Развој софтвера 2

1. Добродошли у свет контрукције софтвера

1.1. Шта је то конструкција софтвера?

Развој рачунарског софтвера може бити доста компликован процес. У протеклих 25 година, истраживачи су идентификовали бројне активности у развоју софтвера. Ту спадају следеће активности:

- Дефиниција проблема
- Дефинисање захтева
- Планирање конструкције
- Софтверска архитектура, или дизајн високог нивоа
- Детаљни дизајн
- Кодирање и дебагирање
- Тестирање јединица (енг. unit testing)
- Тестирање интеграције (енг. integration testing)
- Интеграција
- Тестирање система (енг. system testing)
- Корективно одржавање

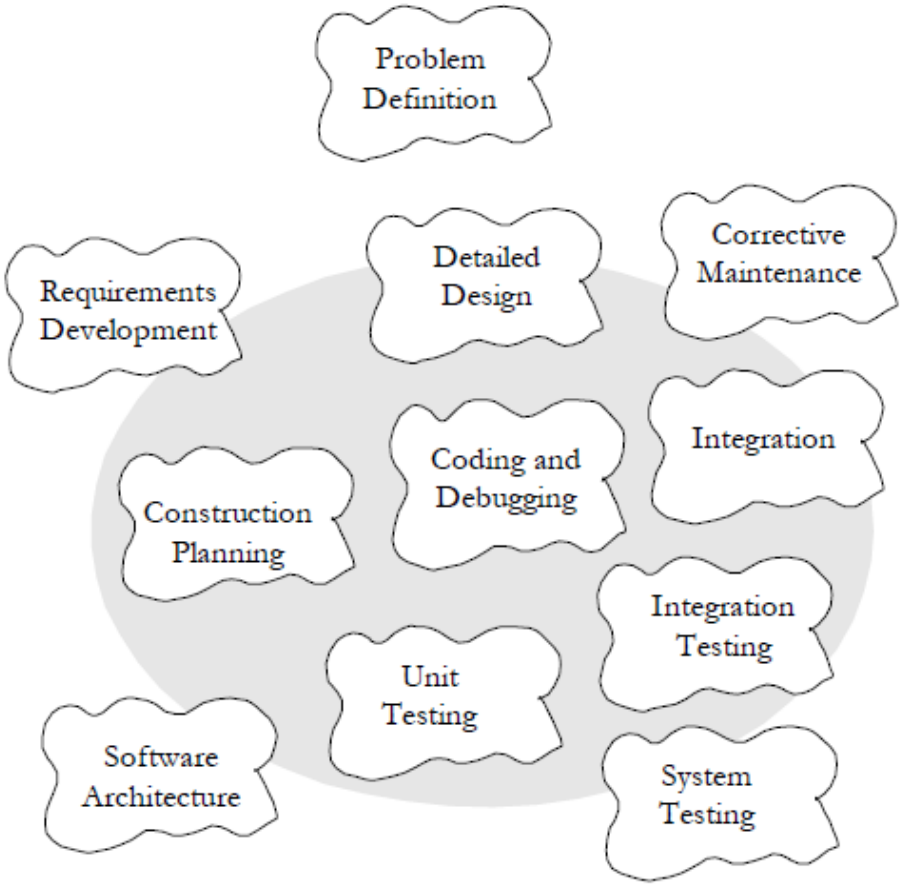
итд.

1.1. Шта је то конструкција софтвера?

Ако сте радили на неформалним пројектима, можда мислите да претходна листа садржи много бирократије. Ако сте радили на пројектима који су превише формални, тада ***знате*** да ова листа садржи много бирократије!
Касније ће бити детаљно разматрано колико је тешко постићи баланс између превише и премало формалности.

Ако сте учили да прогамирате и радили углавном на малим пројектима, могуће је да нисте правили разлике међу многобројним активностима које су укључене у креирање софтвера. Могуће је да сте ментално груписали све те активности и назвали их **програмирање**. Ако се ради на неформалним пројектима, онда је вероватно главна активност на коју се помисли када се говори о креирању софтвера програмирање (још се назива и **конструкција**).

1.1. Шта је то конструкција софтвера?



