

Software Configuration Management



Lecture: Build Management

René Krikhaar

Lecture Objectives



- ◆ Learn the basics of Build Management
- ◆ Being able to understand and analyze Build Management in Industry

Build Management



- ◆ Support building a software system
 - ◆ Compile & Link source files
 - ◆ Generate User Interfaces (from High Level Specs)
 - ◆ Generate Documentation
 - ◆ ...

Basic Notions of Building



- ◆ Derived Object: object as derived from other source(s) or derived objects
- ◆ Build plan: Set of build rules
- ◆ Build rule: Description of an action to produce derived object(s)
- ◆ Build step: Execution of an action (according to some rule)
- ◆ Build record: Trace of an actual build

Build Tools



- ◆ Make: A program to maintaining Computer Programs (S.I. Feldman)
 - ◆ Dependencies between files
 - ◆ Based on Changes of Files (**Time Stamps**)
 - ◆ Synchronized time on build engines
 - ◆ Poor integration with SCM tools
 - ◆ Re-build changed files and files that depend on other changed files
 - ◆ *Makemake* tools exists to create Make scripts
- ◆ Visual Studio Project/Solution Files (Microsoft)
- ◆ ANT and NANT (Java + .Net extension)

Simple Example of Make



Main.exe: main.o a.o b.o

cc main.o a.o b.o -o main.exe

Main.o: main.c main.h a.h b.h

cc -c main.c

a.o: a.c a.h

cc -c a.c

b.o: b.c b.h

cc -c b.c

Variations in Build



- ◆ Building various “targets” / “configurations”
 - ◆ Test version (-Dtest)
 - ◆ Debug version (-debug)
 - ◆ Machine version (WXP, W2000)
- ◆ Being able to build parts of the system
 - ◆ Certain Components only
 - ◆ Configuration Selection (versions to be build)

Configuration Selection

- ◆ Configuration Selection:
selection of a set of CI versions

- ◆ Build Management should be aware of *configuration selection*

| Item / Version | Release 2.3 | Stable | Latest |
|----------------|-------------|---------|--------|
| Product | 2.6.1.1 | 3.0 | 3.1 |
| Item A | 1.2 | 1.2 | 1.3 |
| Item B | 1.1 | 1.1 | 1.1 |
| Item C | 1.1 | 1.1.1.1 | 1.2 |
| Item D | 4.2 | 4.3 | 4.3 |

- ◆ *In selecting versions of source*
- ◆ *In selecting versions of derived objects*

Functionality of BM



- ◆ Build & Store derived objects
 - ◆ including build record
- ◆ Performance of Build
 - ◆ Efficient calculation of firing build rules
 - ◆ Keep Dependencies in Memory
 - ◆ Determine reuse of derived objects (stored in SCM; understand build record)
 - ◆ Determine (possible) concurrency in build plan
 - ◆ Distributed build steps (use network of build engines)

Dependencies in Build Rules



- ◆ Dependencies on Source Versions
- ◆ Dependencies on Derived Objects
- ◆ Dependencies on Libraries
- ◆ Dependencies on Tool (versions)
 - ◆ Compilers, Linkers, DocGen, ...

