

# Vývoj softvéru v tíme

Nástroje pre manažment verzií, riadenie životného  
cyklu zostavovania softvéru, riadenie zmien

Michal Tvarožek, tvarozek { at } fiit.stuba.sk, D208

# Obsah

---

- ▶ Integrované vývojové prostredia
- ▶ Manažment verzií (zdrojového kódu)
- ▶ Zostavovanie softvéru, kontinuálna integrácia
- ▶ Manažment zmien
- ▶ Podpora komunikácie v tíme
- ▶ Odporúčania

# Obsah

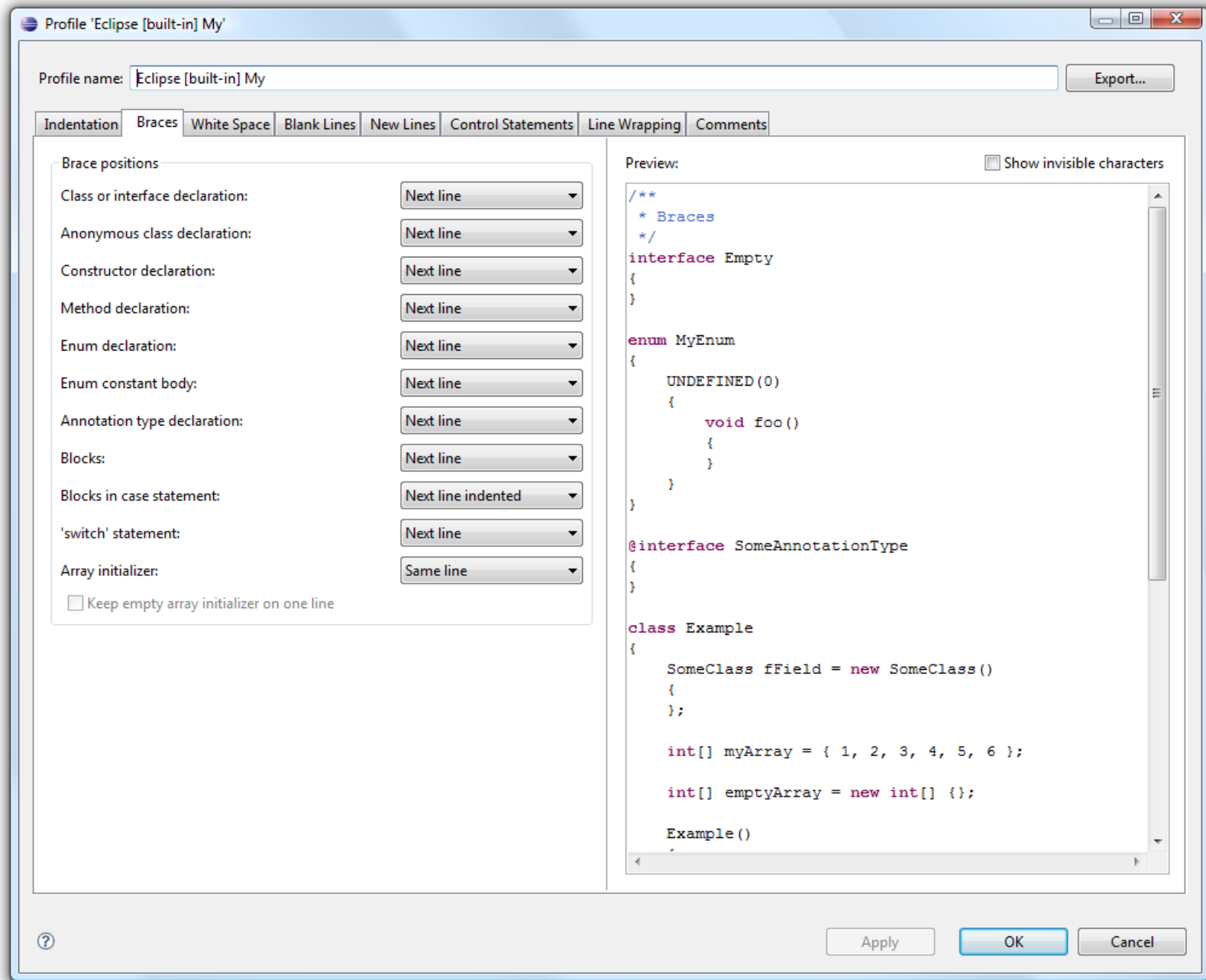
---

- ▶ **Integrované vývojové prostredia**
- ▶ Manažment verzií (zdrojového kódu)
- ▶ Zostavovanie softvéru, kontinuálna integrácia
- ▶ Manažment zmien
- ▶ Podpora komunikácie v tíme
- ▶ Odporúčania

# Integrované vývojové prostredia

---

- ▶ **IDE = Integrated Development Environment**
- ▶ Centrálna pracovná plocha pre vývojový tím
  - ▶ Eclipse, NetBeans (Java)
  - ▶ Visual Studio [Team System], MonoDevelop (C#/.NET)
  - ▶ C++ Builder, Borland Delphi, IDE pre RoR...
- ▶ Metodiky pre písanie zdrojového kódu, ...
- ▶ Šablóna zdrojového kódu (opis, licencia, ...)
- ▶ Formátovanie zdrojového kódu [ukážka v Eclipse]



# Funkcie IDE (ukážka)

---

- ▶ Tvorba zdrojového kódu
- ▶ Ladenie a testovanie
- ▶ Zostavovanie projektov a riešení
- ▶ Tvorba dokumentácie
- ▶ Manažment verzií (zdrojového kódu)
- ▶ Návrh databázových schém

# Obsah

---

- ▶ Integrované vývojové prostredia
- ▶ **Manažment verzií (zdrojového kódu)**
- ▶ Zostavovanie softvéru, kontinuálna integrácia
- ▶ Manažment zmien
- ▶ Podporu komunikácie v tíme
- ▶ Odporúčania

# Manažment verzií

---

- ▶ SCM – **S**ource **C**onfiguration **M**anagement
- ▶ Riadenie viacerých **verzií** tej istej **jednotky**  
**informácií** s ktorou pracuje **viacero ľudí**
  - ▶ **Zdrojového kódu**
  - ▶ Projektovej dokumentácie
  - ▶ Modelov, obrázkov, ľubovoľných dokumentov...



