Pós-Graduação em Computação Distribuída e Ubíqua

INF612 - Aspectos Avançados em Engenharia de Software
Gerência de Qualidade e Integração Contínua

,So(t) are %$&i$eeri$&* Sommer#i--e* . / %dição* Capítu-os 01 23 e 245
,Cotiuous I$te&ratio$* Du#a--* Capítu-os 61 2 e 05

Sa$dro S* ! $drade
sa$droa$drade+i(ba*edu*br

Pós-Graduação em Computação Distribuída e Ubíqua

INF612 - Aspectos Avançados em Engenharia de Software
Gerência de Qualidade - Fundamentos

,So(t) are %$&i$eeri$&* Sommer#i--e* . / %dição* Capítu-os 01 23 e 245
,Cotiuous I$te&ratio$* Du#a--* Capítu-os 7 8 95

Sa$dro S* ! $drade
sa$droa$drade+i(ba*edu*br

Ob:eti#os

- ! prese$tar as moti#aç ; es para estudo e uso pr<ti''o da Ger=$''ia de >ua-idade
- ! prese$tar as re-aç ; es e$tre Ger=$''ia de >ua-idade e Pro''esso de Dese$#o-#ime$to de So(t) are
- ! prese$tar ''omo uti-i?ar i$speç ; es1 re#is ; es1 mediç ; es e i$te&ração ''o$tí$ua dura$te o pro''esso de &er=$''ia de qua-idade

Ger=$"ia de >ua-idade

- Pri$"ipais (a"etas@
  - Or&a$i?a"io$a-@
    - %stabe-e"ime$to de um *(rame) orA* de pro"essos or&a$i?a"io$ais e padro$i?aç ; es que "o$du?em a *so(t) are* de a-ta qua-idade
  - De pro:eto@
    - %$#o-#e a ap-i"ação de pro"essos de qua-idade espe"í(i"os1 &aratido que os arte(atos produ?idos estão em "o$(ormidade "om as padro$i?aç ; es uti-i?adas $o pro:eto
    - %$#o-#e o estabe-e"ime$to de um p-a$o de qua-idade1 i$di"a$do as metas de qua-idade para o pro:eto e de(i$i$do quais pro"essos e padro$i?aç ; es serão uti-i?ados

Ger=$"ia de >ua-idade

Quality Assurance (QA): definição dos processos e padronizações que devem conduzir a produtos de alta qualidade e introdução de processos de qualidade na manufatura

Quality Control: aplicação dos processos de qualidade com o objetivo de remover aqueles produtos que não possuem o nível desejado de qualidade

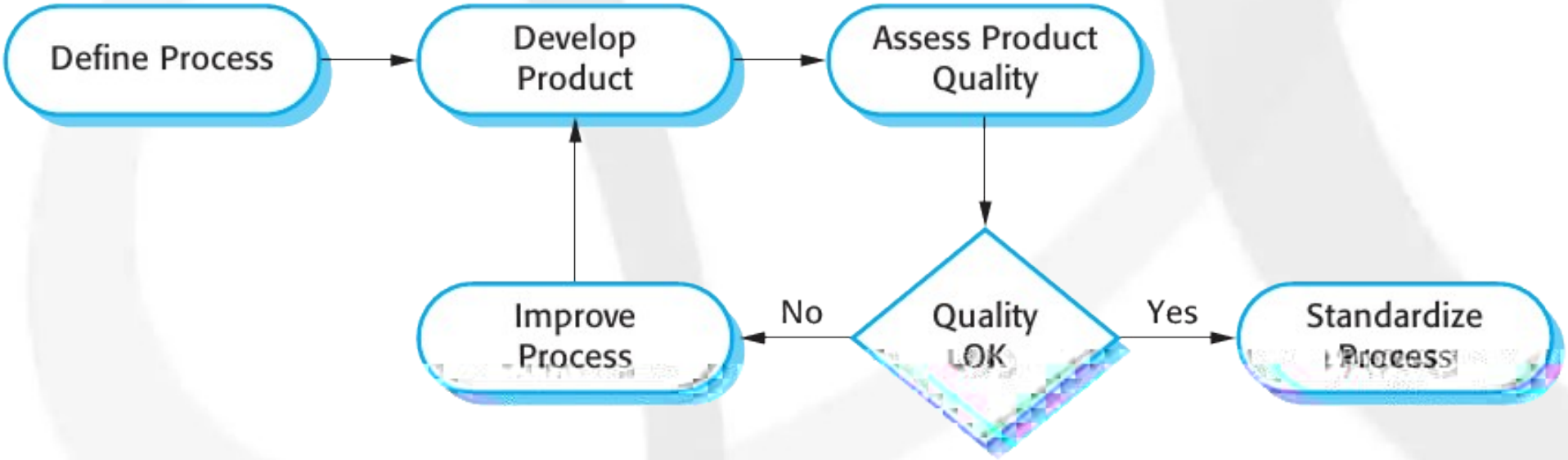
Ger=$"ia de >ua-idade

- ! tributos de >ua-idade@

Usability	Security	Testability
Reusability	Reliability	Adaptability
Efficiency	Resilience	Modularity
Learnability	Robustness	Complexity

Ger=$"ia de >ua-idade

- >ua-idade suportada pe-o pro"esso@



Ger=$"ia de >ua-idade

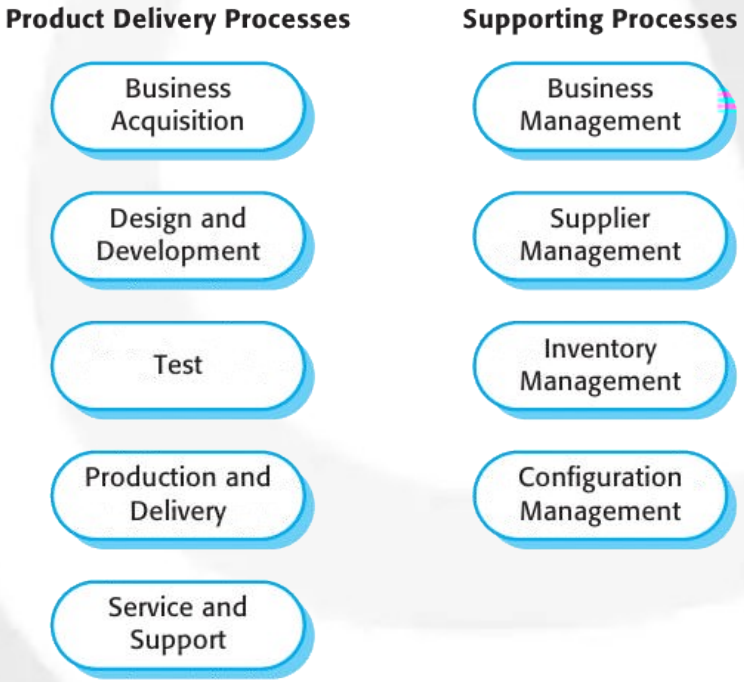
- ImportB$"ia de adoção de padro$i?aç ; es@
  - São baseadas em "o$ ' e"ime$tos sobre as me- ' ores ou mais apropriadas pr<ti"as da or&a$i?ação* Gera-me$te este "o$ ' e"ime$to C "o$struído atra#Cs de te$tati#a-e-erro* Do"ume$t<--o (a"i-ita o reuso e e#ita repetição de erros
  - Dispo$ibi-i?am um *(rame) orA* para de(i$ir o que Dqua-idadeE si&$i(i"a em uma situação parti"u-ar* %stabe-e um "ritCrio para de"idir se determi$ado $í#e- de qua-idade (oi a-"a$çado
  - Gara$te uma "o$ti$uidade qua$do o traba- ' o de uma pessoa C "otiuado por outra* Todos os e$&e$ ' eiros adotam a mesma pr<ti"a

