диаграми най-често се използват:

- диаграмите на случаи на употреба *(Use case diagrams)*
- диаграма на последователностите (*Sequence diagram)*

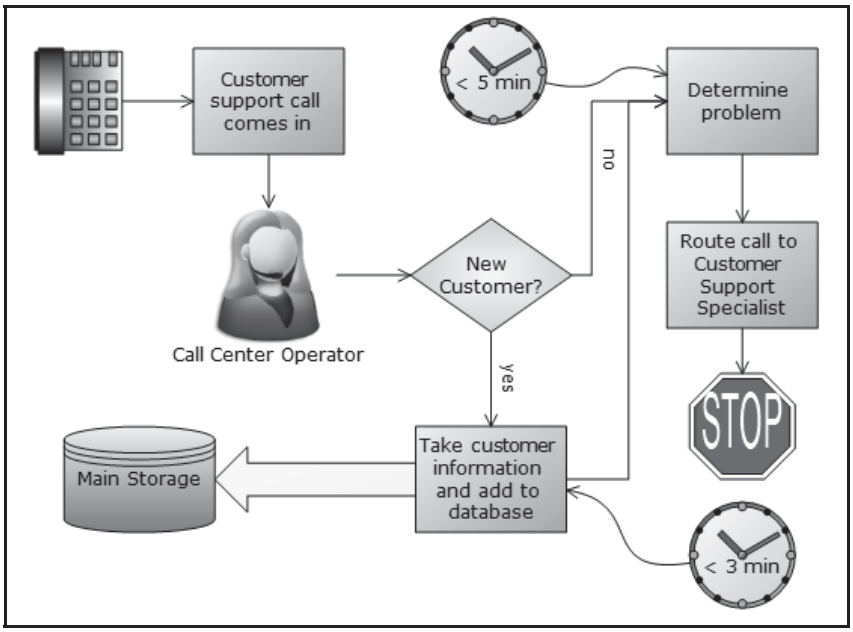
Диаграмите на случаи на употреба и диаграмите на последователностите са се наложили като задължителна част от проектите на всяка по-голяма информационна система. Те дават лесно разбираема картина на функционалността на системата от гледна точка на потребителите (в статичен и в динамичен аспект).

Диаграмите представят определена информация в нагледен и удобен за бързо възприемане вид, но не могат да заменят описателната документация.

Например *диаграмата за разгръщане на системата* показва *къде* се разполагат отделните програми. Но тя не съдържа информация за това как протича инсталационният процес, какви са изискванията към операционната система, към техническите устройства и други важни данни, които се описват в едно “ръководство за инсталиране на системата”.

По отношение на описанието на системата от потребителска гледна точка ситуацията е сходна. Use-case диаграмите и диаграмите на последователностите са много удобни, но не дават пълната информация за проектираното взаимодействие между потребителите и системите.

Понякога една диаграма със свободен формат може да е много по-впечатляваща от UML диаграмите. В диаграмата на следваща фигура са използвани някои елементи от диаграмите на последователностите (блок за разклонение, блок за действие и свързващи стрелки), съчетани с други лесно разбираеми символични означения. Такава диаграма обаче може да се използва при презентации, но не и за целите на систематичното описание на потребителския интерфейс на информационна система.