# Prečo verziovať zdrojový kód<sup>1</sup>?

---

- ▶ **Súčasná** práca viacerých členov tímu
  - ▶ Práca nad najnovším zdrojovým kódom
- ▶ Práca na **viacerých verziách** súčasne
  - ▶ Viacero zákazníkov používa rôzne verzie vášho softvéru, potreba udržiavať všetky
- ▶ Návrat k **predchádzajúcim verziám**
  - ▶ Možnosť programovať „uvoľnenejšie“ – v prípade „slepej“ cesty sa môžeme kedykoľvek vrátiť k poslednej dobrej verzii

# Prečo verziovať zdrojový kód<sup>2</sup>?

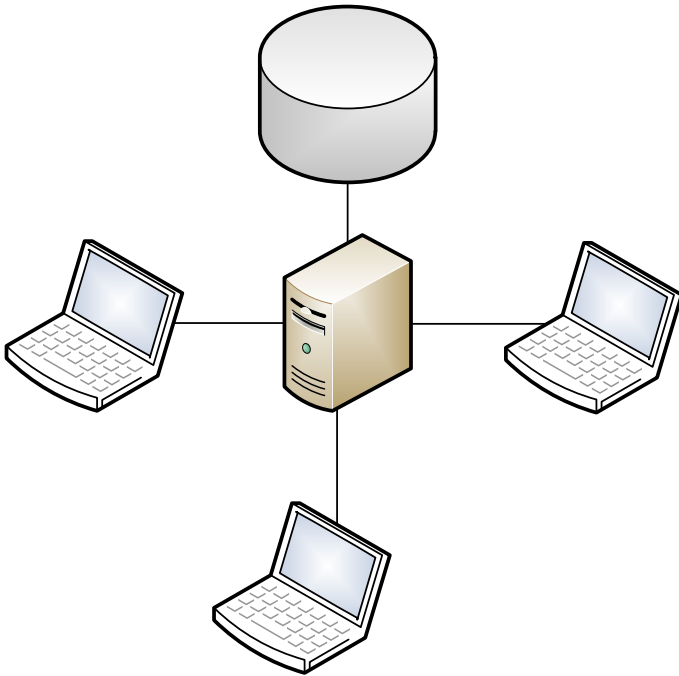
---

- ▶ Tvorba **záložných kópií** dokumentov
  - ▶ Strata/poškodenie lokálnej kópie nie sú kritické
- ▶ **Sledovanie zmien** dokumentov
  - ▶ Kto zmenu vykonal, čo zmenil, kedy to spravil (história)
  - ▶ Riešenie konfliktov, zlučovanie verzií
- ▶ **Sledovanie pokroku**
  - ▶ Monitorovanie zmien, aktivity a práce projektovým manažérom

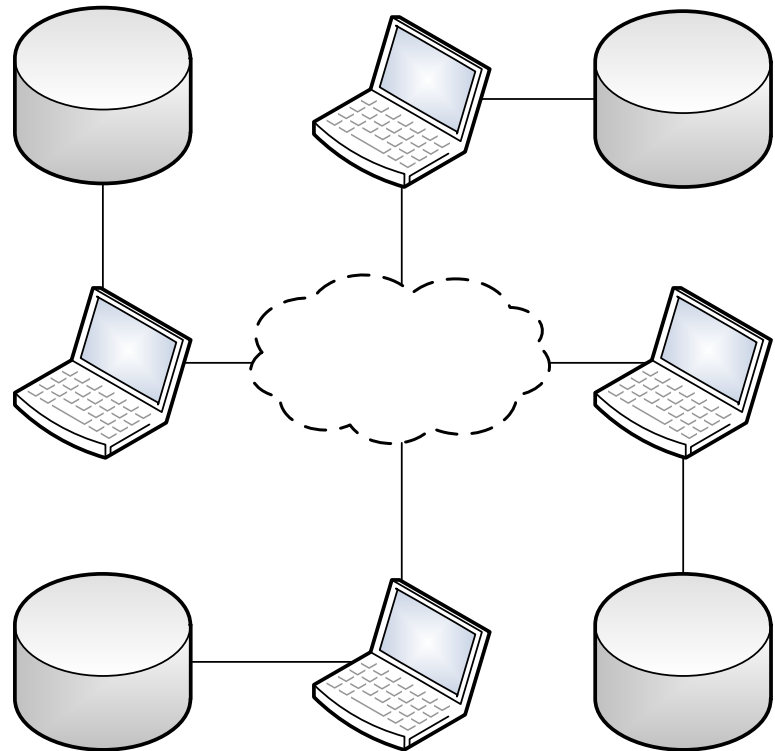
# Prístupy k verziovaniu<sup>1</sup>

## ► Podľa typu úložiska

### Centralizovaný



### Distribúovaný

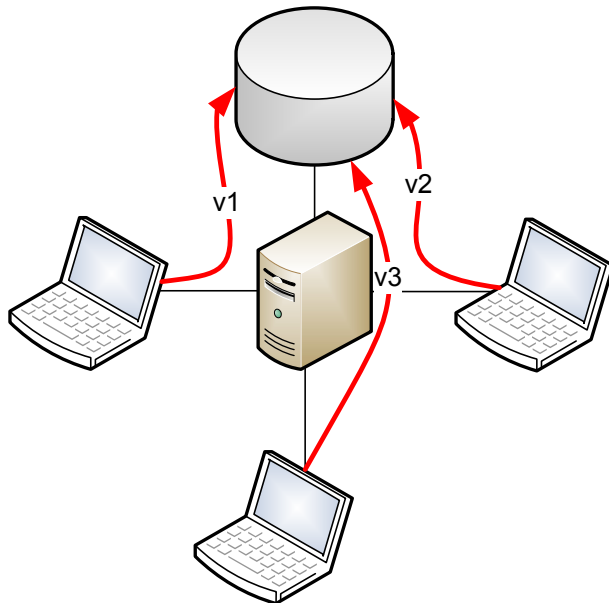


# Prístupy k verziovaniu<sup>2</sup>

- ▶ Podľa spôsobu riadenia zmien
  - ▶ Uzamykanie súborov (len jeden naraz)