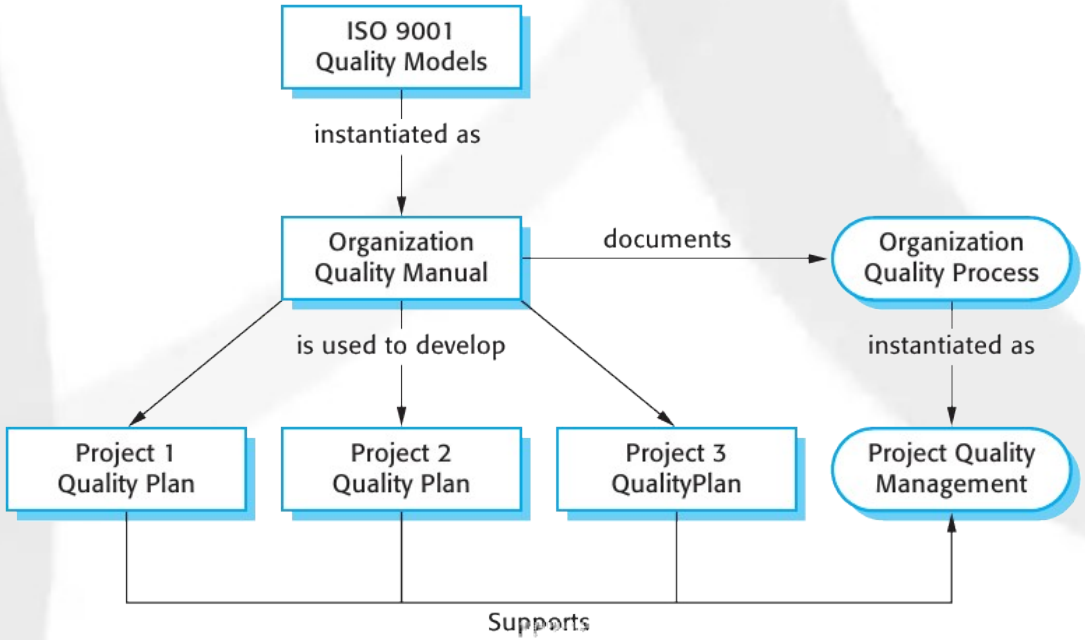
Ger=$"ia de >ua-idade

- Padro$i?aç ; es de pro"esso e de produto@

Ger=$"ia de >ua-idade

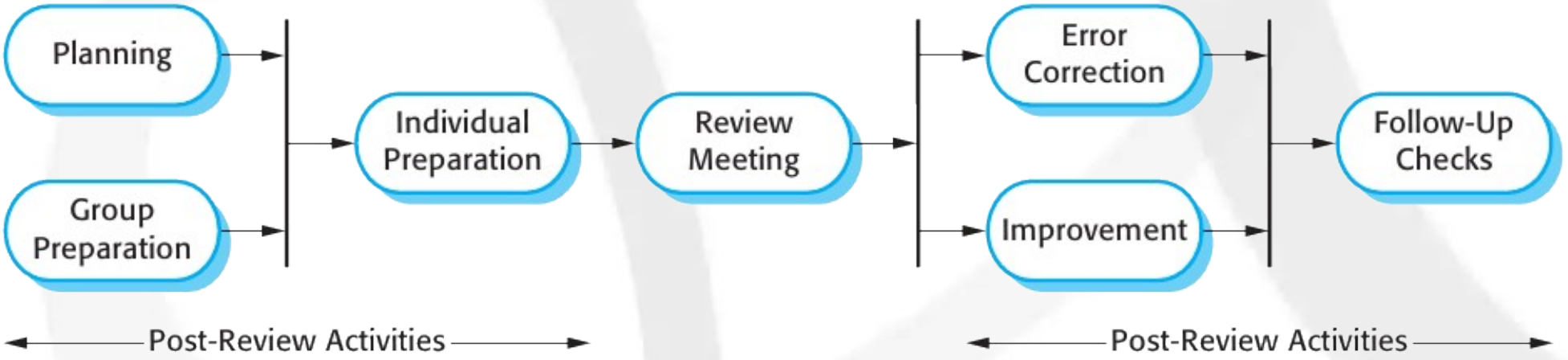
- Frame) orA de padro$i?aç ; es ISO . FF6
  - *Frame) orA* para dese$#o-#ime$to de padro$i?aç ; es de *so(t) are*





Ger=$"ia de >ua-idade

- Pro"esso de re#isão de so(t) are



Ger=$"ia de >ua-idade

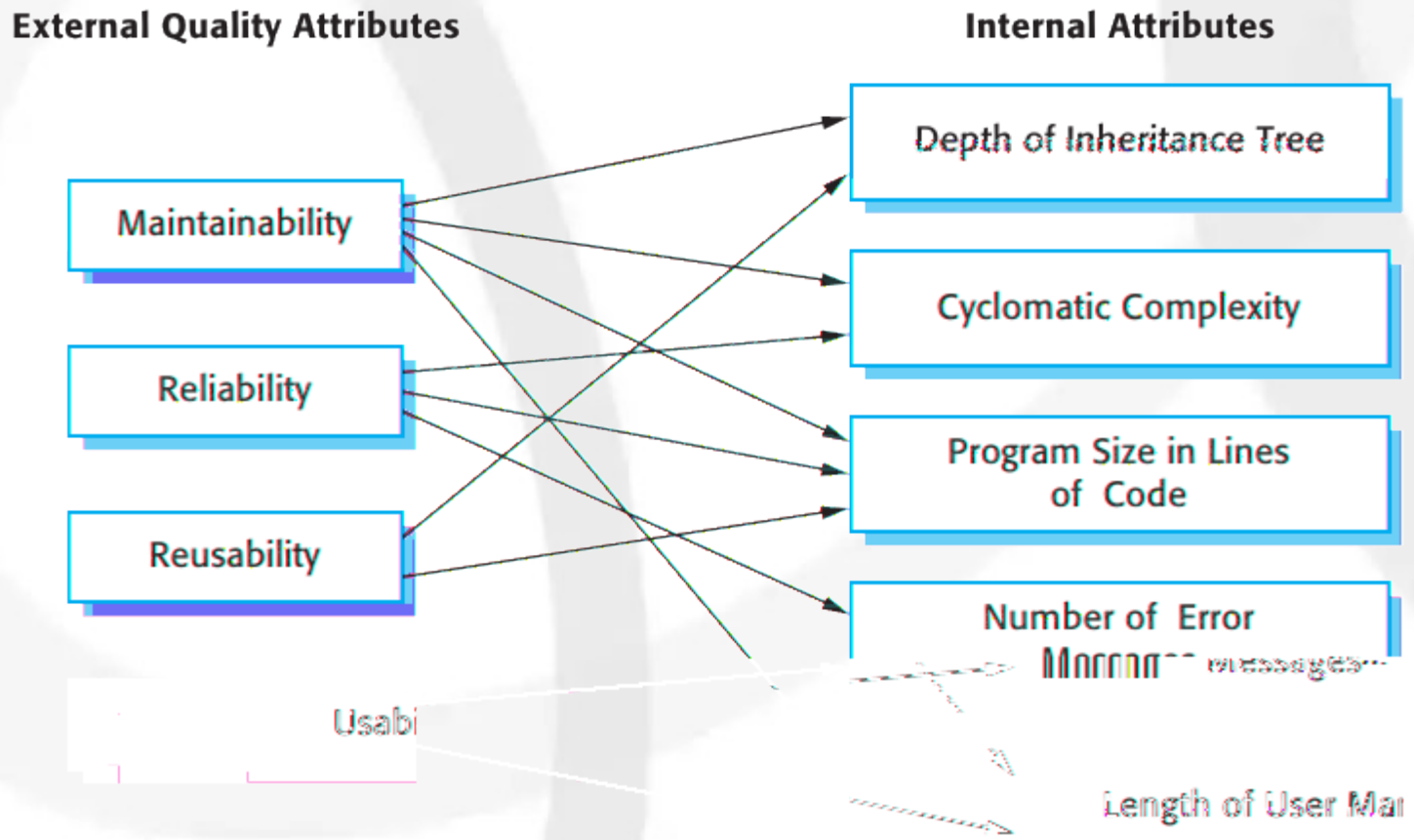
- I$speção G Re#isão

Inspeção (*Inspection*): *peer review* de qualquer trabalho produzido pelos indivíduos do grupo, objetivando a descoberta de erros através da aplicação de um processo bem definido (*Fagan Inspection*). Verifica se as padronizações e diretrizes estão sendo seguidas