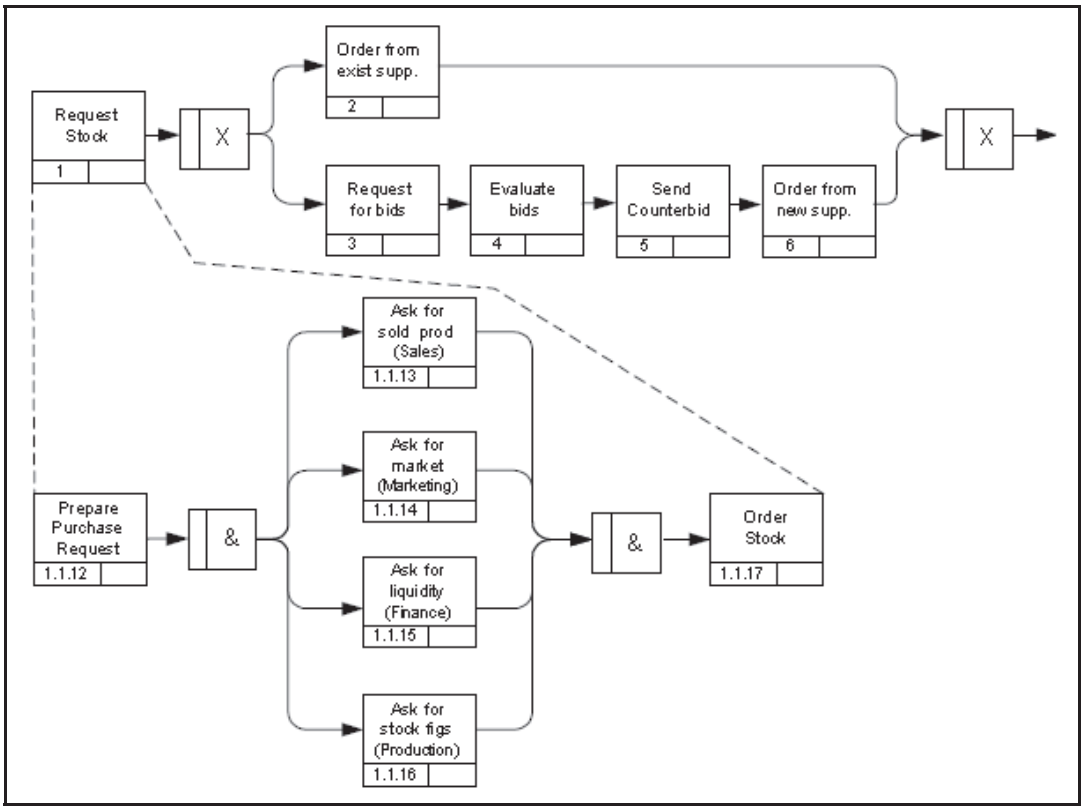



**Фигура 28. Пример за диаграма със свободен формат (Serlio Software Development Corp., 2007)**

Алтернатива на използването на UML, освен диаграма със свободен формат, е прилагането на някой от конкурентните методологии за графично моделиране. Например, в рамките на инициативата Integrated Computer Aided Manufacturing (ICAM), разработвана за Военновъздушните сили на САЩ, е създадена и методологията за графично моделиране IDEF (ICAM DEFinition). Ovidius Noran (2003) представя един обстоен сравнителен анализ между UML и IDEF. Тук ще се ограничим само да отбележим, че каквито и да са относителните предимства и недостатъци на двете системи за графично моделиране, UML следва да се предпочете при проектирането на информационните системи. Решаващият фактор за такъв избор е необходимостта от *единство* в проектната методология и респективно – *взаимно разбиране* между специалистите, разработващи отделните модули на информационни системи или взаимосвързани информационни системи. Дори дадена корпорация да не счита за необходимо да се съобразява със стандартите на ISO, то тя трябва да отчи-

та фактическия статут на UML като общоприета система за моделиране в сферата на проектиране на информационните системи.

Пример за IDEF *диаграма на процесите (process flow diagram*) е показан на следващата фигура.




**Фигура 29. Пример за IDEF диаграма на процесите. (Ovidius Noran, 2003, p. 43)**

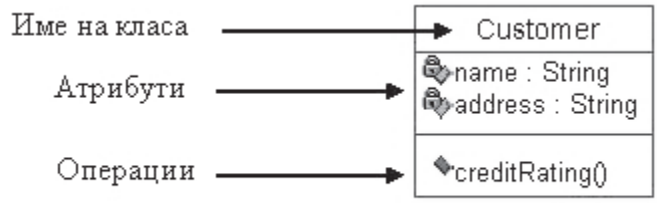
### 4.5. UML диаграми, приложими за моделиране на вътрешната организация на потребителския интерфейс

За моделиране на вътрешната организация на потребителския интерфейс се използват два вида диаграми:

– диаграми на класовете (class diagram) и
– диаграми на обектите.

#### 4.5.1. Диаграма на класовете

Класът е множество от обекти, които споделят обща структура (представя се чрез атрибути), общо поведение (представя се чрез операции), общи връзки и семантика. С други думи класът е шаблон за създаване на обекти. Всеки обект е инстанция на даден клас. Класът се представя като правоъгълник с три части: име, атрибути и операции.

