



Tools

- Continuous Integration
- CVS
- Communication

The logo for ARENA, featuring the word "ARENA" in a bold, metallic, 3D-style font. The letters are dark with a lighter, reflective top surface, giving them a three-dimensional appearance. The background behind the text is a textured, golden-brown surface with concentric, wavy lines, resembling sand or a wood grain.

ARENA Continuous Integration I

- Benefits
 - Bugs appear earlier in the SE process
 - Faster to track the bug
 - Saves lots of time when it comes to integrating the whole product
 - Does NOT catch all integration bugs
- The More Often The Better
 - Build at least once a day/night
 - Example: Microsoft does nightly builds for projects with >30 mill LOC
 - Absolutely necessary to automate build process

The logo for ARENA, featuring the word "ARENA" in a bold, metallic, 3D-style font. The letters are dark with a lighter, reflective top surface, giving them a three-dimensional appearance. The background behind the text is a textured, golden-brown surface with concentric, wavy lines, resembling a cross-section of wood or a similar natural material.

ARENA Continuous Integration II

- What is a successful build?
 - All the latest sources are checked out of the configuration management system
 - Every file is compiled from scratch
 - The resulting object files (Java classes in our case) and linked and deployed for execution (put into jars).
 - The system is started and suite of tests is run against the system.
 - If all of these steps execute without error or human intervention and every test passes, then we have a successful build

The logo for ARENA, featuring the word "ARENA" in a bold, metallic, 3D-style font. The letters are dark with a lighter, reflective top surface, giving them a three-dimensional appearance. The background behind the letters is a textured, golden-brown surface that looks like wood grain or a similar natural material.

ARENA Continuous Integration III

- Single Source Point
 - That's why we use CVS
- Automated Build Scripts
 - With lots of files, it's not going to be possible to compile everything by hand
 - We are going to use ANT
- Checking In
 - IMPORTANT: Check your code in early!
- Regression Testing




CVS - Introduction

- *Concurrent Versions System*
- Enables groups of persons to work simultaneously on source code
- Central *repository* holds all versions of the source code, dividing it into *modules*
- You can *check out* code from the repository, edit it and *commit* it again
- Possible to fetch every earlier version from the repository
- Detects and marks *conflicts*



CVS enviroment variables

- **CVSROOT**
(setenv CVSROOT :ext:<login>@cvsbruegge.in.tum.de:/cvs/arena)
- **CVSEEDITOR**
(setenv CVSEEDITOR emacs)
specify the editor for comments (default is vi)
- **CVS_RSH**
must point to ssh
(setenv CVS_RSH ssh)

The logo for ARENA, featuring the word "ARENA" in a bold, metallic, 3D-style font. The letters are dark with a lighter, reflective top surface, giving them a heavy, industrial appearance. The background behind the text is a textured, golden-brown surface that looks like wood grain or a similar natural material.

ARENA CVS – Basic commands

- `cv`s checkout <module> (**cv**s co)
 - Creates a local copy of the repository
- `cv`s update
 - Updates you local copy from the repository
- `cv`s commit
 - Checks in your changed files
- `cv`s add `cv`s remove
 - Adds or removes files from the repository
- `cv`s log
 - Shows the comments made by the persons who committed it
- Many many more ... `man cv`s is your friend



CVS checkout

- To get a copy of the repository you must use the following command:

```
cvs checkout [-D <Date>] <module>
```

(example: `cvs checkout -D yesterday .`)



CVS add

- Get a working copy of the repository
`cvsv checkout <module>` (e.g. `cvsv checkout .`)
- Create the new file
- Checkin the new file
`cvsv add [-m <description>] <filename>`
- `cvsv commit [-m „Early version“] <filename>`



CVS remove

- Get a working copy of the repository
cvs checkout <module>
- Delete the file(s)
- Call: cvs remove
to mark the files for removing
- Call: cvs commit
to remove the files from the repository



CVS rename

- Get a working copy of the repository
cvs checkout .
- Rename the file (mv <oldfile> <newfile>)
- Call: cvs remove <oldfile>
- Call: cvs add <newfile>
- Call: cvs commit <oldfile> <newfile>