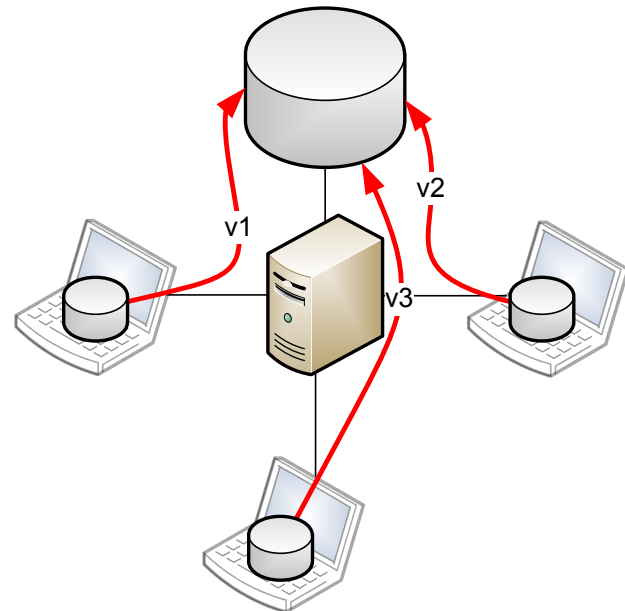
## Spájanie verzii

- ▶ Riešenie konfliktov



## Spájanie úložísk

- ▶ Každý za seba, všetci za?



# Nástroje pre manažment verzií<sup>1</sup>

---

- ▶ Klient – server architektúra
- ▶ Server
  - ▶ Úložisko údajov
  - ▶ Oddelená databáza pre každý projekt
  - ▶ Správa používateľov a prístupových práv
- ▶ Klient
  - ▶ Prehliadanie úložiska, prezeranie zmien, porovnávanie
  - ▶ Práca so súbormi (commit, update, checkout, ...)

# Nástroje pre manažment verzií<sup>2</sup>

---

- ▶ Voľne dostupné systémy
  - ▶ S centralizovaným prístupom
    - ▶ CVS, **Subversion** (SVN), Aegis, ...
  - ▶ S decentralizovaným prístupom
    - ▶ Mercurial, Git, svk, GNU Arch, Bazaar-NG, Monotone, ...
- ▶ Komerčné systémy
  - ▶ IBM Rational ClearCase
  - ▶ Microsoft Visual SourceSafe
  - ▶ Microsoft Studio Team System
  - ▶ SourceGear Vault
- ▶ Iné
  - ▶ Office (TRK), Wiki, ...

# Subversion (SVN)

---

- ▶ Open source (<http://subversion.tigris.org>)
- ▶ Centrálné úložisko
- ▶ Inkrementálne číslovanie revízií (1, 2, ..., 345, ...)
- ▶ Uchovávanie verzií adresárov
- ▶ Atomické potvrdenia zmien (commit)
- ▶ Uchovávanie verzií metadát (súborov, adresárov)
- ▶ Voľba sieťových vrstiev (file, http, https, svn, svn+ssh)
- ▶ Efektívny branching a tagging (konštantný čas)
- ▶ Vyvíjaný s cieľom nahradiť CVS