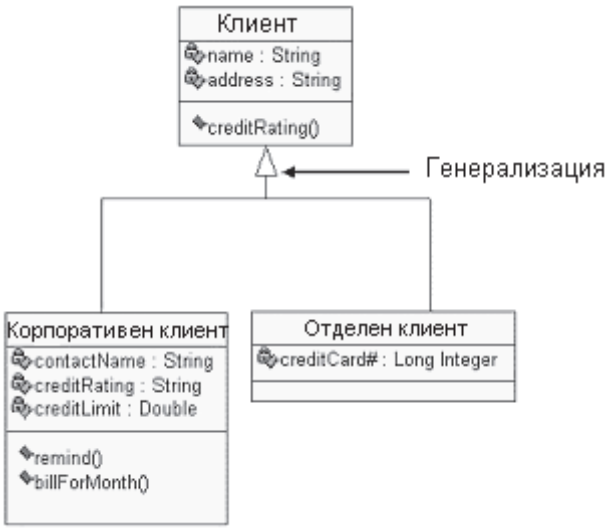



**Фигура 30. Пример за клас**

Диаграмата на класовете (class diagram) описва класове (в смисъла на обектно ориентираното програмиране) и различните видове взаимоотношения, които съществуват между тях.

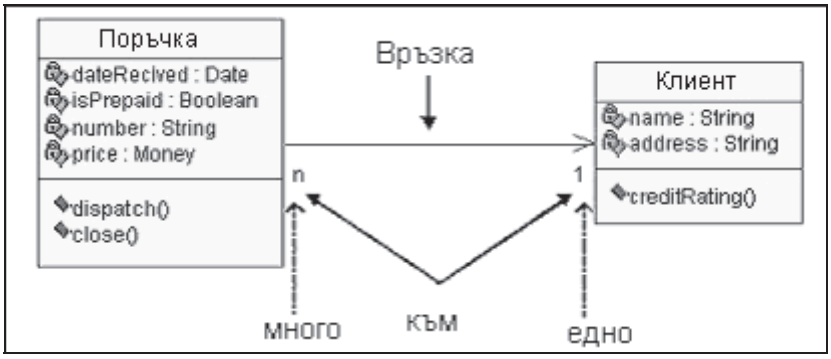
Основни типове връзки, представяни в диаграмите на класовете, са:

– *Генерализация* (*generalization*). От логическа гледна точка това е отношение вид към род. В примера родовото понятие е "*Клиент*" (Customer), а видовите понятия са "*Корпоративен клиент*" (Corporate customer) и "*Клиент - физическо лице*" (Personal customer). От гледна точка на обектно ориентираното програмиране това е отношение на наследяване между класовете.