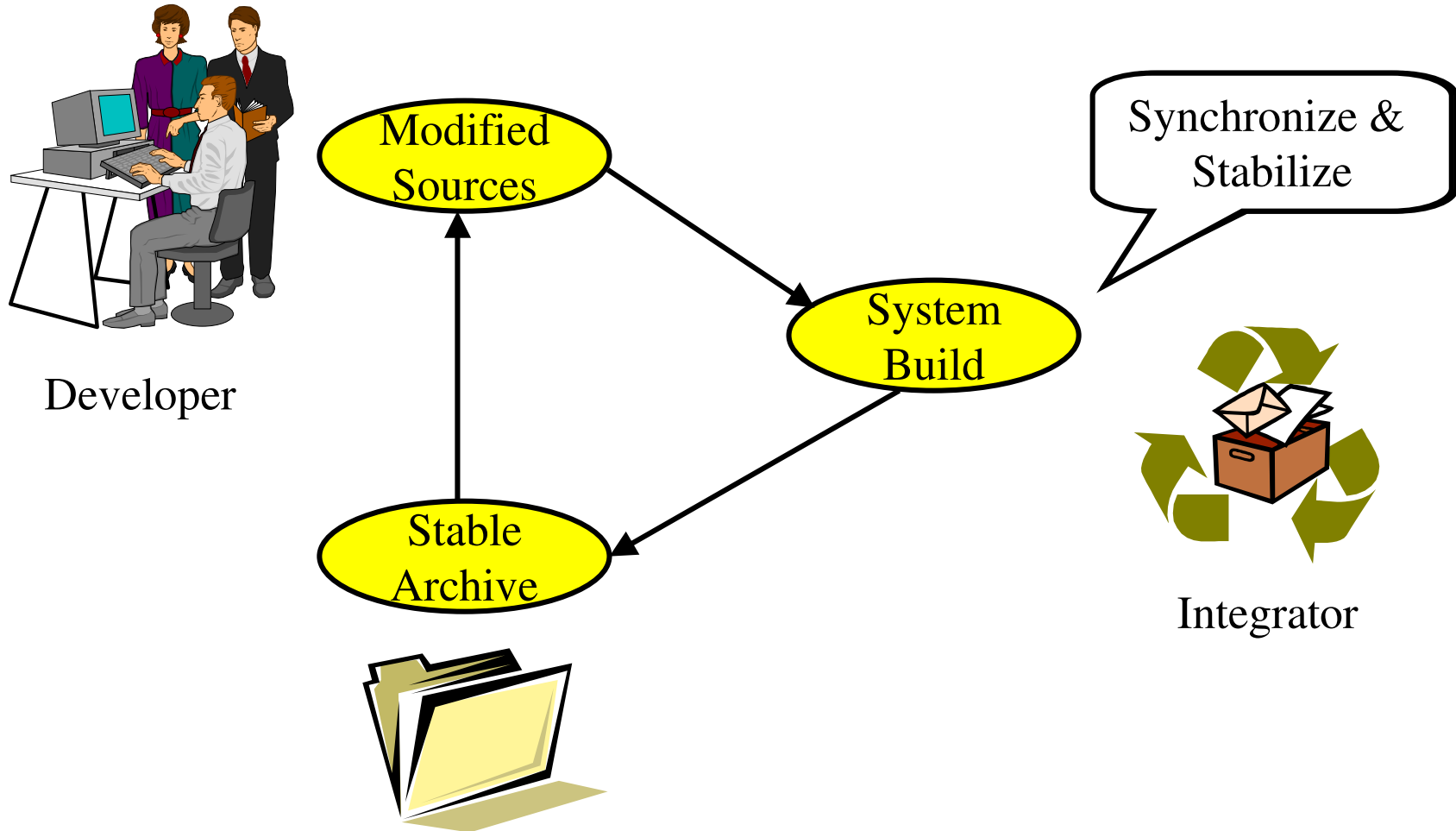
- ◆ Build must rely on correct dependency information

Build in Development Process



- ◆ Daily (System) Build
 - ◆ Build Complete System in a controlled way
 - ◆ Incorporate “Stable” versions of developers
- ◆ Good principles
 - ◆ Daily build repair has highest priority
 - ◆ Build repairs are resolved by “creator”
 - ◆ Build manager owns the build plans

Daily Build and Smoke Test



Related Issues



- ◆ Automatic Regression Tests implemented by extra Build Rules (as part of Build Plan)
- ◆ Which targets should be build?
 - ◆ E.g. with “Debug” compiled sources
 - ◆ E.g. for vxworks and WXP Operating System
- ◆ Storing Build Numbers in End Product

Conclusions



- ◆ Build Management provides build plans for different stakeholders
 - ◆ Developers
 - ◆ Maintenance Engineers
 - ◆ System builders
 - ◆ Testers
- ◆ Build Management must be aware of Version Control system