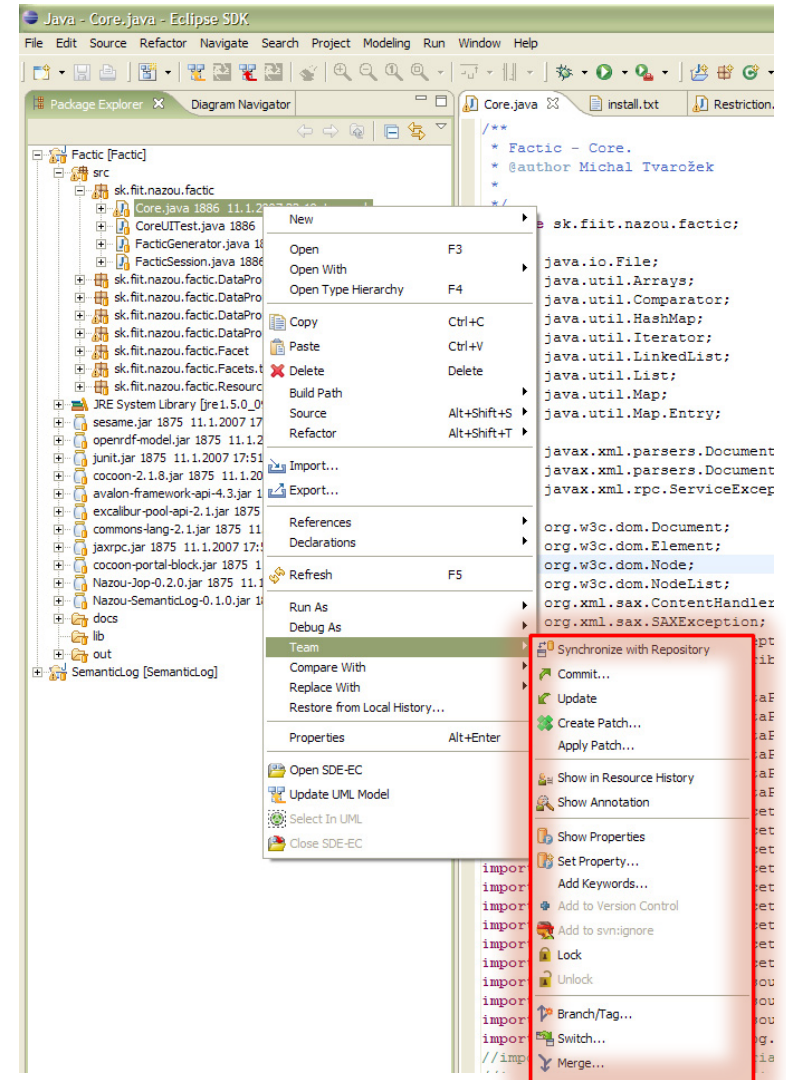
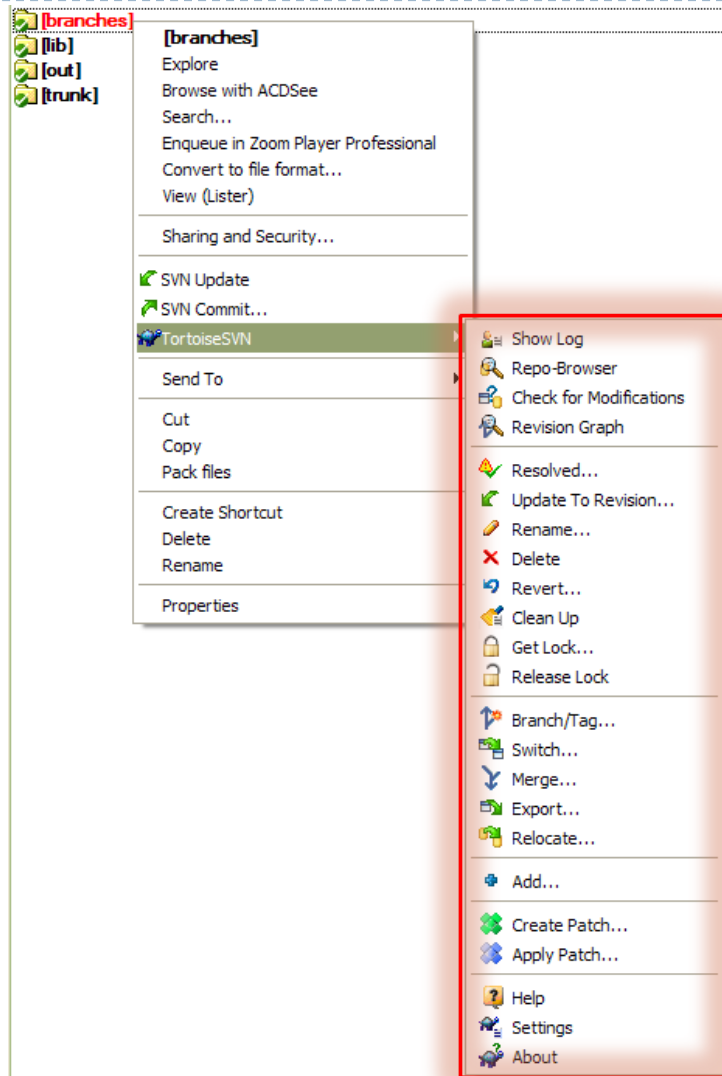
# Použitie Subversion

---

- ▶ Klient v príkazovom riadku
- ▶ Klient s GUI – TortoiseSVN ([tortoisesvn.tigris.org](http://tortoisesvn.tigris.org))
- ▶ Integrácia do vývojových prostredí  
(Eclipse, NetBeans, JetBrains IDEA, VS2008, ...)

