Софтуерно инженерство и етика (L1-1) Резюме

Софт. инженерство: **Инженерна дисциплина, която обхваща всички етапи на производството на софтуер – от спецификация до поддръжката и еволюцията.** Основна цел: Надеждни, ефективни и икономични системи.

Атрибути на добър софтуер:

-Поддържане(Maintainability): Лесна адаптация към нови изисквания.

- Надеждност и сигурност (Dependability and security): Защита от грешки и злоупотреби; надеждност, сигурност и безопасност; Надежден софтуер не трябва да причинява физически или икономически щети в случай на повреда на системата.

- Ефективност (Efficiency): Оптимално използване на ресурси.

- Приемливост (Acceptability): Достъпност и съвместимост с други системи; разбираема, използваема за потребителя

-да предоставя **необходимата функционалност и производителност** на потребителя

Различия между софтуерно инженерство и компютърни науки:

Компютърните науки са насочени към теории и основи, докато софт. инженерство е фокусирано върху практическата разработка.

Типове софтуерни продукти:

-Генерични /Generic/: Продават се на различни клиенти (напр. графични програми); собственост на софт. разработчик

-Персонализирани /Customized/: Разработени за конкрет. клиент - собственост на клиента

Софтуер - Компютърни програми, данни и свързаната с тях документация.

Основни софтуерни инженерни дейности- софт. спецификация, разработка на софтуер, софт. валидиране и еволюция на софтуера

Разликата между софт.инженерство и компютърни науки: Компютърните науки се фокусират върху теорията и основите;

софтуерното инженерство се занимава с практическите неща за разработване и доставяне на полезен софтуер;

Разликата между софт. инженерство и системно инженерство:

Системното инженерство се занимава с всички аспекти на разработка на компютърно базирани системи, вкл. хардуерно, софтуерно и процесно инженерство. Софт. инженерството е част от този по-общ процес.

Основните предизвикателства пред софт. инженерство:

справяне с нарастващото разнообразие, изискванията за намалени

срокове за доставка и разработване на надежден софтуер

Разходите за софт.инженерство: Приблизително 60% от разходите за софтуер са разходи за разработка, 40% са разходите за тестове. За персонализиран софтуер, разходите по еволюцията често надвишават разходите за разработка.

Най-добри софт. инженери техники и методи: различните техники са подходящ за различни видове системи.

Мрежата/Интернет доведе до наличието на софт. услуги и възможността за разработване на силно разпределени системи, базирани на услуги-напредък в езиците за програмиране и reuse

Дейности на софтуерния процес: Софтуерна спецификация; Разработка на софтуер; Проверка на софтуера; Развитие на софтуера

Общи **проблеми**, които засягат повечето софтуери:

-Хетерогенност - системите да работят като разпределени системи в мрежи, които включват различни видове компютри и мобилни устройства; **Бизнес и социална промяна; Сигурност и доверие**

Видове приложения:

-Самостоятелни /Stand-alone/ приложения- на локален компютър

-**Интерактивни приложения/Interactive transaction-based/** - базирани на транзакции- на отдалечен компютър и са достъпни от потребителите от техните собствени компютри или терминали- приложения за електронна търговия

-**Вградени системи за управление /Embedded control systems/ -** софтуерни системи за управление, които контролират и управляват хардуерни устройства

-**Системи за пакетна обработка /Batch processing systems/** - бизнес системи, които са проектирани да обработват данни в големи партиди. Те обработват голям брой индивидуални входове за създаване на съответните изходи

-**Системи за забавление / Entertainment/** - предимно за лична употреба и за забавления

-**Системи за моделиране и симулация /modeling and simulation/-** разработени от учени и инженери да моделират физически процеси или ситуации, които включват много, отделни, взаимодействащи обекти

-**Системи за събиране на данни /Data collection systems/-** събират данни от своята среда, използвайки набор от сензори и изпраща тези данни до други системи за обработка

-**Системи на системите /Systems of systems/**- съставени от редица други соф. системи

Основи на софтуерното инженерство **- основни принципи**:

-Системите трябва да се разработват чрез контролиран и разбираем процес на разработване

-Надеждността и производителността са важни за всички видове

системи

-разбиране и управление на софтуерната спецификация и изискванията (какво трябва да прави софтуерът) са важни

-трябва да използвате повторно софтуер, който вече е бил разработен, вместо да се пише нов софтуер

Софтуерно инженерство и уеб:

- Мрежата вече е платформа за стартиране на приложения и организациите все повече развиват уеб базирани системи, а не локални системи

-Уеб услугите позволяват функционалността на приложението да бъде достъпни през мрежата.

-Облачните програми са подход за предоставяне на компютърни услуги, при които приложенията работят отдалечено в "облакът". Потребителите не купуват софтуер, а плащат според употребата

Web software engineering: Software reuse; Уеб базираните системи трябва да бъдат разработени и доставени /incrementally/постепенно;

Потребителските интерфейси са ограничени от възможностите на уеб браузъри

Етика в софтуерното инженерство

Въпроси на професионалната отговорност: Конфиденциалност; Компетентност; Права на интелектуална собственост; Злоупотреба с компютър

ACM (Association for Computing Machinery) /IEEE (Institute of Electrical and Electronics Engineers) Code of Ethics дефинира 8 основни принципа:

-действат в обществен интерес

- защитават интересите на своя клиент и работодател;

-качествен продукт

-етичен подход на управление

- почтеност и независимост в професионалната преценка

- насърчаване на колегиалност

- доживотно обучение

Примери за етични дилеми: некоректно тестване на критични системи, използване на умения за злонамерени цели

Обучение на следващото поколение Софт. Инженерство,Alistair Cockburn

SE- 1/craft (занаят) 2/ Cooperative Game 3/ Lean Manufacturing 4/ Knowledge Acquisition (Придобиване на знания)

Craft = Lifelong deepening skills in a medium

7 основни **crafts** в разработката на софтуер:

1 Решаване какво да се построи

2 Управление (хора и проекти)

3 моделиране

4 Проектиране на външния изглед

5 Мащабен дизайн (архитектура)

6 Фин дизайн (програмиране)

7 Валидиране на работата

Хората учат умения на 3 етапа:

Shu Learn

Ha Collect

Ri Invent

Кооперативната игра извежда проблемите на хората

SE е кооперативна игра на изобретения и комуникация... конкретизиране на идеи в икономически контекст ...

Кооперативната игра учи на работа в екип и стратегии в

ситуации, които никога не се повтарят

Разработката на софтуер е серия от игри. Конфликтни цели се конкурират за ресурси:1/ доставяне система 2/ Настройване за следващата игра

Няма проста формула за победа, само стратегии в конкретни ситуации!

Lean Manufacturing- може да научи дизайнерите на много

Design=manufacturing if inventory=unvalidated decisions

Lean процесите използват по-малки стъпки и партиди

Knowledge Acquisition

Дизайнът е игра на стратегическо обучение

Платете, за да научите рано