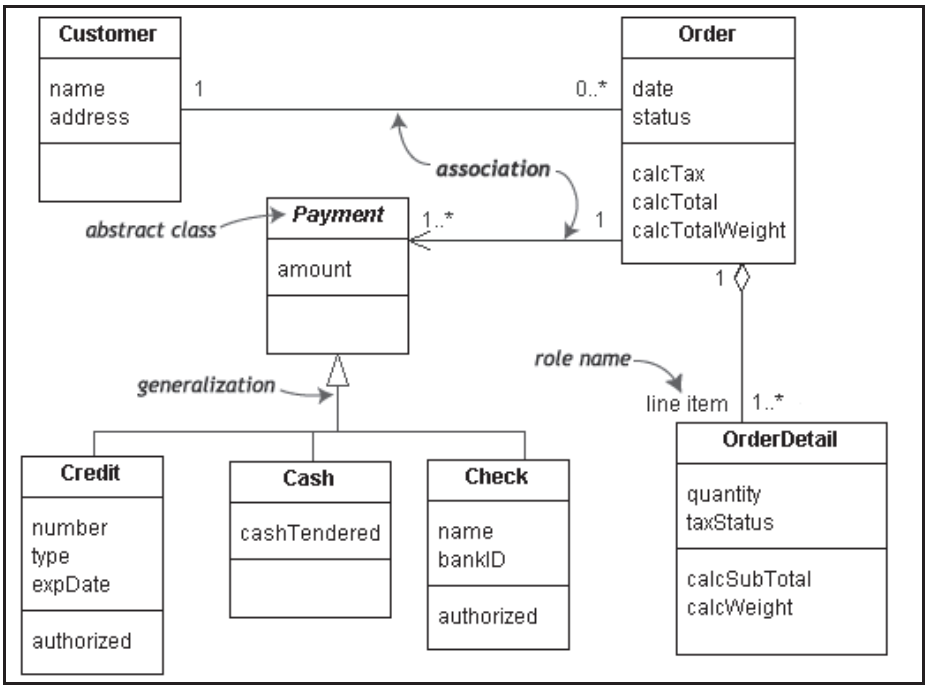

**Фигура 31. Диаграма на класове: пример за отношение на генерализация**

– *Асоциация (association).* Асоциациите могат да представят различни логически типове отношения между класовете. Ето защо, когато типът на отношението не е ясен, той се изписва. В примера на фиг. 32 типът на отношението между обектите от клас "Поръчка" (Order) и обектите от клас "Клиент" (Customer) не е изписан, но се подразбира че отношението е "*поръчка за клиент*". В двата края на стрелките, представящи асоциациите, се поставят означения за броя на обектите, които свързва асоциацията. В случая *n* (много) поръчки се отнасят за *точно един* клиент.




**Фигура 32. Диаграма на класове**

В горните две диаграми за всяко свойство на даден клас е посочен и типът на данните (*Boolean*, *String* и други). Когато диаграмите на класовете обхващат голям брой класове, тогава тази детайлна информация за класа се пропуска.

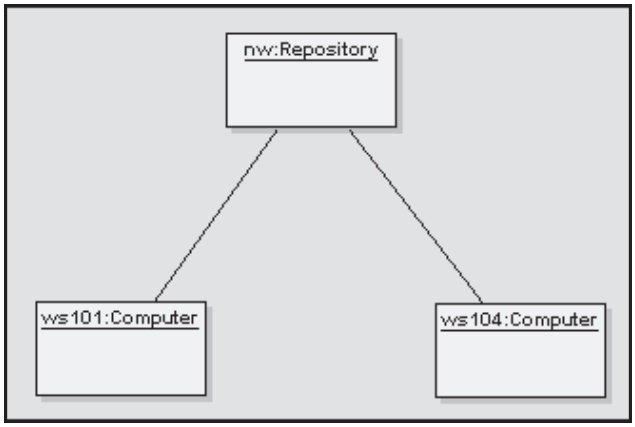



**Фигура 33. Пример за диаграма на класове (Randy Miller, online)**

### 4.5.2. Диаграма на обектите

Диаграмата на обекти (object diagram) представя *обектите* (т.е. инстанциите на класовете) в една система в определен момент. Тази информация не може да се представи с диаграмите на класовете, които показват дефинициите на класовете. Имената на обектите имат вида <идентификатор на обекта>:< име на класа>

На фиг. 34 е показана диаграмата на обектите, в която са представени:
- един обект (с идентификатор ***nw***) от клас *Repository* и
- два обекта (с идентификатори ***ws101*** и ***ws104***) от клас *Computer*.




**Фигура 34. Пример за диаграма на обектите**

Свойствата (атрибутите, методите) на обектите са определени от класа, към който принадлежат. Поради това те не са изобразени в диаграмата на обектите, а се предполага, че има лесна препратка към диаграмата на класа.

### 4.5.3. Предимства и недостатъци

Диаграмите на класовете и диаграмите на обектите изразяват аспекти на вътрешната структура на потребителския интерфейс, които много трудно могат да бъдат представени "описателно". Диаграмите са често единствената документация на класовете.

Друго голямо предимство е възможността диаграмите на класовете да бъдат използвани за автоматизация на програмирането.

CASE (Computer-aided software engineering) – компютърно-подпомагано софтуерно инженерство, се нарича използването на специализирани програми за подпомагане на програмирането и поддръжката на софтуера; респективно, програмните средства се наричат ***CASE*** *инструменти*.

Разграничават се два големи класа CASE инструменти:
- Структурни (Structured) CASE инструменти – които се базират на структурното (необектно ориентирано) програмиране. Тези продукти са характерни за периода до 1990-те години.
- Обектно ориентирани (ОО) CASE инструменти, които се развиват заедно с обектно ориентираното програмиране.

ОО CASE инструментите, които тук разглеждаме, са свързани с UML в два аспекта:
- предоставят средства за построяване на UML диаграмите;
- позволяват, въз основа на диаграмите на класовете, автоматично да се генерира програмен код на някои от популярните езици за обектно ориентирано програмиране.