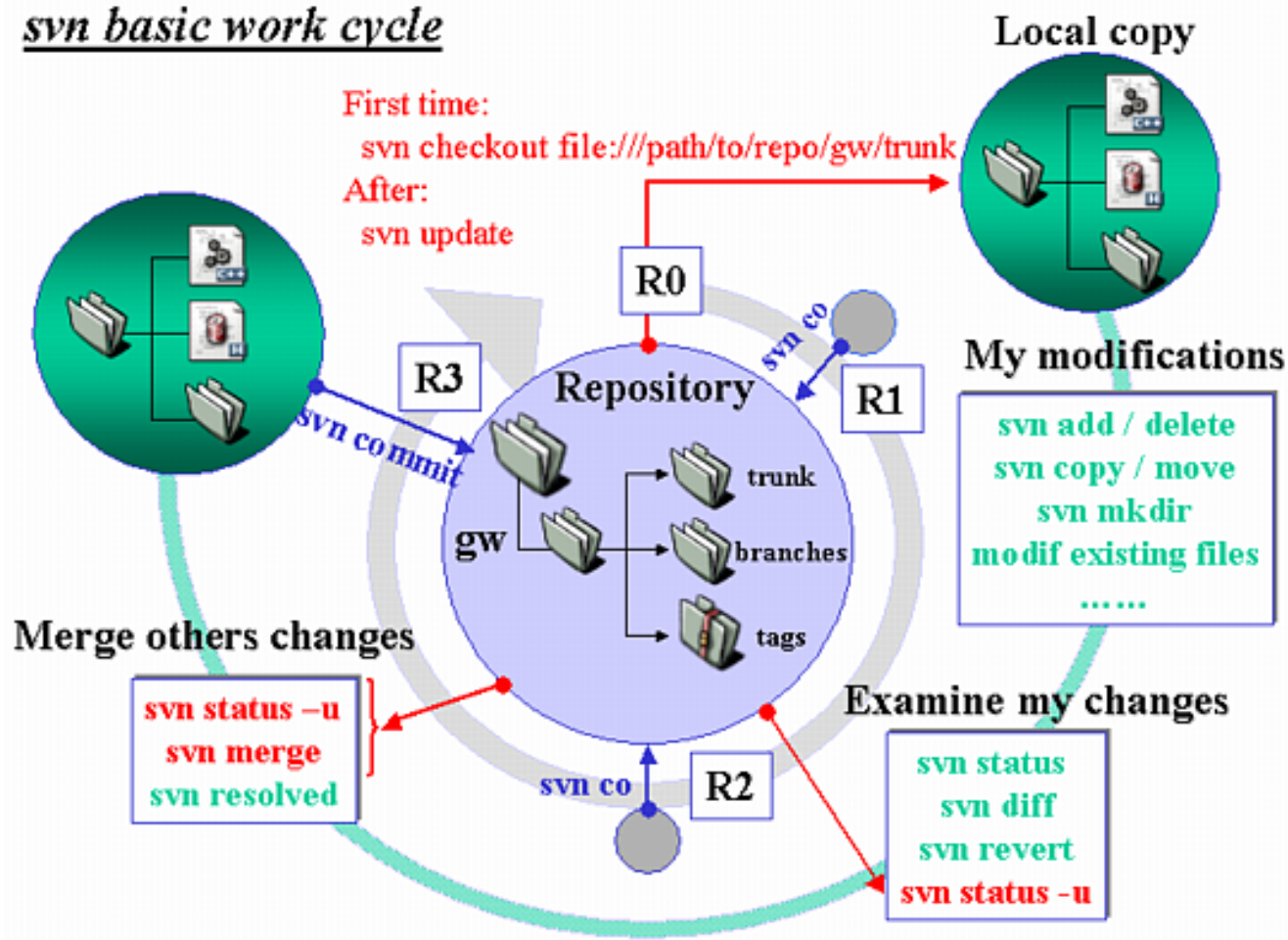


# SVN v TortoiseSVN a Eclipse



# Bežný pracovný cyklus (ukážka)

## svn basic work cycle



**Zdroj:** <http://agata.in2p3.fr/dotclear/index.php?2005/05/13/6-subversion-svn-quickstart>

# Mercurial | Git

---

- ▶ [www.selenic.com/mercurial/](http://www.selenic.com/mercurial/)
- ▶ <http://git-scm.com/>
- ▶ code.google.com, sourceforge.net, bitbucket.org
- ▶ github.com – social coding
  
- ▶ Klient v príkazovom riadku (hg|git)
- ▶ Klient v GUI – tortoise (hg|git)
- ▶ Integrácia do vývojových prostredí
  
- ▶ Možnosť konfigurácie „merge“ nástroja na riešenie konfliktov

# Distribúované úložisko

---

- ▶ Vývojár si kopíruje/klonuje úložisko k sebe
  - ▶ `hg|git clone` <cesta-k-ulozisku>
- ▶ Prípadne vytvorí úložisko z existujúceho adresára
  - ▶ `hg|git init`
- ▶ Prístup cez niekoľko protokolov (schém)
  - ▶ `http`, `ssh`
  - ▶ Ale aj priamym napísaním adresára s úložiskom

# Bežný pracovný cyklus

---

1. Aktualizuj svoje lokálne úložisko o zmeny z rodičovského úložiska
  - `hg|git pull` (prípadne použi predtým `hg incoming`, `git fetch`)
2. Postupuj ako keby si pracoval s SVN, len namiesto `svn` daj `hg|git` 😊
3. Prešír zmeny do rodičovského úložiska
  - `hg|git push`

# Prečo sú distribuované SCM lepšie?

---

- ▶ **Odstráni sa centrálny bod zlyhania**
  - ▶ Všetci majú u seba kompletnú históriu
- ▶ **Podpora lepšieho štýlu vývoja**
  - ▶ Časté commity (lokálne), fungujú dokonca aj offline
  - ▶ Všetko je branch, takže nemá význam sa toho báť
- ▶ **Väčšia sloboda vývoja**
  - ▶ Ak vám niekto dá read-only prístup k SVN, tak nič necommitnete
  - ▶ Ak vám niekto dá read-only prístup ku Gitu, tak jediné, čo nepôjde je push – svoje zmeny môžete verziovať a potom napr. poslať patch
  - ▶ Social coding na [github.com](https://github.com) (fork, pull request)

# Odporúčanie: **určite používať!**

---

## ▶ Čo robiť

- ▶ Často vkladat' zmeny do úložiska (commit)
- ▶ Komentovat' zmeny
- ▶ Robiť malé zmeny
- ▶ Vkladať veci funkčné a bez chýb

## ▶ Čoho sa vystríhať

- ▶ Riešenie konfliktov
- ▶ Vloženie (commit) nefunkčných zdrojových kódov
- ▶ Práca s neaktuálnymi súbormi

# Obsah

---

- ▶ Integrované vývojové prostredia
- ▶ Manažment verzií (zdrojového kódu)
- ▶ **Zostavovanie softvéru, kontinuálna integrácia**
- ▶ Manažment zmien
- ▶ Podpora komunikácie v tíme
- ▶ Odporúčania



# Manažment zostavovania softvéru

---

- ▶ Build = zostavenie [Clean build, incremental build]
- ▶ Automatizácia procesu **zostavovania** softvéru (kompilácie, testovania a distribúcie projektov)
- ▶ Odstránenie závislosti od IDE
- ▶ Založené na vykonávaní definovaných cieľov
  
- ▶ Voľne dostupné nástroje:
  - ▶ Make, Cmake, Rake
  - ▶ **Ant**, NAnt
  - ▶ **Maven**