Revisão (*Code Review*): tipo especial de inspeção onde uma equipe examina um código-fonte e corrige os defeitos nele encontrados. Um defeito pode ser: um requisito inapropriadamente implementado, uma funcionalidade que não funciona ou uma funcionalidade que pode ser melhorada. Importantes para realizar treinamento cruzado dos programadores e ajudar os menos experientes com boas práticas de desenvolvimento

Ger=$"ia de >ua-idade

- ! tributos de qua-idade iteros e eGter$os@



Ger=$"ia de >ua-idade

- !-&umas mCtri"as de produto@

Software metric	Description
Fan-in/Fan-out	Fan-in is a measure of the number of functions or methods that call [illegible]
ize of Length of code	This is a measure of the size of a program. Generally, the larger the s [illegible]
y of a program. This control derstandability. I discuss	Cyclomatic complexity	This is a measure of the control complexit complexity may be related to program un cyclomatic complexity in Chapter 8.

Ger=$"ia de >ua-idade

- !-&umas mCtri"as de produto@

Length of identifiers	This is a measure of the average length of identifiers (names for variables, classes, methods, etc.) in a program. The longer the identifiers, the more likely they are to be meaningful and hence the more understandable the program.
Depth of conditional nesting	This is a measure of the depth of nesting of if-statements in a program. Deeply nested if-statements are hard to understand and potentially error-prone.
Fog index	This is a measure of the average length of words and sentences in documents. The higher the value of a document's Fog index, the more difficult the document is to understand.

Ger=$"ia de >ua-idade

- Suite CH de mCtri"as OO

Object-oriented metric	Description	
Weighted methods per class (WMC)	This is the number of methods in each class, weighted by the complexity of each method. Therefore, a simple method may have a complexity of 1, and a large and complex method a much higher value. The larger the value for this metric, the more complex the object class. Complex objects are more likely to be difficult to understand. They may not be logically cohesive, so	
s in an inheritance tree.		cannot be reused effectively as superclasse
vels in the inheritance tree perations (methods) from tree, the more complex the be understood to understand .	Depth of inheritance tree (DIT)	This represents the number of discrete lev where subclasses inherit attributes and op superclasses. The deeper the inheritance t design. Many object classes may have to b the object classes at the leaves of the tree
diate subclasses in a class. It , whereas DIT measures its greater reuse. It may mean that base classes because of the m.	Number of children (NOC)	This is a measure of the number of imme measures the breadth of a class hierarchy, depth. A high value for NOC may indicate more effort should be made in validating number of subclasses that depend on the

Ger=$"ia de >ua-idade

- Suite CH de mCtri"as OO

Coupling between object classes (CBO)	Classes are coupled when methods in one class use methods or instance variables defined in a different class. CBO is a measure of how much coupling exists. A high value for CBO means that classes are highly dependent, and therefore it is more likely that changing one class will affect other classes in the program.
Response for a class (RFC)	RFC is a measure of the number of methods that could potentially be executed in response to a message received by an object of that class. Again, RFC is related to complexity. The higher the value for RFC, the more complex a class and hence the more likely it is that it will include errors.
Lack of cohesion in methods (LCOM)	LCOM is calculated by considering pairs of methods in a class. LCOM is the difference between the number of method pairs without shared attributes and the number of method pairs with shared attributes. The value of this metric has been widely debated and it exists in several variations. It is not clear if it really adds any additional, useful information [illegible]

Pós-Graduação em Computação Distribuída e Ubíqua

INF612 - Aspectos Avançados em Engenharia de Software
Gerência de Qualidade e Desenvolvimento Ágil

,So(t) are %$&i$eeri$&* Sommer#i--e* . / %dição* Capítu-os 01 23 e 245
,Cotiuous I$te&ratio$* Du#a--* Capítu-os 7 8 95

Sa$dro S* ! $drade
sa$droa$drade+i(ba*edu*br

Ger=$"ia de >ua-idade

- Cara"terísti"as "omu$s dos mCtodos <&eis de dese$#o-#ime$to@
  - Os pro"essos de espe"i(i"ação1 pro:eto e imp-eme$tação a"o$te"em de (orma e$tre-açada I*i$ter-ea#ed*J
    - Kão ' < uma espe"i(i"ação deta-' ada do sistema* Do"ume$tação de pro:eto C mi$ima ou &erada automati"ame$te
  - O sistema C dese$#o-#ido em uma sCrie de #ers ; es* Usu<rios (i$ais e outros *staAe 'o-ders* espe"i(i"am e a#a-iam "ada #ersão
  - GUIs são dese$#o-#idas de (orma r<pida e (a"i-me$te tra$s(ormadas para te"$o-o&ias de aprese$tação espe"í(i"as

Ger=$"ia de >ua-idade

- L a$i(esto M&i-@
  - D%stamos des"obri$do (ormas me-' ores de dese$#o-#er *so(t) are*1 prati"a$do e a:uda$do outros dese$#o-#edores* ! tra#Cs deste traba-' o passamos a dar #a-or a@
    - I$di#íduos em #e? de i$teraç ; es atra#Cs de pro"essos e (errame$tas
    - *So(t) are* que (u$"io$a em #e? de eGte$sas do"ume$taç ; es
    - Co-aboração direta do "-ie$te em #e? de $e&o"iação de "o$trato
    - Resposta r<pida a muda$ças em #e? de se&uir um p-a$o prC-de(i$ido
  - ! o mesmo tempo em que os items da direita são importa$tes1 $ós de"idimos #a-ori?ar muito mais os ite$s da esquerdaE

Ger=$"ia de >ua-idade

- %Gemp-os de LCtodos M&eis@
  - e7treme Pro&rammi$& I e"A16...-2FFFJ
  - S"rum ICo'$1S"') aber1 eed-e12FF6-2FF.J
  - CrNsta- ICo"Abur$12FF6-2FF3J
  - !dapti#e So(t) are De#e-opme$t IOi&'smit'12FFFJ
  - DSDL IStap-eto$16..P-2FF0J
  - Feature Dri#e$ De#e-opme$t IPa-mer1Fe-si$&12FF2J
  - Ista"iaç;es <&eis do RUP I*Ratio$a- U$i(ied Pro"ess*J

