Introduction to Software Configuration Management

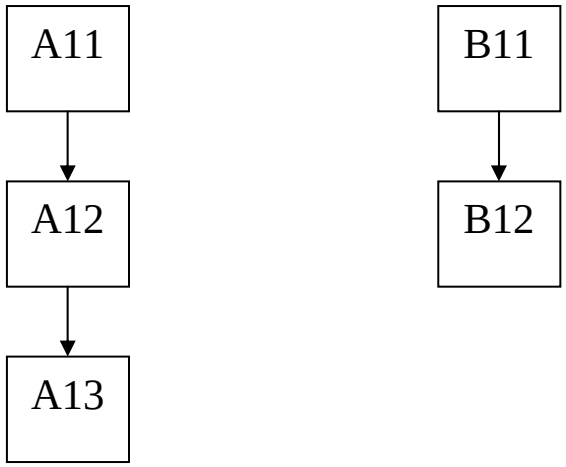
CprE 556

Electrical and Computer Engineering Department

Iowa State University

Example

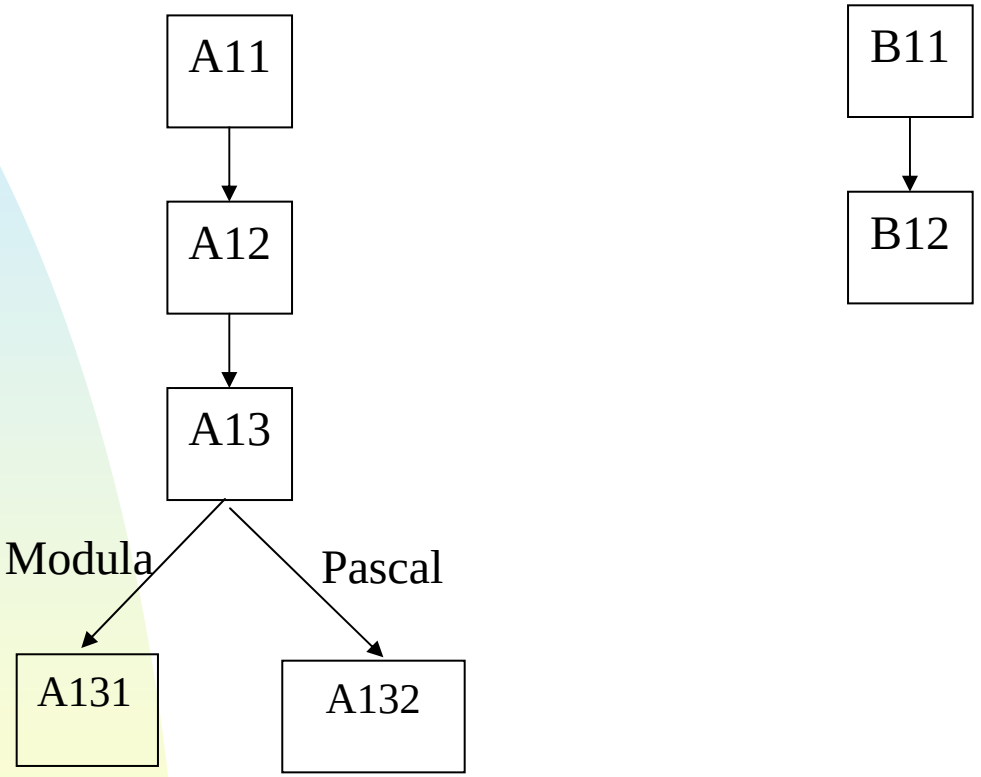
- Initially, implementation is in Modula-2 on a Mac.



Example

- Next: a variant for Pascal is needed. Component B does not differ for the Pascal and Modula-2. But component A needs a variant.