# Ant

---

- ▶ <http://ant.apache.org> (Java)
- ▶ <http://nant.sourceforge.net/> (MS .NET)
- ▶ Založený na XML **skripte** (build.xml) riadenom definovanými **cieľmi** (targets) a ich vzájomnými závislosťami
- ▶ Ciele obsahujú jednotlivé **úlohy** (tasks), ktoré sú postupne vykonávané

```
- <project name="MyProject" default="dist" basedir=".>
  <description>simple example build file</description>
  <!-- set global properties for this build -->
  <property name="src" location="src" />
  <property name="build" location="build" />
  <property name="dist" location="dist" />
- <target name="init">
  <!-- Create the time stamp -->
  <tstamp />
  <!-- Create the build directory structure used by compile -->
  <mkdir dir="${build}" />
</target>
- <target name="compile" depends="init" description="compile the source">
  <!-- Compile the java code from ${src} into ${build} -->
  <javac srcdir="${src}" destdir="${build}" />
</target>
- <target name="dist" depends="compile" description="generate the distribution">
  <!-- Create the distribution directory -->
  <mkdir dir="${dist}/lib" />
  <!-- Put everything in ${build} into the MyProject-${DSTAMP}.jar file -->
  <jar jarfile="${dist}/lib/MyProject-${DSTAMP}.jar" basedir="${build}" />
</target>
- <target name="clean" description="clean up">
  <!-- Delete the ${build} and ${dist} directory trees -->
  <delete dir="${build}" />
  <delete dir="${dist}" />
</target>
</project>
```

# Maven

---

- ▶ <http://maven.apache.org>
- ▶ Riadi, resp. pomáha pri automatizovaní
  - ▶ Zostavovania projektu (predefinované fázy)
  - ▶ Technickej dokumentácie (xref, javadoc, xdoc)
  - ▶ Súhrnných správ (changes, file activity, unit tests)
  - ▶ Závislostí na iných knižniciach
  - ▶ SCM (SVN, CVS, ...)
  - ▶ Zverejňovania oficiálnych verzií (releases)
  - ▶ Tvorby distribúcie (jar, war, ear, OSGi bundles)

# Predvolené fázy zostavenia projektu

---

- ▶ Validate
  - over vstupy
- ▶ Compile
  - prelož zdrojový kód
- ▶ Test
  - testovanie jednotiek
- ▶ Package
  - vytvorenie distribúcie
- ▶ Integration-test
  - vloženie do test. rámca
- ▶ Verify
  - kontrola kvality výstupu
- ▶ Install
  - inštalácia do lok. úložiska
- ▶ Deploy
  - nasadenie „von“

# Kontinuálna integrácia<sup>1</sup>

---

- ▶ CI – **C**ontinuous **I**ntegration
- ▶ Priebežne sleduje zmeny
- ▶ Automaticky zostavuje a testuje projekt
- ▶ Informuje o priebehu zostavovania projektu
- ▶ Umožňuje všetkým okamžite vidieť aktuálnu zostavu

# Kontinuálna integrácia<sup>2</sup>

---

## ► Existujúce nástroje

- **CruiseControl** (<http://cruisecontrol.sourceforge.net>)
- **CruiseControl.NET** (<http://ccnet.thoughtworks.com>)
- **Continuum** (<http://maven.apache.org/continuum>)
- **Gump** (<http://gump.apache.org>)

## ► Porovnanie vybraných nástrojov

<http://confluence.public.thoughtworks.org/display/CC/CI+Feature+Matrix>

Continuum

About

Show Projects

Add Project

Maven 2.0+ Project

Maven 1.x Project

Ant Project

Shell Project

Administration

Schedules

Configuration

User Groups Management

Users Management

Legend

Build Now

Build History

Build In Progress

Checking Out Build

Queued Build

Delete

Edit

Build in Success













Build in Failure

Build in Error

→ Continuum Projects

	Name	Version	Build	Group	
	<a href="#">Continuum API</a>	1.1-SNAPSHOT	<u>3</u>	Continuum Parent Project	
	<a href="#">Continuum Core</a>	1.1-SNAPSHOT	0	Continuum Parent Project	
	<a href="#">Continuum Core Integration Test</a>	1.1-SNAPSHOT	0	Continuum Parent Project	
	<a href="#">Continuum Cruise Control Importer</a>	1.1-SNAPSHOT	0	Continuum Parent Project	
	<a href="#">Continuum IRC Notifier</a>	1.1-SNAPSHOT	<u>3</u>	Continuum Parent Project	
	<a href="#">Continuum Jabber Notifier</a>	1.1-SNAPSHOT	<u>3</u>	Continuum Parent Project	
	<a href="#">Continuum MSN Notifier</a>	1.1-SNAPSHOT	<u>3</u>	Continuum Parent Project	
	<a href="#">Continuum Model</a>	1.1-SNAPSHOT	<u>4</u>	Continuum Parent Project	
	<a href="#">Continuum Notifier API</a>	1.1-SNAPSHOT	<u>3</u>	Continuum Parent Project	
	<a href="#">Continuum Notifiers</a>	1.1-SNAPSHOT	<u>5</u>	Continuum Parent Project	
	<a href="#">Continuum Parent Project</a>	1.1-SNAPSHOT	<u>9</u>	Continuum Parent Project	
	<a href="#">Continuum Store</a>	1.1-SNAPSHOT	0	Continuum Parent Project	
	<a href="#">Continuum Test</a>	1.1-SNAPSHOT	<u>3</u>	Continuum Parent Project	
	<a href="#">Continuum Web APP</a>	1.1-SNAPSHOT	0	Continuum Parent Project	
	<a href="#">Continuum XMLRPC Interface</a>	1.1-SNAPSHOT	0	Continuum Parent Project	
	<a href="#">Doxia</a>	1.0-alpha-8-SNAPSHOT	<u>8</u>	Doxia	
	<a href="#">Doxia Confluence Module</a>	1.0-alpha-8-SNAPSHOT	0	Doxia	
	<a href="#">Doxia Core</a>	1.0-alpha-8-SNAPSHOT	0	Doxia	
	<a href="#">Doxia Decoration Model</a>	1.0-alpha-8-SNAPSHOT	<u>4</u>	Doxia	
	<a href="#">Doxia Docbook Simple Module</a>	1.0-alpha-8-SNAPSHOT	0	Doxia	
	<a href="#">Doxia Modules</a>	1.0-alpha-8-SNAPSHOT	<u>4</u>	Doxia	
	<a href="#">Doxia Sink API</a>	1.0-alpha-8-SNAPSHOT	<u>4</u>	Doxia	
	<a href="#">Doxia Site Renderer Component</a>	1.0-alpha-8-SNAPSHOT	<u>4</u>	Doxia	
	<a href="#">Doxia TWiki Module</a>	1.0-alpha-8-SNAPSHOT	0	Doxia	
	<a href="#">Maven</a>	2.1-SNAPSHOT	<u>4</u>	Maven JXR	
	<a href="#">Maven</a>	2.0.3-SNAPSHOT	<u>6</u>	Maven JXR	
	<a href="#">Maven Ant Mojo Support</a>	2.1-SNAPSHOT	<u>3</u>	Maven JXR	
	<a href="#">Maven Ant Mojo Support</a>	2.0.3-SNAPSHOT	<u>2</u>	Maven JXR	
	<a href="#">Maven Ant Plugin</a>	2.0-beta-2-SNAPSHOT	<u>2</u>	Maven Plugins	