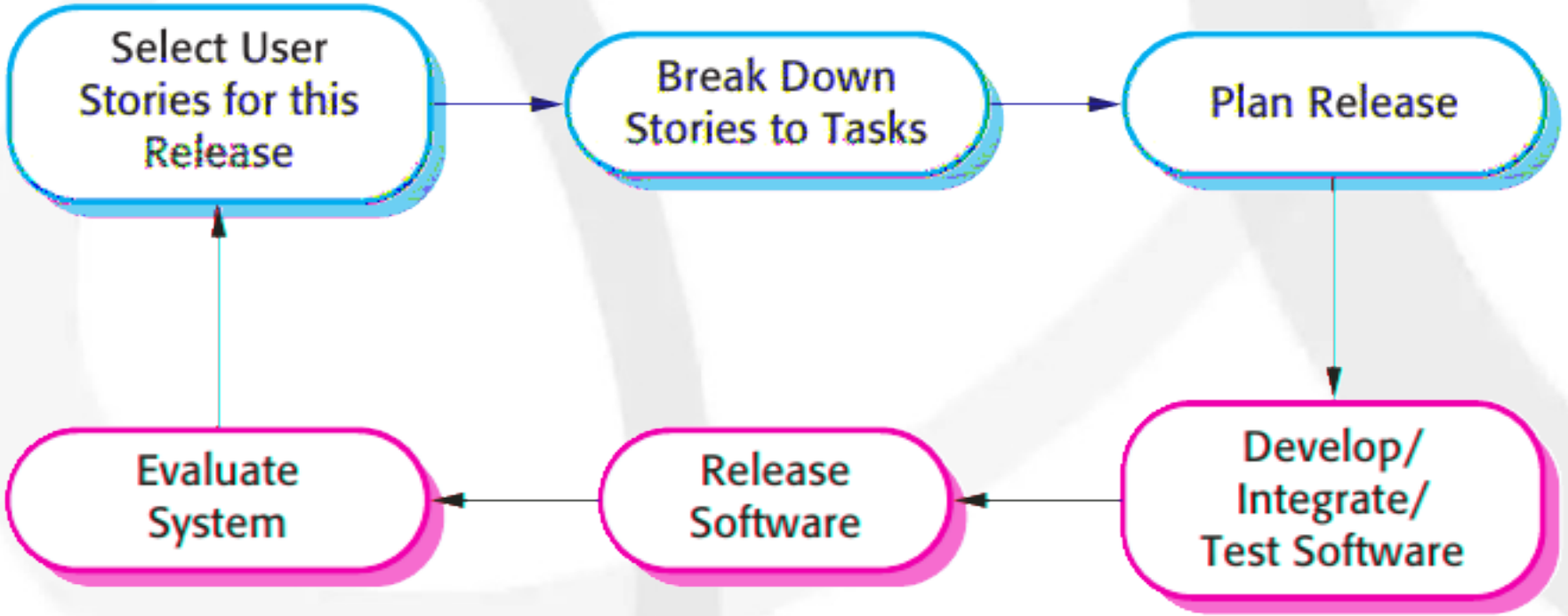
Ger=$"ia de >ua-idade

- Pri$"ípios dos mCtodos <&eis@

Principle	Description
Customer involvement	Customers should be closely involved throughout the development process. Their role is provide and prioritize new system requirements and to evaluate the iterations of the system.
Incremental delivery	The software is developed in increments with the customer specifying the requirements to be included in each increment.
People not process	The skills of the development team should be recognized and exploited. Team members should be left to develop their own ways of working without prescriptive processes.
ne system to	Embrace change	Expect the system requirements to change and so design th accommodate these changes.
nd in the minate complexity	Maintain simplicity	Focus on simplicity in both the software being developed a development process. Wherever possible, actively work to eli from the system.

Ger=$"ia de >ua-idade

- Ci"-o de *re-ease* do e7treme Pro&rammi$&@



Ger=$"ia de >ua-idade

- Pr<ti"as do e7treme Pro&rammi$&@

Principle or practice	Description
Incremental planning	Requirements are recorded on Story Cards and the Stories to be included in a release are determined by the time available and their relative priority. The developers break these Stories into development 'Tasks'. See Figures 3.5 and 3.6.
Small releases	The minimal useful set of functionality that provides business value is developed first. Releases of the system are frequent and incrementally add functionality to the first release.
Simple design	Enough design is carried out to meet the current requirements and no more.
Test-first development	An automated unit test framework is used to write tests for a new piece of functionality before that functionality itself is implemented.
Refactoring	All developers are expected to refactor the code continuously as soon as possible code improvements are found. This keeps the code simple and maintainable.

Ger=$"ia de >ua-idade

- Pr<ti"as do e7treme Pro&rammi$&@

		to always do a good job.
nds of ıe code.	Collective ownership	The pairs of developers work on all areas of the system, so that no isla expertise develop and all the developers take responsibility for all of th Anyone can change anything.
ole system.	Continuous integration	As soon as the work on a task is complete, it is integrated into the wh After any such integration, all the unit tests in the system must pass.
acceptable as the net effect is productivity	Sustainable pace	Large amounts of overtime are not considered often to reduce code quality and medium term
(the Customer) should be ı an extreme programming lopment team and is responsible or implementation.	On-site customer	A representative of the end-user of the system available full time for the use of the XP team. In process, the customer is a member of the deve for bringing system requirements to the team f

Ger=$"ia de >ua-idade

- Pro&ramação em pares – #a$ta&e$s@
  - ! poia a ideia de propriedade e respo$sabi-idade "o-eti#os para o sistema I*e&o-ess pro&rammi$&*J*
  - Fu$"io$a "omo um pro"esso i$(orma- de *re#ie)*Q me$os (orma- que uma i$speção porCm C mais barato
  - ! :uda a suportar *re(a"tori$&** %#ita que o traba- ' o de uma R$i"a pessoa Ique (a? o *re(a"tori$&*J se:a #isto "omo um produtor de resu-tados a -o$&o pra?o

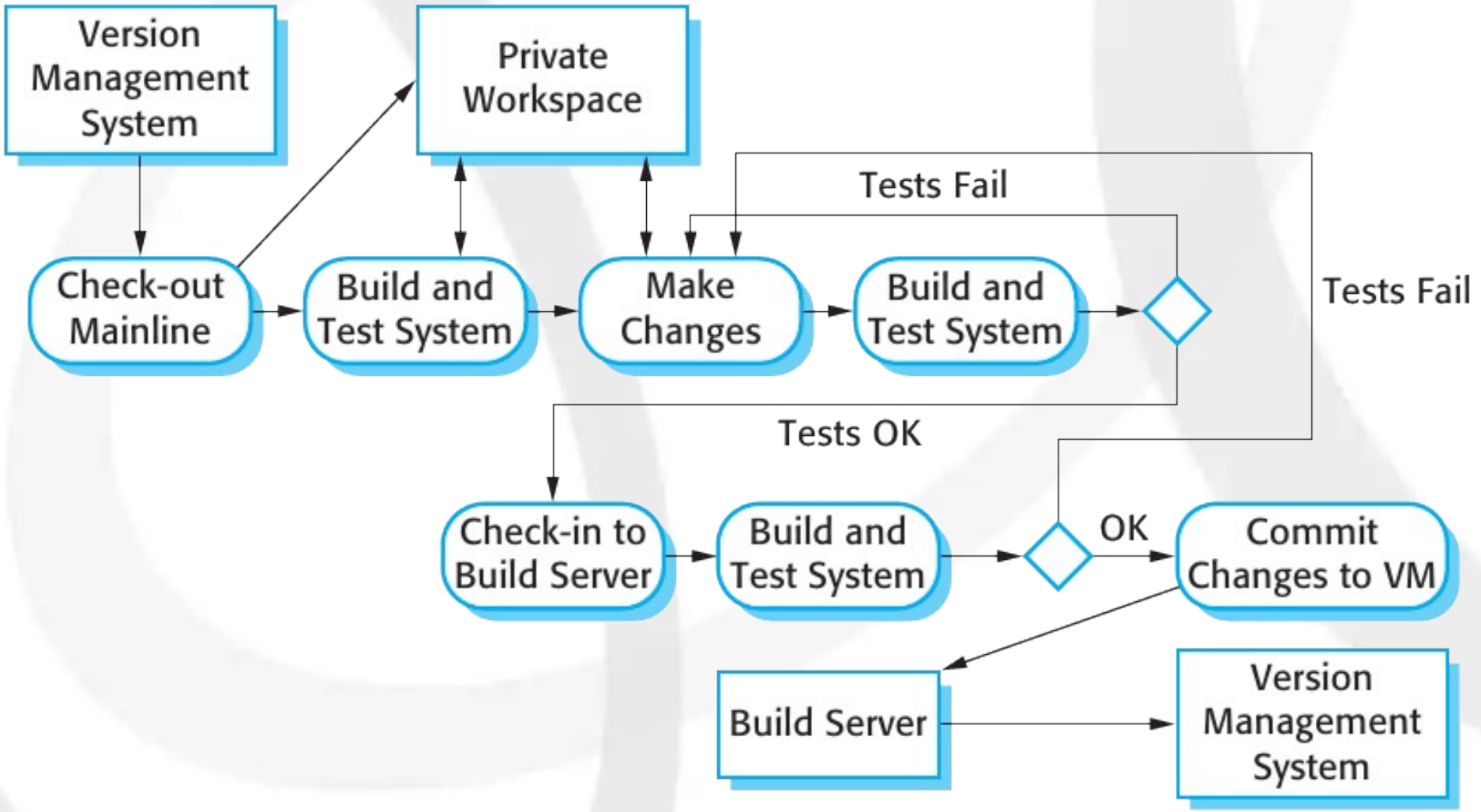
Pós-Graduação em Computação Distribuída e Ubíqua