Слика 1-1
Активности конструкције су приказане унутар сиве елипсе. Конструкција се фокусира на кодирање и дебагирање, али укључује и нешто од детаљног дизајна, тестирања јединица, тестирања интеграције и од других активности.

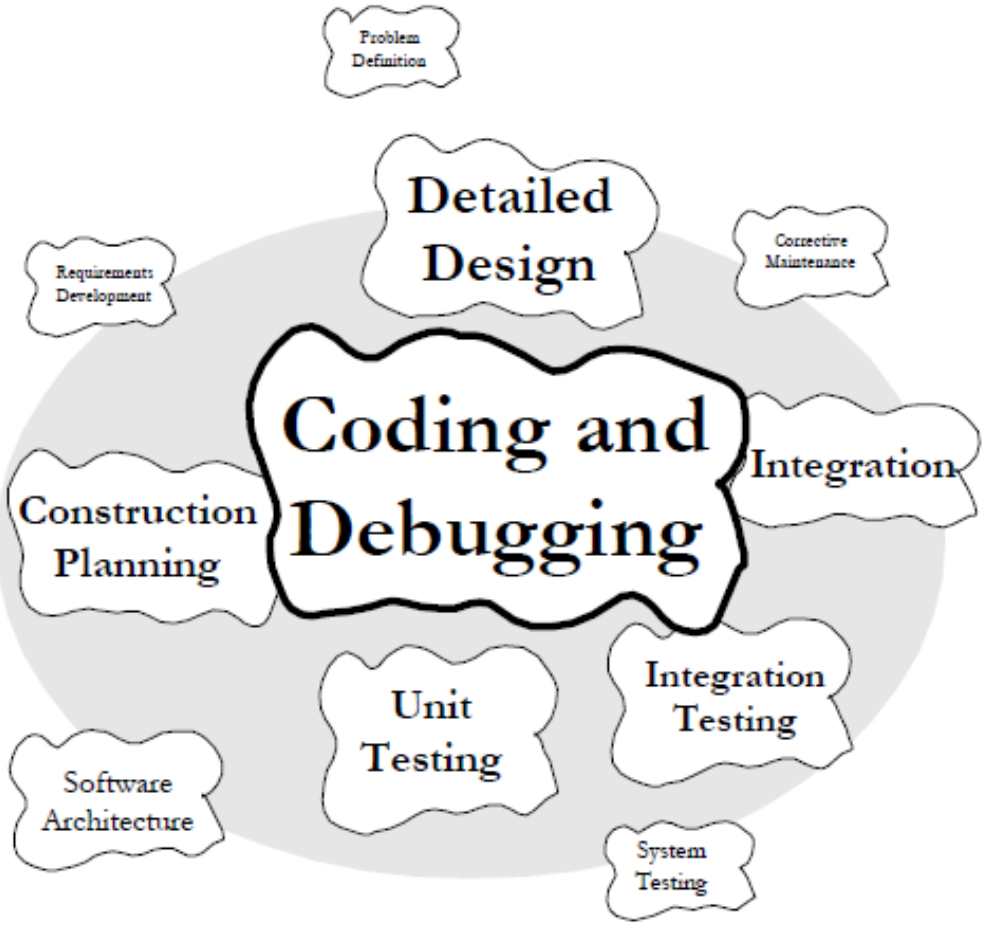
1.1. Шта је то конструкција софтвера?

Конструкција је доминантно кодирање и дибагирање, али оно укључује и елементе детаљног дизајна, тестирања јединица, интеграције, тестирања интеграције и других активности.

Ако се неки курс односи на све аспекте развоја софтвера, он треба да садржи фино балансирана разматрања свих активности садржаних у процесу развоја софтвера.

Конструкција се понекад још назива и кодирање или програмирање. Кодирање баш и није најбоља реч за опис те активности, зато што она имплицира да се ради о механичком превођењу дизајна који већ постоји на програмски језик који разуме рачунар. Међутим, конструкција није потпуно механичка активност и она укључује велику дозу креативности и просуђивања. У презентацијама које следе, термини програмирање и конструкција ће означавати исти појам, па ће се користити и један и други термин.

1.1. Шта је то конструкција софтвера?



Слика 1-2
У презентацијама које следе је однос између времена проведеног у разматрању детаљног дизајна, кодирања, дебагирања и тестирања јединица грубо подељен као што је приказано на горњој слици.

