София 1849, кв. Кремиковци„ тел./факс: +359 2 994 70 18, +359 882 82 66 83, +359 876 24 82 04
www.chrdri.net

ВЪВЕДЕНИЕ В СОФТУЕРНОТО ПРОИЗВОДСТВО

1. Софтуерния проблем и софтуерната криза

Какво се разбира под софтуер? Това не е просто множество от компютърни програми. Съществува разлика между програма и програмен продукт.

Програмата обикновено е завършен продукт и се използува единствено от нейния автор. За нея почти липсва документация или други средства, подпомагащи нейното използуване. Понеже авторът е и неин потребител, наличието на програмни грешки не е от голямо значение. Ако програмата не работи, то нейният автор я поправя и продължава да я използува отново. Подобни програми са проектирани без да се отчита дали са преносими, надеждни или лесно използваеми.

Програмният продукт, освен от разработчиците на системата се използува от голям кръг хора. Обикновено потребителите са с различни компютърни умения и за тях са предвидени различни интерфейси. Съществува и достатъчна по обем документация относно работата на продукта. Програмните продукти се тестват внимателно преди тяхната експлоатация, защото потребителите нямат право да оправят техните грешки. При тях преносимостта е основна характеристика.

Извод: програма и продукт за решаването на един и същ проблем са две коренно различни неща. Брок е оценил, че за разработването на програмен продукт е нужно 10 пъти повече време, отколкото съответната програма. Софтуерната индустрия се занимава именно с разработването на програмни продукти, които понякога се наричат софтуер с индустриално качество. IEEE дефинира понятието софтуер като множество от компютърни програми, процедури и правила и свързани с тях документация и данни, отнасящи се до функционирането на компютърната система. Ясно се подчертава, че софтуерът не е само програма, но също така включва данни и документация.

Ще считаме термините “програмен продукт” и “софтуер” за еквивалентни по смисъл.

Основни характеристики на софтуера:

1. Софтуерът е скъп

С развитието на технологиите цената на хардуера пада, докато цената на софтуера расте, защото производството му изисква интензивна работна сила, която се заплаща скъпо. Изчислено е, че цената на хардуера, върху който работи дадено приложение е малка част от цената на самото приложение. За разработването на софтуер се влагат значителни ресурси – от време и хора

2. Софтуерът не е надежден, т.е. е с ниско качество

Обикновено софтуерът не се доставя навреме и надхвърля предвидения за разработването му бюджет. Една част от софтуерните проекти стават неуправляеми относно бюджет и план. Освен това софтуерът често не изпълнява онези функции, за които е разработен или извършва други функции, т.е. той е ненадежден и с лошо качество – например ранните провали в полетите на Аполо се дължат на грешки в софтуера. Цялата софтуерна индустрия е известна с това, че произвежда софтуерни системи с лошо качество. Грешките в софтуера се дължат на пропуски в процеса на проектирането и разработването му (за разлика от други системи).

3. Софтуерът се преработва често.

След като софтуерът се достави и инсталира, той навлиза във фаза на поддържане или

съпровождане. За разлика от други системи, софтуерът не се поддържа, защото се износва (модулите не могат да се износят и да се налага тяхната подмяна), а поради наличието на грешки в системата, които трябва да се отстранят при откриването им – т.нар. корективно поддържане. Освен това софтуерът често се подлага на промени за да се добавят нови свойства и услуги, или поради промяна на средата, в която работи. Подобно поддържане се нарича адаптивно поддържане. Независимо, че поддържането не е част от процеса на разработване софтуер, то е особено важна дейност, докато продуктът съществува. Доказано е, че неговата цена надхвърля цената за разработването на софтуера. Според някои автори съотношението е 80:20, 70:30, 60:40. трябва да се отчита, че поддържането не е свързано само с разчитането на кода и документацията, а трябва да се следи какъв ще бъде ефектът от промените, за да не се получат нежелани странични ефекти.

4. Софтуерът се разработва, а не се произвежда в класическия смисъл

Дейностите по разработването на хардуер и софтуер са коренно различни. Възникнали грешки при разработването на софтуер се отстраняват по-лесно. Главните разходи при софтуера са свързани с разработването му. Освен това софтуерът е на много високо ниво на абстрактност.

5. Софтуерът не се износва

За разлика от хардуера софтуерът не се износва и не изпитва влияния на околната среда: дъжд, слънце и т.н. Грешките могат да се отстранят, но няма резервни части, т.е. поддържането му е по-трудно.

6. Софтуерът не се асемблира от съществуващи части, а се разработва за нуждите на клиента.

Очевидно е, че настоящото състояние на софтуера е такова, че има още какво да се желае – софтуерна криза. Причината за това състояние е, че повечето подходи наблягат върху програмирането. Те са подходящи за малки проекти, но са софтуер с индустриално качество не са приложими. За да се преодолее софтуерната криза е необходим методичен подход при разработването на софтуер.

Дефиниция на софтуерно производство (софтуерен инженеринг): систематичен подход за разработването, използуването, поддържането и извеждането от експлоатация на софтуера. Осигурява методологии за разработването на софтуер, които да се прилагат от различни проекти.

Цел на софтуерното производство: да осигури методологии за разработването на софтуер, които да са приложими независимо от размера на разработваната система и да гарантират, че продукта ще е с високо качество, ниска цена и къс производствен цикъл.

2. Етапи на процеса за разработване на софтуер

Процесът на разработване на софтуер се състои от различни етапи (фази), като всеки от тях приключва с добре определен изход. Фазата е отрязък от време, през което се извършват определени дейности.

Редът на етапите зависи от модела на процеса (друг термин – модел на жизнения цикъл), който се прилага.