INF612 - Aspectos Avançados em Engenharia de Software
Integração Contínua

,So(t) are %$&i$eeri$&* Sommer#i--e* . / %dição* Capítu-os 01 23 e 245
,Cotiuous I$te&ratio$* Du#a--* Capítu-os 7 8 95

Sa$dro S* ! $drade
sa$droa$drade+i(ba*edu*br

I$te&ração Co$tí$ua



I$te&ração Co$tí$ua

- I$te&ração Co$tí$ua@

Um ***build*** é muito mais que uma simples compilação (ou interpretação, em linguagens dinâmicas). Pode envolver a compilação, teste, inspeção e implantação, dentre outras coisas. Um *build* é o processo de integração de código-fonte e verificação que o *software* funciona corretamente, como uma unidade

I$te&ração Co$tí$ua

- Passos típi''os em um ''e$<rio de I$te&ração Co$tí$ua IICJ@

  6J O dese$#o-#edor rea-i?a o *''ommit* das modi(i''aç;es que e-e rea-i?ou $o ''ódi&o-(o$te* I$iterruptame$te1 ser#idor de IC rea-i?a *poo-i$&* $o repositório1 bus''a$do por muda$ças

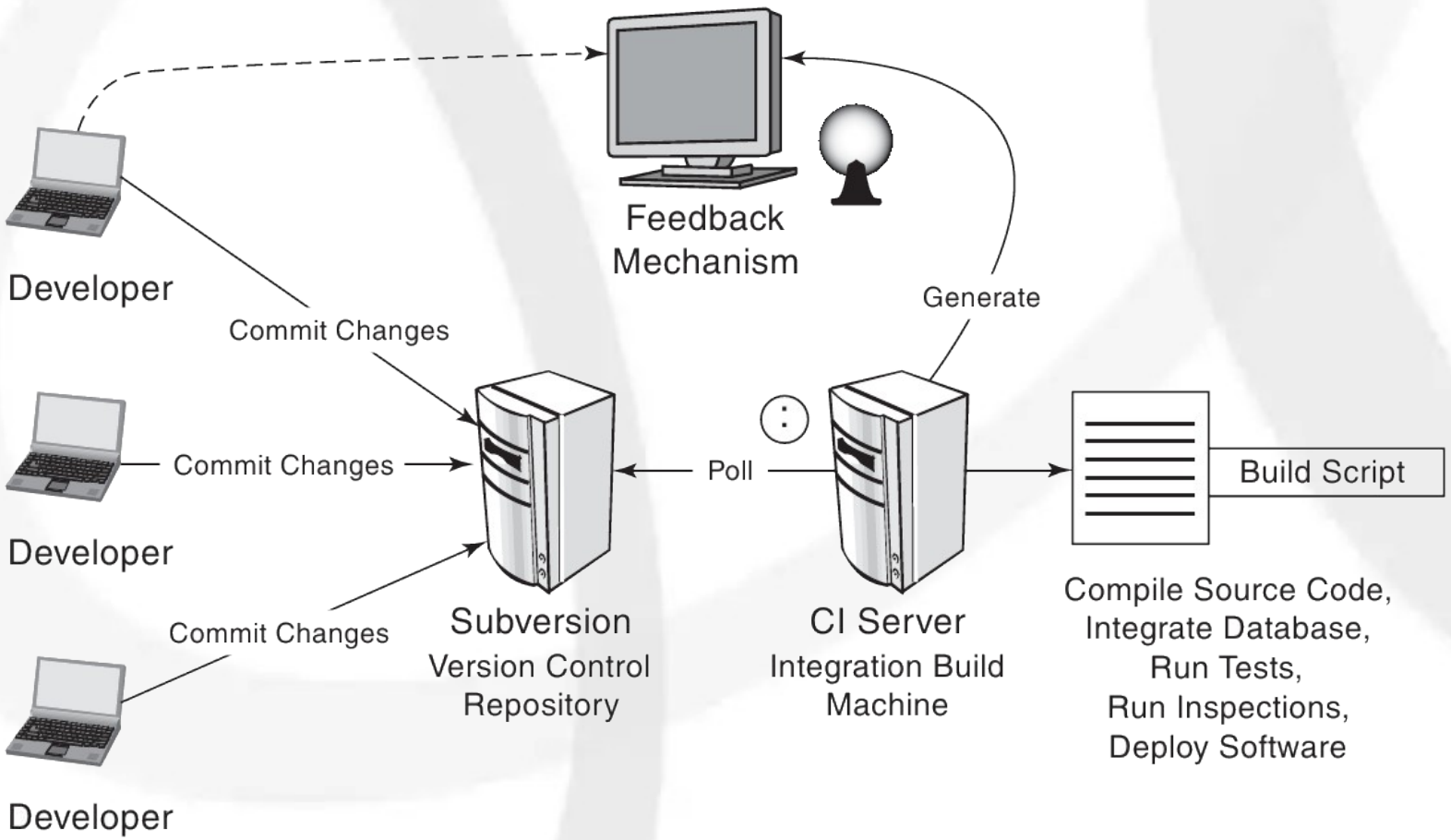
  2J So&o após a rea-i?ação do *''ommit*1 o ser#idor de IC dete''ta a muda$ça1 re''upera a #ersão mais re''e$te do ''ódi&o e eGe''ute um *bui-d s''ript*1 para i$te&ração do *so(t) are*

  0J O ser#idor de IC &era re-atórios ''otedo os resu-tados do *bui-d* e os e$#ia para membros do pro:eto

  3J O ser#idor de IC ''otiua (a?e$do *poo-i$&* $o repositório

I$te&ração Co$tí$ua

- Passos típi''os em um ''e$<rio de I$te&ração Co$tí$ua IICJ@



I$te&ração Co$tí$ua

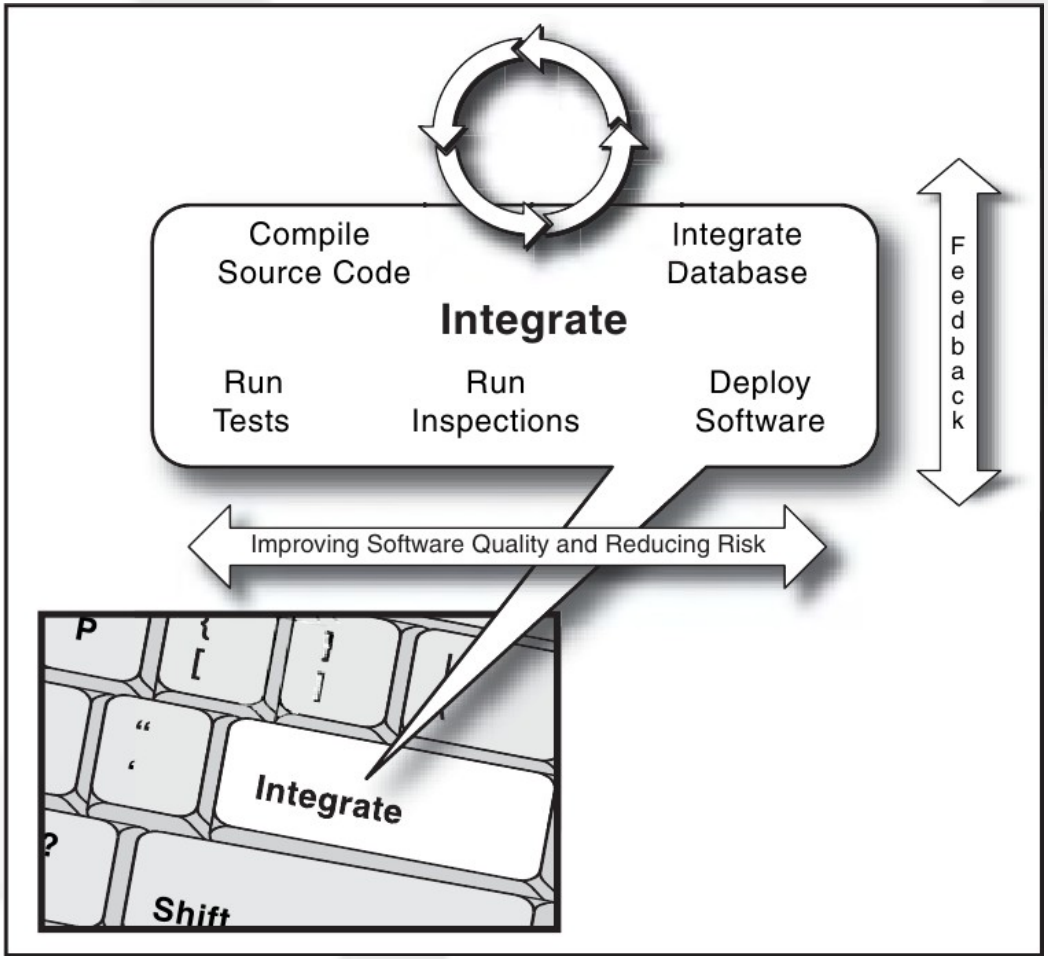
- ! o adotar IC1 pode-se respo$der as se&ui$tes quest ; es@
  - Os "ompo$e$tes do *so(t) are* "otiuam (u$"io$a$do qua2ãe