Повечето съвременни ОО CASE инструменти поддържат езиците за машинно независимите платформи ***.NET*** (C# ; VB) и ***J2EE*** (Java). Съвременните CASE инструменти имат един недостатък, на който най-често се акцентира – тяхната висока цена. При избора на проектни и програмни инструменти разработчиците са принудени да се съобразяват с разчетите за икономическата ефективност на информационните системи (Лазарова, Ваня, Д. Петров. 2007). В таблица 3 са показани цени за четири CASE инструмента (за 1 лиценз за 1 работно място, при закупуване на 1 до 5 лиценза). Цените намаляват при закупуване на по-голям брой лицензи.

**Таблица 3**

Цени на лиценз за 1 работно място за някои CASE системи.
Актуалните цени могат да се видят на посочените сайтове на фирмите разработчици.

| CASE Продукт [Производител] | Генерира код на | | Цена на лиценз за 1 работно място |
|---|---|---|---|
| | ***J2EE*** (Java) | ***.NET*** (C# ; VB) | |
| Rational Rose XDE [IBM] http://www-306.ibm.com/software/ | + | + | $1200 до $3000 |
| Power Designer 12.0 [Sybase] http://www.sybase.com/products/ | + | + | $2,995 до $7,495 |
| Enterprise Architect [Sparx Systems] http://www.sparxsystems.com.au/ea.htm | -- | + | $239 |
| EclipseUML [Omondo] http://www.eclipseuml.com/ | + | -- | Безплатен (отворен код) |

Цените на CASE системите на водещите фирми, които са най-качествени и с най-широк функционален обхват, са над 2000 долара. Поради високата цена мнозинството малки и средни софтуерни фирми не инвестират в CASE инструменти. Изход от това затруднение засега е ползването на CASE инструменти с отворен код, които са безплатни.

Някои от основните предимства от използването на CASE инструментите са:

– ***Увеличават продуктивността на програмистите.*** CASE инструментите осигуряват автоматизация на програмирането и намаляват времето за завършване на много задачи. Програмистът се съсредоточава повече върху логиката на програмата и по-малко върху кодирането на тази логика. Това е валидно само след усвояване на работата с тези инструменти на професионално ниво (виж раздела за недостатъците).

– ***Повишават прецизността на програмния код.*** Използването на CASE инструменти води до ранното отстраняване на дефекти. Значението на ранното отст-

раняване на грешките е много важно. По-малко усилия и време се изразходват, ако корекциите се направят на по-ранен етап, какъвто е етапа на проектиране.
– ***Намаляване на времето за поддръжка***. С използване на CASE инструменти се създава много по-добра документация на програмния код. Това, от своя страна, води до по-бързо и по-ефективно извършване на определени изменения във вече функционираща система.

## 5. ЗАКЛЮЧЕНИЕ

В студията е изяснена ролята на *потребителската ефективност* при съвременните информационни системи, предназначени за пряко ползване от крайните потребители в Интернет. Значимостта на проблема за потребителската ефективност нараства изключително в резултат на бурното развитие на онлайн информационните системи, насочени към стотици хиляди крайни потребители (неспециалисти по информационни технологии) в областите на "трите е-та":
– системите за електронната търговия (e-commerce),
– образователните системи (e-learning),
– системите за административно обслужване на гражданите (e-government)

В студията са представени принципите на потребителски ориентираното проектиране като специфичен подход за постигане висока потребителска ефективност. Анализирани са фактори, свойства и характеристики на потребителския интерфейс, които имат съществено влияние за повишаване на потребителската ефективност.

От осъщественото в раздел 3 разглеждане могат да се обобщят следните методологични правила, приложими при изграждане на потребителски ориентирани онлайн информационни системи:
– Онлайн информационната система трябва да има ясна и логична схема за навигация, еднаква на всички нива и разбираема за потребителя.
– Системата трябва да притежава архитектурна и визуална яснота и единство.
– Функционалността на онлайн информационната система трябва да съответства възможно най-точно на очакванията на крайните потребители и на конкретната предметна област. Препоръчително е всяка функция да има едно и също име в целия сайт.
– Информационната система трябва да осигурява на потребителя чувство за контрол над всяко действие. Ясно да са означени входната точка за начало на работа, начинът за прекъсване на операция, начинът на изход от системата и възврат към предишна страница.
– Езикът използван в информационната система трябва да е точен, съобразен с предметната област, но без жаргони. Използваните термини навсякъде трябва да са с едно и също значение.
– Нужно е информационната система да има средства за онлайн помощ и подходящо ръководство за потребителя. Те трябва да бъдат ненатрапчиви, но да се намират лесно при необходимост.
– Да са осигурени възможности за връзка с администратора на системата, оторизираните лица за поддръжката й или собственика на фирмата. Крайните потре-

бители трябва да могат да изпратят запитване по електронната поща и да получат обратен отговор навреме.