Build #	Start Time	End Time		State	
1	16.07.2006 13:13:17		Started since : 5 sec		<a href="#">Result</a>
	16.07.2006 12:41:44	16.07.2006 12:42:10	Duration : 26 sec		<a href="#">Result</a>
	16.07.2006 12:39:45	16.07.2006 12:39:47	Duration : 2 sec		<a href="#">Result</a>
	16.07.2006 12:39:42	16.07.2006 12:39:45	Duration : 3 sec		<a href="#">Result</a>
	16.07.2006 12:39:17	16.07.2006 12:39:23	Duration : 5 sec		<a href="#">Result</a>
	16.07.2006 12:28:55	16.07.2006 12:28:57	Duration : 2 sec		<a href="#">Result</a>
	16.07.2006 12:18:33	16.07.2006 12:18:35	Duration : 2 sec		<a href="#">Result</a>
	16.07.2006 12:17:44	16.07.2006 12:17:47	Duration : 2 sec		<a href="#">Result</a>
	16.07.2006 12:00:04	16.07.2006 12:00:06	Duration : 1 sec		<a href="#">Result</a>
	16.07.2006 11:52:44	16.07.2006 11:52:46	Duration : 2 sec		<a href="#">Result</a>
	16.07.2006 11:48:54	16.07.2006 11:48:57	Duration : 2 sec		<a href="#">Result</a>
	16.07.2006 11:48:28	16.07.2006 11:48:31	Duration : 2 sec		<a href="#">Result</a>

#### ➔ Add Schedule

<b>Name</b>	<input type="text" value="CONTINUOUS"/> <p>Enter the name of the schedule</p>
<b>Description</b>	<input type="text" value="Check for changes every two minutes"/> <p>Enter a description of the schedule</p>
<b>Cron Expression</b>	<input type="text" value="0/5 * * * * ?"/> <p>Enter the cron expression. Format is described there : <a href="#">Syntax</a></p>
<b>Quiet Period (seconds)</b>	<input type="text" value="30"/> <p>Enter a quiet period period for this schedule</p>
<b>Enabled</b>	<input checked="" type="checkbox"/> <p>Enable/Disable the schedule</p>

# Obsah

---

- ▶ Integrované vývojové prostredia
- ▶ Manažment verzií (zdrojového kódu)
- ▶ Zostavovanie softvéru, kontinuálna integrácia
- ▶ **Manažment zmien**
- ▶ Podpora komunikácie v tíme
- ▶ Odporúčania

# Manažment zmien/požiadaviek

---

- ▶ Zmena – vylepšenie, nová funkcionálnosť, chyba, ...
- ▶ Sledovateľnosť a dokumentácia životného cyklu
  - ▶ Prečo požiadavka vznikla
  - ▶ Koho/čoho sa týka
  - ▶ Aké má/bude mať dopady
- ▶ Ako požiadavku na zmenu vyhodnotíme
  - ▶ Budeme ju riešiť (prioritne)
  - ▶ Necháme ju na neskôr
  - ▶ Vlastne ju nepotrebuje