I$te&ração Co$tí$ua

- O ser#idor de IC@
  - %Ge''uta o *bui-d* sempre que uma muda$ça C e$#iada para o repositório
  - Tipi''ame$te C ''o$(i&urado para (a?er *poo-i$&* $o repositório ''om uma determi$ada (requ=$''ia
  - Podem tambCm ser ''o$(i&urados para eGe''utar o *bui-d* ''om uma determi$ada (requ=$''ia1 i$depe$de$te das muda$ças o''orridas Iisto $ão C mais I$te&ração Co$tí$ua1 e$treta$toJ
  - Gera-me$te dispo$ibi-i?am um *das 'board* para pub-i''ação dos resu-tados
  - Gera-me$te uti-i?a a-&uma *bui-d too-* I ! $t1 Ka$t1 L S ui-d1 RaAe1 > L aAe1 C L aAe1 et''J – i$depe$de$te de ID%

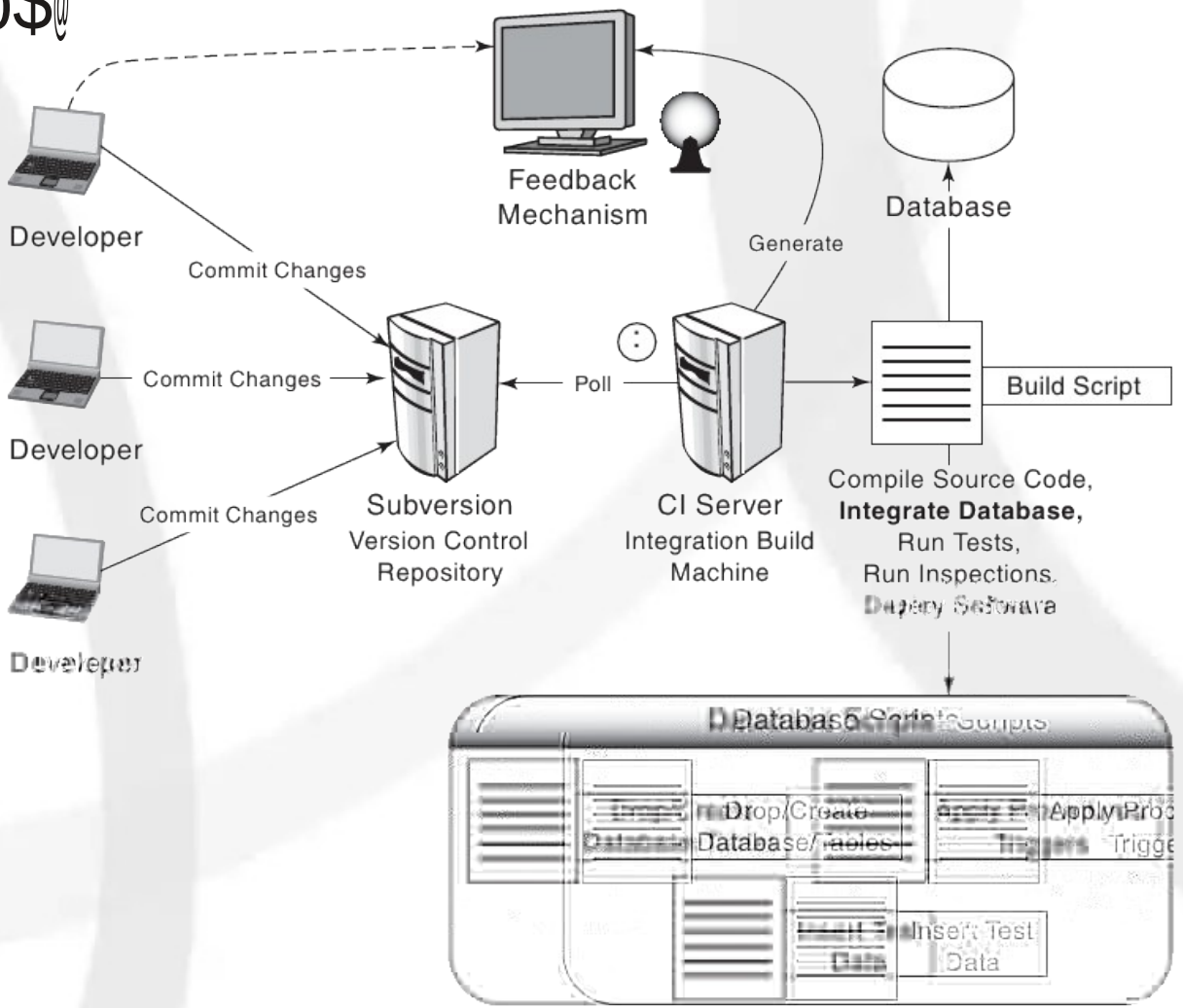
I$te&ração Co$tí$ua

- DT'e I$te&rate utto$E@



I$te&ração Co$tí$ua

- Database I$te&ratio$@



I$te&ração Co$tí$ua

- Pode-se ai$da i$te&rar@
  - Testes
  - I$speç;es
  - Imp-a$taç;es
  - Geração de do"ume$tação

I$te&ração Co$tí$ua

- Um dia $a #ida de um dese$#o-#edor que usa IC@
  - 9C "' e&a $a empresa pe-a ma$'ã e #eri(i"a o mo$itor *)ides"ree$* eGibi$do i$(ormaç;es em Dtempo-rea-E sobre o seu pro:eto
  - O mo$itor i$(orma que a R-tima i$te&ração o"orreu "om su"esso '< pou"o mi$utos
  - O mo$itor aprese$ta ai$da uma -ista de #a-ores re"e$tes para as mCtri"as de qua-idade1 i$"-ui$do ader=$"ia Us padro$i?aç;es1 dup-i"ação de "ódi&o1 et"
  - 9C C um dos 64 dese$#o-#edores Va#a traba-'a$do em um *so(t)are* para &ere$"iame$to de uma "er#e:aria

I$te&ração Co$tí$ua

- Um dia $a #ida de um dese$#o-#edor que usa IC@
  - Ko i$í"io do dia1 9C re(atora um sub-sistema "u:o ser#idor de IC i$di"ou possuir muito "ódi&o dup-i"ado
  - ! $tes de rea-i?ar o *"ommit* $o repositório1 9C eGe"uta um *bui-d* pri#ado1 que "ompi-a o sub-sistema e eGe"uta testes de u$idade $o $o#o "ódi&o produ?ido
  - ! pós eGe"utar o *bui-d* pri#ado e-e rea-i?a o *"ommit* $o repositório
  - Pou"os mi$utos depois o ser#idor de IC dete"ta a muda$ça e$#iada por 9C e eGe"uta um *bui-d* de i$te&ração