1.1. Шта је то конструкција софтвера?

Слика 1-1 и слика 1-2 представљају поглед на активности конструкције са високог нивоа. Међутим, поставља се питање: шта је са детаљима? Следе неки од конкретних задатака који су укључени у конструкцију:

- Проверавање да ли су извршени сви припремни радови, тако да се успешно може наставити са конструкцијом
- Одређивање начина на који ће бити тестиран програмски код
- Дизајн и писање класа и рутина
- Креирање и именовање променљивих и именованих константи
- Избор контролних структура организација блокова наредби
- Тестирање јединица, тестирање интеграције и дебагирање написаног кода
- Прегледање дизајна на ниском нивоу и програмског кода који су написали други, као и омогућавање да други прегледају ваш дизајн и код
- Пажљиво реформатирање и коментарисање кода
- Интеграција софтверских компоненти које су одвојено креиране
- Подешавање кода (енг. tuning) тако да буде мањи и бржи

итд.

1.1. Шта је то конструкција софтвера?

Када се види које су све активности контрукције, природно се поставља питње: које све активности ***нису*** у оквиру конструкције? То је добро питање. Важен активности које нису део конструције укључују управљање, дефинисање захтева, архитектуру софтвера, дизајн корисничког интерфејса, тестирање система и одржавање.

Свака од горе побројаних активности утиче на коначан успех пројекта онолико колико и конструкција. Претходно тврђење се односи на пројекте који захтевају више од једног-два човека и који трају дуже од неколико недеља.

Многи од најузбудљивијих пројеката данашњице користе софтвер у великој мери. Неки примери за то су Интернет, специјални ефекти у филмовима, системи за одржавање живота у медицини, свемирски програм, аеронаутика, брзе финанијске трансакције и научна истраживања. Ово пројекти, исто као и конвенционалнији пројекти, могу много напредовати захваљујући побољшаној пракси, јер су многе фундаменталне поставке потпуно исте.

1.2. Зашто је важна конструкција софтвера?

Зашто је важно да се концентришемо на конструкцију софтвера? Ево неколико разлога:

Конструкција представља велики део развоја софтвера
Зависно од величине пројекта, контрукција обично заузме од 30% до 80% укупног времена проведеног у развоју пројекта. Ако нешто захтева толико много времена у пројекту, тада је то нешто повезано са пројектом и утиче на његов успех

Конструкција је централна активност у развоју софтвера
Захтеви и архитектура се реализују пре конструкције, да би се на тај начин омогућила ефикасна конструкција. Тестирање система се реализује после конструкције и на тај начин се верификује коректност извшеног конструисања. Дакле, конструкција је у центру процеса развоја софтвера.

1.2. Зашто је важна конструкција софтвера?