# Jednoduchý životný cyklus požiadavky

## Identifikácia požiadavky

- Nová požiadavka
- Nový problém
- Zmena v prostredí

## Analýza požiadavky

- Posúdenie technickej realizovateľnosti
- Určenie nákladov a prínosov

## Plánovanie zmeny

- Analýza dopadov zmeny
- Úprava plánu

## Implementácia zmeny

- Implementácia zmeny
- Testovanie zmeny
- Aktualizácia dokumentácie

## Posúdenie zmeny

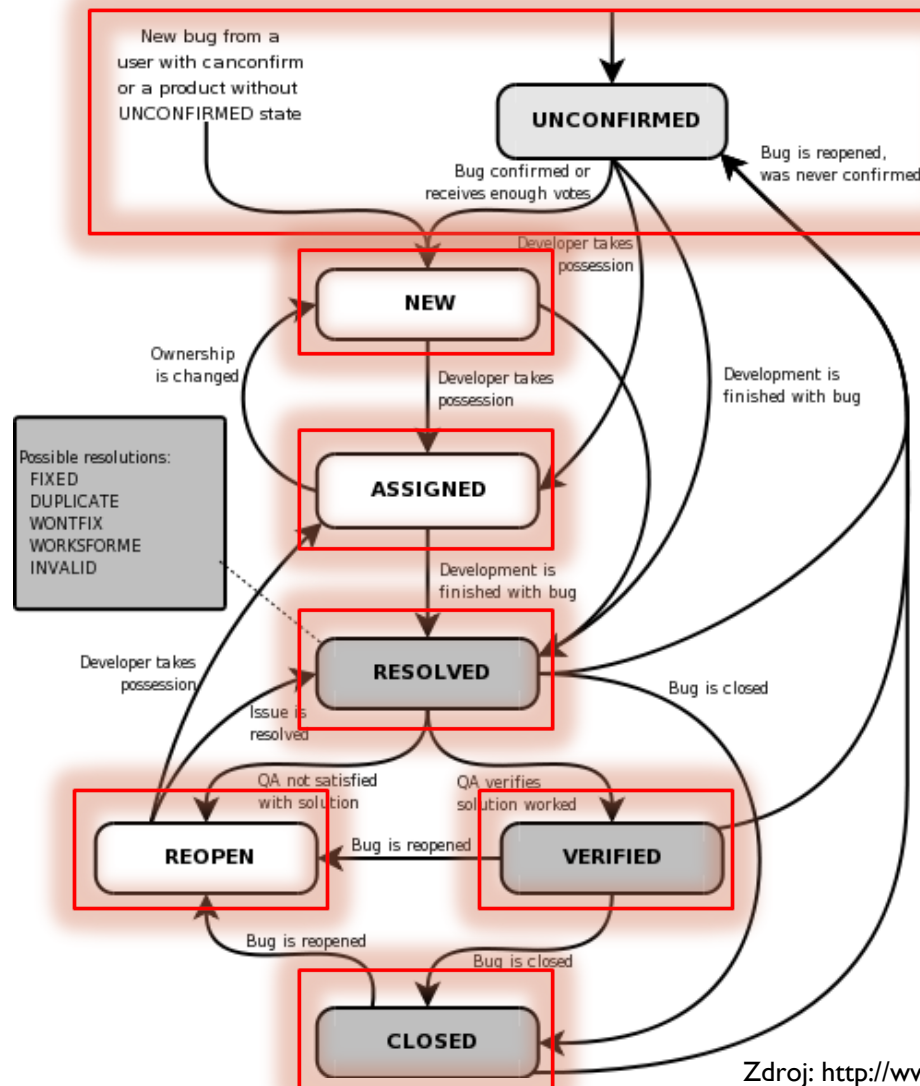
- Overenie zmeny
- Uzavretie zmeny

# Nástroje pre manažment zmien

---

- ▶ Centrálna databáza zmien
- ▶ Široké možnosti vyhľadávania podľa rôznych kritérií (priorita, typ, dátum, „mne priradené“, „mnou vložené“)
- ▶ Voľne dostupné nástroje
  - ▶ **Bugzilla** (<http://www.bugzilla.org>)
  - ▶ **Scarab** (<http://scarab.tigris.org>)
  - ▶ **Trac** (<http://trac.edgewall.org>)
  - ▶ **Atlassian JIRA** (<http://jira.atlassian.com>)  
pre projekty s otvoreným kódom používanie zadarmo

# Bugzilla - životný cyklus chyby



Zdroj: <http://www.bugzilla.org/docs/3.2/en/html/lifecycle.html#lifecycle-image>

# Obsah

---

- ▶ Integrované vývojové prostredia
- ▶ Manažment verzií (zdrojového kódu)
- ▶ Zostavovanie softvéru, kontinuálna integrácia
- ▶ Manažment zmien
- ▶ **Podpora komunikácie v tíme**
- ▶ Odporúčania

# Komunikácia v tíme

---

- ▶ Kvalita komunikácie → kvalita práce
  - ▶ Kontakt v tíme, synchronizácia, riešenie problémov
  - ▶ Možnosť riadenia, podpory práce, synergia
- ▶ Dôležitosť rastie s veľkosťou tímu, zásadný význam v distribuovaných tímoch (ale aj teleworking)
- ▶ Rôzne formy:
  - ▶ Emaily, stretnutia, IM (Instant Messaging), diskusné skupiny, ...



# Emaily v tíme

---

- ▶ TO:           zvážiť, pridáva prácu prijímateľovi
  - ▶ Udržovať zoznam adries, ReplyToAll
- ▶ CC:           na info vedúcemu, iným zainteresovaným
- ▶ BCC:          „tajne“ na info tomu, čo by to mal vedieť
- ▶ Priorita:     aktívne používať (HIGH, LOW)
- ▶ Subject:     stručne, vecne, skratky
- ▶ Vždy uviesť
  - ▶ Kontext mailu
  - ▶ Obsah
  - ▶ **Zhrnutie a vyznačenie toho, čo očakávame**

# Zdieľaný kalendár

---

- ▶ Poskytne prehľad o zdrojoch – kto, kedy má čas
- ▶ Umožňuje efektívnejšie plánovanie
  - ▶ Priradenie (spoločných) úloh a zodpovedností
  - ▶ Zohľadnenie časových možností
  - ▶ Plánovanie stretnutí
  - ▶ Preplánovanie existujúcich činností

# Online komunikácia

---

- ▶ Priama synchrónna komunikácia
  - ▶ Keď treba rýchlo niečo vybaviť
- ▶ Chat - posielanie správ cez IM
- ▶ Tele-/video-konferencie napr. cez Skype
- ▶ Pozor na

**Zahltenie a neustále vyrušovanie pri práci (!)**

# Obsah

---

- ▶ Integrované vývojové prostredia
- ▶ Manažment verzií (zdrojového kódu)
- ▶ Zostavovanie softvéru, kontinuálna integrácia
- ▶ Manažment zmien
- ▶ Podpora komunikácie v tíme
- ▶ **Odporúčania**

# Čo určite použiť

---

- ▶ IDE podľa použitého jazyka [Eclipse, VS, ...]
- ▶ Systém pre manažment verzií [SVN, ...]
- ▶ Automatické testovanie jednotiek [J/Nunit]
- ▶ Komunikačné nástroje [IM, email, wiki, TRK...]
- ▶ Generovanie dokumentácie [Java/Ndoc, Sandcastle, ...]

# Čo ďalšie možno použiť

---

- ▶ Riadenie zmien/chýb [Bugzilla, ...]
- ▶ Analyzátor kódu [Checkstyle]
- ▶ Automatické zostavovanie projektu [Ant, Maven, ...]
- ▶ Kontinuálnu integráciu a zostavovanie [CC/CC.net...]
- ▶ Profilovacie nástroje