Дейностите са:

1. Спецификация на изискванията – фаза дефиниция, какво ще се прави
2. Разработване – включва следните фази: проектиране, кодиране, тестване, как ще се прави
3. Поддържане на системата – фаза съпровождане.

Етапите са:

2.1. Анализ на изискванията

Извършва се за да се разбере проблемът, който трябва да се решава. Целта е да се

идентифицира КАКВО е нужно за разработването на проекта, а не КАК системата ще постигне това. Задачата се усложнява от факта, че при разработването на софтуер участвуват две страни: клиент и разработчик. Разработчикът трябва да разработи система, която да задоволява клиентските изисквания. Той обикновено не разбира проблемната област на клиента, както и клиентът не разбира как се разработват системи. получава се комуникационна празнина и трябва да се намери общ език. Етапът завършва с документ, който се нарича Спецификация на софтуерните изисквания. Разработва се от системен аналитик.

През този етап се извършват две основни дейности:

- Анализ на задачата – изследват се съществуващи системи, ако има такива, какви са входните данни, функциите и изходите. За целта се провеждат разговори с крайните потребители и клиентите, изучават се съществуващите ръководства и процедури
- Формулиране на изискванията – създава се документът Спецификация, който описва всички изисквания относно функциите и ефективността на системата, ограничения при проектирането, интерфейсите за потребителите.

2,2, Проектиране на софтуера

Целта на този етап е да се изготви план за решаването на задачата и да се отговори на въпроса "как да се задоволят потребителските изисквания". Тя е критична по отношение на качеството на разработваната система и оказва съществено влияние върху останалите дейности. Изход от нея е документът Проект.

В етапа се извършват следните две основни дейности:

- Системно проектиране – нарича се още проектиране на високо ниво, цел е идентифицирането на отделните компоненти на системата, връзките помежду им, тяхното взаимодействие. КАКВИ КОМПОНЕНТИ?
- Детайлно проектиране – задава се вътрешната логика на всеки компонент, уточняват се алгоритмите, структурите от данни, псевдокод, КАК ЩЕ СЕ РЕАЛИЗИРА?

2.3. Кодиране и тестване

Целта е да се преведе проекта на системата в код на избрания език за програмиране по най-добрия възможен начин. Кодирането оказва силно влияние върху тестването и поддържането на системата. Затова програмите трябва да се пишат така, че да са лесни за четене и разбиране от друг човек.

Тестването е мярка за управление качеството на софтуера. Основната му функция е да се открият дефектите в софтуера, получени не само поради грешки в кодирането, но и поради грешки в предишните фази. Затова тестването се извършва на различни нива:

- Тестване на модул
- Тестване на интегрирането
- Тестване на системата
- Тестване пред клиента

Тестването отнема доста време и се нуждае от внимателно планиране.

3. Еволюция на софтуера:

I Етап 1950-1965 г.: Софтуерът се разработва специално за всеки тип приложение и е с ограничено разпространение като се използува единствено от автора си или определена организация. Грешките се поправят от определен човек, документиране почти липсва.

II Етап 1965-1975 г.:Мултипрограмирането и многопотребителските системи налагат нов начин на общуването "човек-машина". Обработката става интерактивна, появят се първите real

time системи. Създават се първите СУБД. Софтуерът започва да се разглежда като пазарен продукт, появяват се първите софтуерни къщи. Разширяват се библиотеките със софтуер, към които програмистите добавят собствен код. Възниква въпросът поддържането на софтуера. Понеже програмите са писани от различни хора, това ги прави практически невъзможни за подържане. Възниква т.нар. "софтуерна криза".

III Етап 1975-1985 г.: Компютърните системи се усложняват поради разпределената обработка и свързването им в мрежи. Микропроцесорите започват да се използуват навсякъде и компютрите стават достъпни за всички, защото цената им намалява.

IV Етап 1985-досега: възниква средата "клиент-сървър", Интернет. Софтуерът се разработва индустриално – Microsoft. Възникват обектно- ориентираните технологии, паралелните машини, експертните системи

3.1. Въпреки всичко следните проблеми свързани със софтуера остават:

- Не се използува напълно потенциалът на компютрите;
- Софтуерът не смогва да отговори на растящите нужди на пазара
- Широкото разпространение на компютрите поставя въпроса за надеждност на софтуера, защото при грешки загубите са огромни;
- Софтуерът трябва да е качествен;
- Съществуващите програми се поддържат трудно поради лоши проектантски решения.

Поради тези проблеми се създава методология за разработването на софтуер.

3.2. Митове, свързани със софтуера, в които вярват мениджърите:

1. Има разработени стандарти и процедури за разработване на софтуер. Да, ама използуват ли се от разработчиците, пълни ли са, отговорът обикновено е НЕ!
2. Използуват се с най- съвременните средства за разработване на софтуер, на най – нови компютри. Новите конфигурации отнемат повече време при разработването на софтуер, отколкото последния модел голяма ЕИМ. CASE средствата са много важни за постигане на добро качество и производителност, но колко разработчици ги използуват. Rational Rose - UML??
3. Ако закъснеем с графика ще наемем допълнително програмисти и ще успеем (понятието: монголска орда). Разработването на софтуер не е механичен процес като производството на други стоки. Привличането на нови хора налага обучение, а дали има време за това!!

3.3. Митове, свързани със потребителите:

1. За разработването на продукта са необходими само най-общи указания за целите и програмирането може да започне
2. Изискванията към софтуера непрекъснато се менят, но промените се отразяват лесно, защото софтуерът е гъвкав.

3.4. Митове, свързани със разработчика:

1. Да напиша веднъж програмата и да я подкарам с това работата ми свършва!
2. Не ме вълнува въпроса за качеството стига само да подкарам програмата
3. Проектът е успешен щом програмата работи – а документацията??