– Необходимо е онлайн информационната система да е така проектирана, че информацията и действията в отделните секции да засягат един и същ обект. Информацията за обекта трябва да е достатъчно изчерпателна и свързана, да отговаря на очакванията на потребителя за конкретната предметна област.
– Съобщенията за грешки трябва да са разбираеми и да се появяват само в краен случай. В онлайн информационната система е нужно да са предвидени и да има заложена предварителна реакция на голяма част от потребителските грешки. Те трябва да се отстраняват без да е необходима намеса на самия потребител.

В раздел 4 подробно бяха разгледани възможностите за използване на унифицирания графичен език UML. Всеки от видовете UML диаграми акцентират на определен аспект от поведението на потребителите.

Диаграмите на случаите на употреба представят общата структура на поведението; диаграмите на взаимодействията представят колективна дейност на множество участници; а диаграмите на последователността представят взаимодействието между потребителите и системата във времеви аспект. Когато различните диаграми са добре обвързани една с друга, а това се постига относително лесно със съвременните системи за UML моделиране, те заедно изграждат цялостен модел на потребителя.

При потребителски ориентирания подход именно *моделът на потребителя* трябва да се постави в основата на разработването на една информационна система. Правилното построяване на този модел в голяма степен гарантира, че функционалността на изградената системата ще съответства на нуждите, познанията и практическите умения на потребителите. Това, от своя страна, води до постигане на двете главни цели на подхода – ефективно използване на системата и удовлетвореност на потребителите.

## ЛИТЕРАТУРА

1. Велев, Димитър, и др. (2007) SOA – основно направление в развитието на софтуерните технологии. В. „Бизнес информатика”- сборник доклади от юбилейна международна научна конференция по повод 40-та годишнина на специалност „Информатика”, УНСС, София.
2. Денчев, Емил и др. (2007). Проблеми и решения при избор на ERP система. В. „Бизнес информатика”- сборник доклади от юбилейна международна научна конференция по повод 40-та годишнина на специалност „Информатика”, С., УНСС, София.
3. Кисимов, Валентин, Р. Николов. (2007) Бизнес интелигентни инфраструктури – основа за бизнес интелигентни системи в реално време. В: „Бизнес информатика”- сборник доклади от юбилейна международна научна конференция по повод 40-та годишнина на специалност „Информатика”, УНСС, София.

4. Лазарова, Ваня и др. (2007) Критерии за определяне на потребителската ефективност на информационните системи в Интернет. В: „Бизнес информатика”- сборник доклади от юбилейна международна научна конференция, УНСС, София.
5. Лазарова, Ваня, Д. Петров. (2007) Определяне на икономическата ефективност при информационните системи и технологии. В: „Бизнес информатика”- сборник доклади от юбилейна международна научна конференция по повод 40-та годишнина на специалност „Информатика”, УНСС, София.
6. Мурджева, Александрина и др. (2007) Разделяне на бизнес логиката в многослойни приложения и съвременните технологии. В. „Бизнес информатика”- сборник доклади от юбилейна международна научна конференция по повод 40-та годишнина на специалност „Информатика”, УНСС, София.
7. Николова, Мария (2001). *Лаборатория по ползваемост отваря врати в НБУ* Computerworld - бр. 2, 2001 http://www.computerworld.bg/?call=USE~home;&page=paper&n=3953
8. Стефанова, Камелия и др. (2007). Проектиране на център за компетентност по бизнес интелигентност. В. „Бизнес информатика”- сборник доклади от юбилейна международна научна конференция по повод 40-та годишнина на специалност „Информатика”, УНСС, София.
9. Фаулър, Мартин (2007). UML основи. СофтПрес, София.
10. Цанева, Моника и др. (2007). Интегриране на бизнес приложения – основно предизвикателство към системното програмиране. В. „*Бизнес информатика*”- сборник доклади от юбилейна международна научна конференция по повод 40-та годишнина на специалност „Информатика”, УНСС, София.
11. Braun, David, Jeff Sivils, Alex Shapiro and Jerry Versteegh (2001) Unified Modeling Language (UML) Tutorial http://atlas.kennesaw.edu/~dbraun/csis4650/A&D/UML_tutorial/index.htm
12. Constantine, L. L**.(**2004). Beyond User-Centered Design and User Experience: Designing for User Performance. Constantine & Lockwood, Ltd.
13. Dewalt, Craig (1999) Business process modeling with UML. http://www.omg.org/docs/ad/00-02-04.pdf
14. Jeckle, Mario (online). Unified Modeling Language (UML) Tools**.** http://www.jeckle.de/umltools.html
15. Miller, Randy (online) Practical UML: A Hands-On Introduction for Developers http://dn.codegear.com/article/31863
16. Noran, Ovidius (2003) Business modeling: UML vs IDEF http://www.cit.gu.edu.au/~noran/Docs/UMLvsIDEF.pdf
17. Norman, Donald and Stephen Draper (ed.) (1986). *User-Centered System Design: New Perspectives on Human-Computer Interaction.* Lawrence Erlbaum Inc, Ney Jersey, USA.