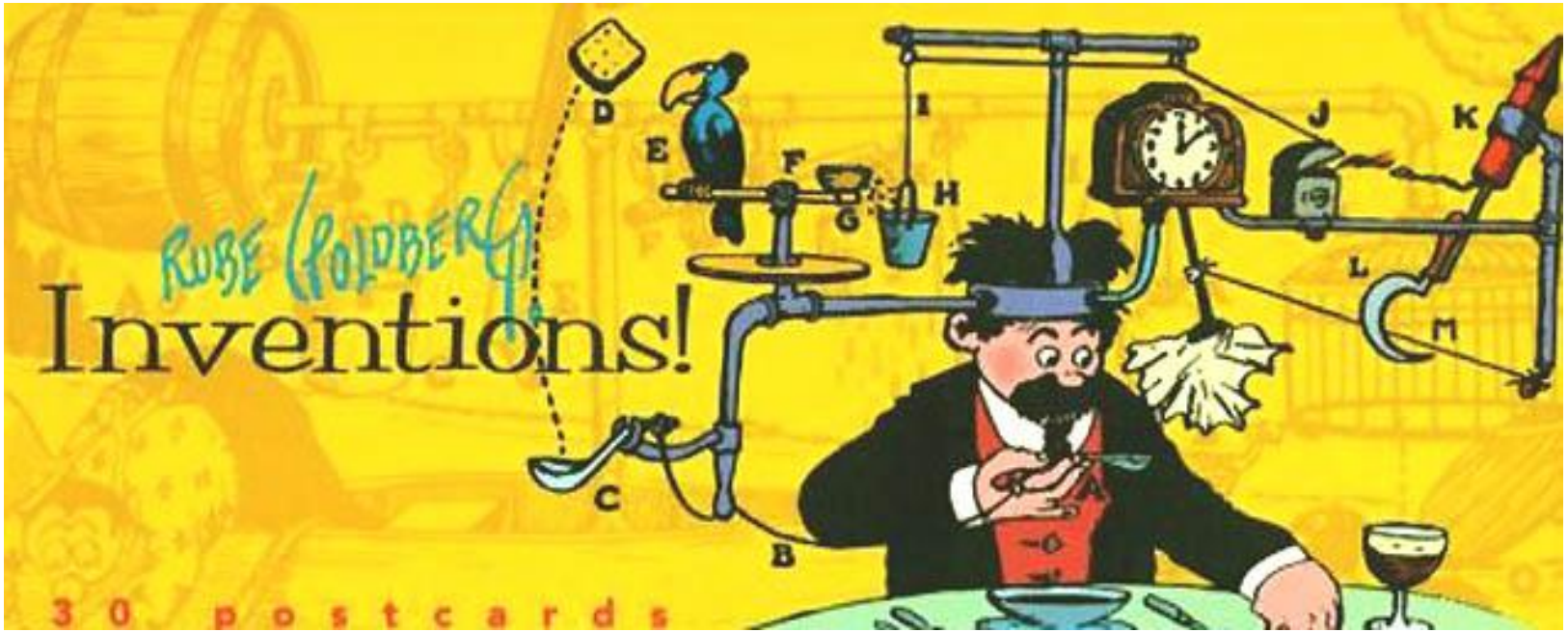
Ако се фокусира на конструкцију, тада продуктивност програмера може битно да се повећа

Класична студија аутора Сакман, Ериксон и Грант из 1968. године је показала да продуктивност програмера током конструкције варира од 10 до 20 **пута**. Од тог времена, новим испитивањима су више пута потврђени резулатти до којих су дошли ови аутори. Дакле, свим програмерима помаже уколико науче технике које већ користе најбољи међу њима.

Производ конструкције, тј. изворни код, често представља једини тачан опис софтвера

У великом броју пројеката, једина документација која је доступна програмеру је сам код. Спецификације захтева и документи о дизајну могу застарити, али изворни код је увек ажуран. Према томе, императив је да изворни код буде најквалитетнији могући. Конзистентна примена техника за побољшање изворног кода чини разлику између превише сложеног, нејасног и непрегледног програма (енг. Rube Goldberg machine) и детаљног, коректног и информативног програма.

1.2. Зашто је важна конструкција софтвера?



Слика 1-3
Илустрација сложеног, нејасног и непрегледног програма (енг. Rube Goldberg machine).

1.2. Зашто је важна конструкција софтвера?

Конструкција је једина активност за коју се гарантује да ће бити реализована

Идеални софтверски пројекат пролази кроз пажљиво разматрање захтева, архитектуре и дизајна пре него што почне конструкција. Идеално пројекат подразумева и исцрпно статистички контролисано тестирање система по завршетку конструкције.

Пројекти у реалном свету, међутим, нису савршени. Ту се често дешава да се прескоче захтеви и дизајн и да се одмах пређе на конструкцију. Такође се дешава да се избаци тестирање зато што још много грешака треба да се поправи а време је истекло. Ипак, колико год пројекат био пожуриван или лоше планиран, сигурно је да се не може избацити конструкција – то је место где “гума додирује пут”. Унапређење конструкције стога представља начин унапређења било које активности за развој софтвера, без обзира на то колико је тај развој скраћен.

1.3. Рекапитулација

● Конструкција софтвера је централна активност у развоју софтвера; конструкција је једина активност која ће се сигурно догодити у сваком пројекту.

● Главне активности у конструкцији су детаљни дизајн, кодирање, дегагирање и тестирање јединица.

● Уобичајенио термини за конструкцију су и “кодирање и дебагирање” и “програмирање”.

● Квалитет конструкције суштински утиче на квалитет софтвера.