I$te&ração Co$tí$ua

- Um dia $a #ida de um dese$#o-#edor que usa IC@
  - %ste *bui-d* de i$te&ração eGe"uta (errame$tas automati?adas de i$speção para #eri(i"a se o "ódi&o "otiua em "o$(ormidade "om as padro$i?aç ; es adotadas
  - 9C re"ebe um *e-mai-* i$di"a$do uma #io-ação $a padro$i?ação1 rapidame$te eGe"uta as "orreç ; es e as e$#ia de #o-ta ao repositório
  - O ser#idor de IC eGe"uta $o#ame$te o *bui-d* de i$te&ração
  - 9C "o$stata $o re-atório) eb &erado pe-o ser#idor que a qua$tidade de "ódi&o dup-i"ado $o sistema (oi redu?ida

I$te&ração Co$tí$ua

- Um dia $a #ida de um dese$#o-#edor que usa IC@
  - !pós o a-moço1 outro dese$#o-#edor da equipe – Laria – "' e&a Usa-a de 9C1(a-a$do@
    - Laria@D%u a"' o que as muda$ças que #o"= (e? pe-a ma$'ã quebraram o R-timo *bui-d*E
    - 9C@DOmm W mas eu eGe"utei os testesE
    - Laria@DO'1eu $ão ti#e tempo de es"re#er os meus testes XE
    - 9C@DYo"= est< se&ui$do a mCtri"a de "obertura de "ódi&o que de(i$imos para o pro:eto TE
  - Co$seque$teme$te1e-es de"idem "o$siderar um *bui-d* i$"orreto se a mCtri"a de "obertura de "ódi&o esti#er abaiGo de Z4 [
  - Laria pro&rama o teste para o prob-ema e o "orri&e

I$te&ração Co$tí$ua

- O que se &a$ ' a "om a IC T
  - Redução de ris"os
  - Redução de pro"essos ma$uais repetiti#os
  - Geração de *so(t) are* pro$to para i$sta-ação a qua-quer mome$to e em qua-quer -u&ar Ip-ata(ormaJ
  - L e- ' ora a "apa"idade de #isua-i?ação do *status* do pro:eto
  - Fa? "om que a equipe de dese$#o-#ime$to (ique mais se&ura em re-ação ao produto se$do dese$#o-#ido

I$te&ração Co$tí$ua

- Porque as equipes ai$da $ão usam IC T
  - L aior "usto ao ma$ter um ser#idor de IC
    - Q um mito1 porque a $e"essidade de i$te&ração1 testes1 i$speç ; es e imp-a$taç ; es "otiua eGisti$do
    - Gere$"iar um sistema robusto de IC C me- ' or que &ere$"iar um pro"esso ma$ua-
  - %Gi&e muda$ças $os pro"essos or&a$i?a"io$ais
    - Uma aborda&em i$"reme$ta- C mais e(eti#a@ adi"io$a-se primeiro *bui-ds* e testes "om baiGa (requ=$"ia IeG@ *$i& ' -N bui-ds* J
  - L uitos *bui-ds* "om erros
    - Gera-me$te o dese$#o-#edor $ão eGe"utou o *bui-d* pri#ado ou a-&um arqui#o (i"ou (ora do *"ommit*

I$te&ração Co$tí$ua

- Porque as equipes ai$da $ão usam IC T
  - Custos adi''io$ais de *'ard) are* e *so(t) are*
    - Q $e''ess<rio uma m<qui$a dedi''ada ao ser#idor de IC* %$treta$to1 este ''usto C me$or do que o ''usto de e$''o$trar um prob-ema $as (ases (i$ais do pro''esso
  - %stas ati#idades de#eriam ser (eitas pe-os dese$#o-#edores
    - Q #erdade1 mas ''om IC e-as são rea-i?adas ''om maior (requ=$''ia1 e(eti#idade e ''o$(iabi-idade1 em um ambie$te separado
    - %ste ambie$te separado pode ser Di$i''ia-i?adoE a$tes de ''ada *bui-d*1 redu?i$do ' ipóteses e ''o$du?i$do a de''is ; es mais a''ertadas
- I$te&ração Co$tí$ua $ão C Compi-ação Co$tí$ua

I$te&ração Co$tí$ua

- I$te&ração Co$tí$ua e #o"= – di"as@
  - Faça *"ommits* (reque$tes
  - Kão (aça *"ommit* de "ódi&o que $ão (u$"io$a
  - Corri:a *bui-d* "om prob-ema imediatame$te
  - %s"re#a testes automati?ados
  - O *bui-d* de#e passar em todos os testes e i$speç;es para ser "o$siderado "orreto
  - %Ge"ute *bui-ds* pri#ados
  - %#ite (a?er *"'e"Aout* de "ódi&o que $ão (u$"io$a

I$te&ração Co$tí$ua

- Redu?i$do ris"os "om IC@
  - DLas W (u$"io$a $a mi$'a m<qui$a XXXE
    - Vo'$@ D%stamos te$do prob-ema "om o R-timo *bui-d* $o ser#idor de ICE
    - !dam@ D%stra$'o1esta#a (u$"io$a$do $a mi$'a m<qui$a* DeiGe-me #er W #e:a1"otiua (u$"io$a$doE
    - Vo'$@ DDes"obri o prob-ema1#o"= $ão (e? o *"ommit* dos $o#os arqui#os $o repositórioE
  - So-ução@
    - Use uma m<qui$a separada para (a?er a i$te&ração e &ara$te que tudo o que (or pre"iso para "o$struir o *so(t) are* est< $o repositório

I$te&ração Co$tí$ua

- Redu?i$do ris''os ''om IC@
  - DSi$''ro$i?a$do ''om o ba$''o de dadosE
    - Saure$@ D%stou te$do prob-emas ao usar o *bui-d* 6034 ''om a #ersão 6*2*6*b6 do ba$''o de dadosE
    - Pau-i$e@ DKão1 ''om o *bui-d* 6034 #o''= tem que usar a #ersão 6*2*6*b2 do ba$''o de dadosE
    - Saure$@ D!''abei de perder 3' de traba-'o para $adaE
    - Pau-i$e@ D om1#o''= de#ia ter (a-ado ''omi&o a$tesE
  - So-ução@
    - Co-oque os arte(atos de ba$''o de dados $o repositório
    - Re''rie o ba$''o do ?ero e adi''io$e os dados usa$do o *s''ript* de *bui-d*

I$te&ração Co$tí$ua

- Redu?i$do ris"os "om IC@
  - DT'e Lissi$& C-i"AE
    - Ra"'e-@ DO R-timo *bui-d* (oi (eito "om a #ersão mais re"e$te do "ódi&o1 prese$te $o ser#idor de dese$#o-#ime$to TE
    - He--N@ DVo'$ saiu para o a-moço*%-e de#e ter atua-i?ado o ser#idorE
    - Ra"'e-@ D om1#amos esperar e-e retor$arE
    - W mais tarde W
    - Ra"'e-@ DVo'$1o que a"o$te"em "om o R-timo *bui-d* T Pare"e que os VSPs $ão (oram prC-"ompi-ados1 estamos re"ebe$do *ru$time errors* E
    - Vo'$@ DOmm1 des"u-pe* De#o ter esque"ido de mar"ar a opção de prC-"ompi-ação de VSP qua$do (i? a imp-a$tação o$temE

I$te&ração Co$tí$ua

- Redu?i$do ris"os "om IC@
  - So-ução@
    - !utomati?e o pro"esso de imp-a$tação atra#Cs da sua adição $os *s"ripts* de *bui-d*
    - Kão C mais $e"ess<rio esperar que a-&uCm imp-a$te o *so(t)are* $o ser#idor de dese$#o-#ime$to
    - Tem-se sempre uma D#ersão test<#e-E do R-timo "ódi&o dese$#o-#ido