**Цитирани институционални и фирмени сайтове**


1. Министерски съвет (2002) "Стратегия за електронно правителство" www.ccit.government.bg/e-govstrategia.doc
2. Facebook. http://www.facebook.com
3. michigan.gov (2003) Usability *Guidelines for e-Government Applications.* Department of Information Technology, State of Michigan, USA http://www.michigan.gov/documents/Usability_guidelines_2003v1_72381_7.pdf
4. MIT IS&T (2007) *Usability Guidelines.* Information Services and Technologies, MIT http://web.mit.edu/ist/usability/usability-guidelines.html
5. Serlio Software Development Corp. (2007). *CaseComplete diagrams.* http://www.casecomplete.com/Diagrams.aspx
6. UML Resources Center (online). http://umlcenter.visual-paradigm.com/
7. UsabilityNet (online). *International standards for HCI and usability.* http://www.hostserver150.com/usabilit/tools/r_international.htm#9241-11


**Стандарти на ISO за ползваемостта и потребителски ориентираното проектиране**


1. ISO 9241-11 (1998). Ergonomic requirements for office work with visual display terminals - Part 11: *Guidance on usability* http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16883
2. ISO 13407 (1999). Human-centred design processes for interactive systems http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=21197
3. ISO 9126-1 (2001). Software Engineering - Product quality - Part 1: *Quality model* http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=22749
4. ISO 19501 (2005) Information technology - Open Distributed Processing - *Unified Modeling Language*. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=32620
5. ISO 9241-110 (2006). Ergonomics of human-system interaction - Part 110: *Dialogue principles* http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=38009

## ПОТРЕБИТЕЛСКА ЕФЕКТИВНОСТ НА ИНФОРМАЦИОННИТЕ СИСТЕМИ В ИНТЕРНЕТ: МЕТОДОЛОГИЧНИ ПРОБЛЕМИ ПРИ ПОТРЕБИТЕЛСКИ-ОРИЕНТИРАНОТО ПРОЕКТИРАНЕ

Резюме

Значимостта на проблема за потребителската ефективност на уеб-базираните информационните системи нараства днес, в резултат на бурното развитие на системите за електронната търговия, електронно обучение и особено на системите за онлайн административно обслужване, разработвани по националната програмата "електронно правителство".

В студията са представени принципите на потребителски ориентираното проектиране като специфичен подход за постигане на висока потребителска ефективност. Анализирани са фактори, свойства и характеристики на потребителския интерфейс, имащи силно влияние върху ползваемостта на информационните системи. Анализират се възможностите за използване на езика за визуално обектно моделиране UML (Unified Modeling Language) при потребителски ориентираното проектиране.

## USER EFFECTIVENESS OF INFORMATION SYSTEMS IN INTERNET: METHODOLOGICAL PROBLEMS OF USER-CENTERED DESIGN

Abstract:

The significance of the problem of usability of web-based information systems grows up nowadays, as a result of the rapid development of the systems for e-commerce, e-learning, and especially the online systems for administrative services, developed on the national program "e-government".

The principles of user-centered design are presented in this paper. Some factors, properties and characteristics of the user interface, having strong influence on the usability of the information systems, are analyzed. The abilities of use of the language for visual object modeling UML (Unified Modeling Language) by the user-centered design are analyzed.