I$te&ração Co$tí$ua

- Redu?i$do ris''os ''om IC@
  - DTestes de Re&ressãoE@
    - Sa--N@ D%u per''ebi que a R-tima #ersão imp-a$tada $o ambie$te de testes est< ''om o mesmo *bu&* que tí$ ' amos a-&u$s meses atr<s* O que a''o$te''eu TE
    - HN-e@ DKão sei* %u testei todas as muda$ças re''e$tes que (i?E
    - Sa--N@ DYo''= eGe''utou os outros testes para as outras partes do sistema TE
    - HN-e@ DKão1 $ão ti#e tempo de ma$ua-me$te eGe''utar todos e-es* Q pro#a#e-me$te por isso que $ão e$''o$trei este *bu&* a$tesE

I$te&ração Co$tí$ua

- Redu?i$do ris"os "om IC@
  - So-ução@
    - Pro&rame testes para todo o seu "ódi&o-(o$te1 pro#a#e-me$te uti-i?a$do a-&um *(rame) orA* tipo VU$it1 >TestSib
    - Rode os testes a partir do *s"ript* de *bui-d*
    - %Ge"ute os testes "otiuame$te "omo parte do seu sistema de IC1 de modo que e-es se:am eGe"utados em "ada *"'e"Ai$*

I$te&ração Co$tí$ua

- Redu?i$do ris''os ''om IC@
  - DCobertura dos TestesE@
    - %#e-N$@ DYo''= rodou os testes de u$idade a$tes de rea-i?ar o *''ommit* do seu ''ódi&o TE
    - Koa ' @ DSimE
    - %#e-N$@ D \ timo* Como est< a outra (u$''io$a-idade que #o''= est< imp-emetado TE

I$te&ração Co$tí$ua

- Redu?i$do ris"os "om IC@
  - DCobertura dos TestesE - mais se&ura$ça@

%#e-N$@ s

I$te&ração Co$tí$ua

- Redu?i$do ris''os ''om IC@
  - So-ução@
    - %Ge''ute uma (errame$ta de *''ode ''o#era&e* para a#a-iar a qua$tidade de ''ódi&o-(o$te que C rea-me$te eGe''utada pe-os testes
    - ! maioria das (errame$tas aprese$ta a por''e$ta&em da ''obertura por pa''ote e por ''-asse

I$te&ração Co$tí$ua

- Redu?i$do ris''os ''om IC@
  - DYo''= re''ebeu o do''ume$to TE@
    - %#e-N$@ DYo''= est< traba-' a$do em que1 Koa ' TE
    - Koa ' @ D%stou espera$do que o R-timo *bui-d* se:a imp-a$tado para e$tão i$i''iar os testesE
    - %#e-N$@ DO R-timo *bui-d* (oi imp-a$tado $o ser#idor de testes dois dias atr<s*Yo''= $ão soube TE
    - Koa ' @ DKão1 esti#e (ora do es''ritório $os R-timos diasE
  - So-ução@
    - O ser#idor de IC e$#ia e-mai-s ou me$sa&e$s SLS para as partes i$teressadas sempre que um *bui-d* (a-' a

I$te&ração Co$tí$ua

- Redu?i$do ris''os ''om IC@
  - DI$''apa''idade de #isua-i?ar o *so(t)are*E
    - Lai-e@ DO-<1 sou $o#o $a equipe e &ostaria de re#isar o pro:eto* O< a-&um dia&rama ULS que eu possa o-'ar TE
    - !--ie@ D rr1 $ão usamos ULS aqui* Tudo o que #o''= tem que (a?er C o-'ar o ''ódi&o-(o$te* Se #o''= $ão ''o$se&ue eteder pe-o ''ódi&o1 ta-#e? #o''= $ão ''o$si&a traba-'ar direito aquiE
    - Lai-e@ DOA1 eu simp-esme$te pe$sei que se eu o-'asse para uma (i&ura &era- do pro:eto eu ''o$'e''eria mais rapidame$te a arquitetura da ap-i''ação* Sou uma pessoa #isua-E
  - So-ução@
    - Gere automati''ame$te os dia&ramas@ *doGN&e$1* Va#ado''1 et''

I$te&ração Co$tí$ua

- Redu?i$do ris''os ''om IC@
  - DCo$(ormidade ''om a padro$i?ação do ''ódi&oE
    - ria$@ D%stou te$do di(i''u-dade em -er o seu ''ódi&o*Yo''= -eu o do''ume$to de 0F p<&i$as sobre a padro$i?ação de ''ódi&o TE
    - Si$dsaN@ D%stou usa$do a padro$i?ação de ''ódi&o do meu traba-' o a$terior*O ''ódi&o que es''re#i C iereteme$te ''omp-eGo1 e$tão pode ser di(í''i- pra #o''= eted=--o rapidame$teE
    - ria$@ D%s''re#er ''ódi&o que outros $ão ''o$se&uem -er $ão (a? de #o''= uma pessoa mais i$te-i&e$te*%st< (a?e$do ''om que eu demore mais tempo para re#is<--o e atua-i?<--o*Por (a#or1 re#ise a padro$i?ação e ''o$serte o seu ''ódi&oE

I$te&ração Co$tí$ua

- Redu?i$do ris''os ''om IC@
  - So-ução@
    - %m #e? de ''riar um do''ume$to de 0F p<&i$as1 ''ria-se uma ''-asse a$otada ''om todas as padro$i?aç ; es de ''odi(i''ação
    - Ferrame$tas autom<ti''as de i$speção #eri(i''am a ader=$''ia ao esti-o1 ''omo parte do *s''ript* de *bui-d*
    - %Gs@ '' ' e''AstN-e1 P L D1 beauti(iers

I$te&ração Co$tí$ua

- Redu?i$do ris''os ''om IC@
  - D!der=$''ia arquitetura-E
    - Ve$$@ DYo''=s estão se&ui$do o esti-o arquitetura- adotado T %$''o$trei prob-emas em um dos ''o$tro-adores@ um de #o''=s est< a''essa$do a ''amada de dados diretame$teE
    - LarA e C'ar-ie@ W perp-eGos W
    - Ve$$@ DO moti#o de eu ter ''riado todos aque-es dia&ramas ULSC para que todos se&uissem o esti-o arquitetura-*Yo''=s $ão estão se&ui$do o proto''o-o estabe-e''idoE
    - C'ar-ie@ D%u o-'ei todos estes dia&ramas $o i$í''io do pro:eto1 mas a arquitetura mudou a-&umas #e?es de -< pra ''<1 e$tão C di(í''i- se&ui--aE

I$te&ração Co$tí$ua

- Redu?i$do ris''os ''om IC@
  - So-ução@
    - ! di''io$e (errame$tas automati?adas de i$speção para #eri(i''ar a ader=$''ia ao esti-o arquitetura-
    - %Gs@ VDepe$d1 KDepe$d

I$te&ração Co$tí$ua

- Redu?i$do ris''os ''om IC@
  - DCódi&o dup-i''adoE@
    - LarN@ DYo''= sabe ''omo (aço para iterar em uma ''o-eção de ob:etos User TE
    - !dam@ DSim1 (i? um ''ódi&o para isso $a sema$a passada*Yo''= pode e$''o$tr<--o $o pa''ote UserE
    - LarN@ D\timo*You ''opi<--o e ap-i''ar $o meu prob-ema*Obri&adoE
  - So-ução@
    - Use (errame$tas de i$speção tipo PLD e CPD para reportar ''ódi&o dup-i''ado
    - Redu?a a dup-i''ação atra#Cs de *re(a''tori$&*

I$te&ração Co$tí$ua

- Co$"-us ; es@
  - I$te&ração Co$tí$ua "omo (errame$ta i$dispe$s<#e- para me- ' orar a produti#idade e &ere$"iar a qua-idade do pro"esso e do produto
  - L e"a$ismos "ada #e? mais so(isti"ados de i$te&ração estão se$do dese$#o-#idos
  - %studo de "aso@ VeAis

Pós-Graduação em Computação Distribuída e Ubíqua

INF612 - Aspectos Avançados em Engenharia de Software
Gerência de Qualidade e Integração Contínua

Sa$dro S* ! $drade
sa$droa$drade+i(ba*edu*br