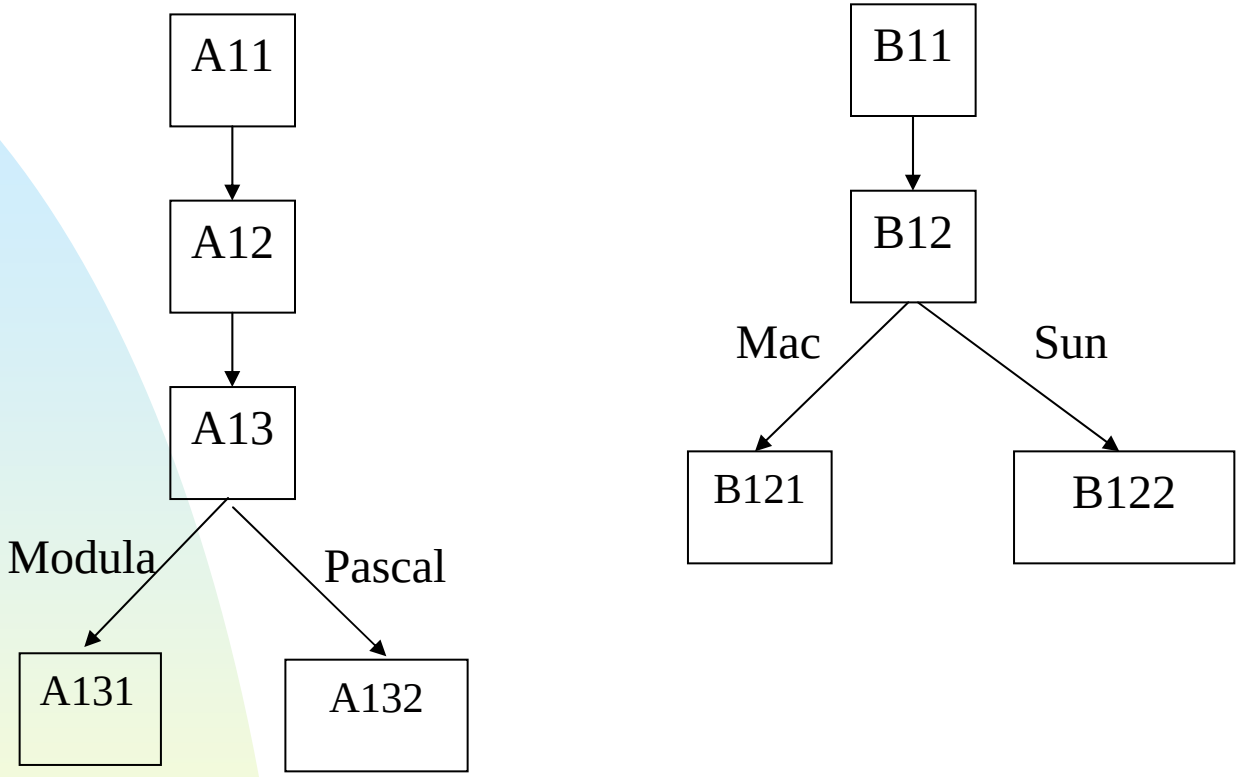
Example



Example

- Next: a variant of component B is needed for Sun and Mac implementations, where component A is believed identical for Mac and Sun.

Example

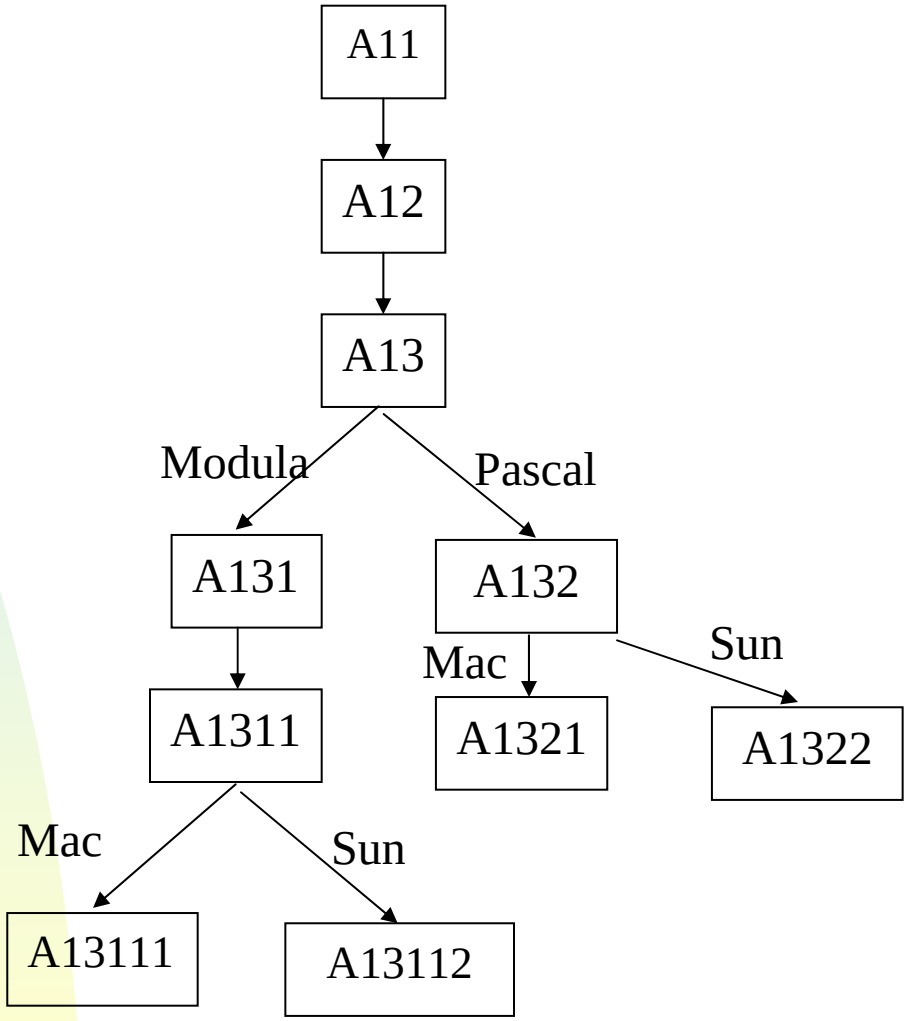


+ In Modula for Mac, A131 and B121
+ In Pascal for Mac, A132 and B121
+ In Modula for Sun, A131 and B122

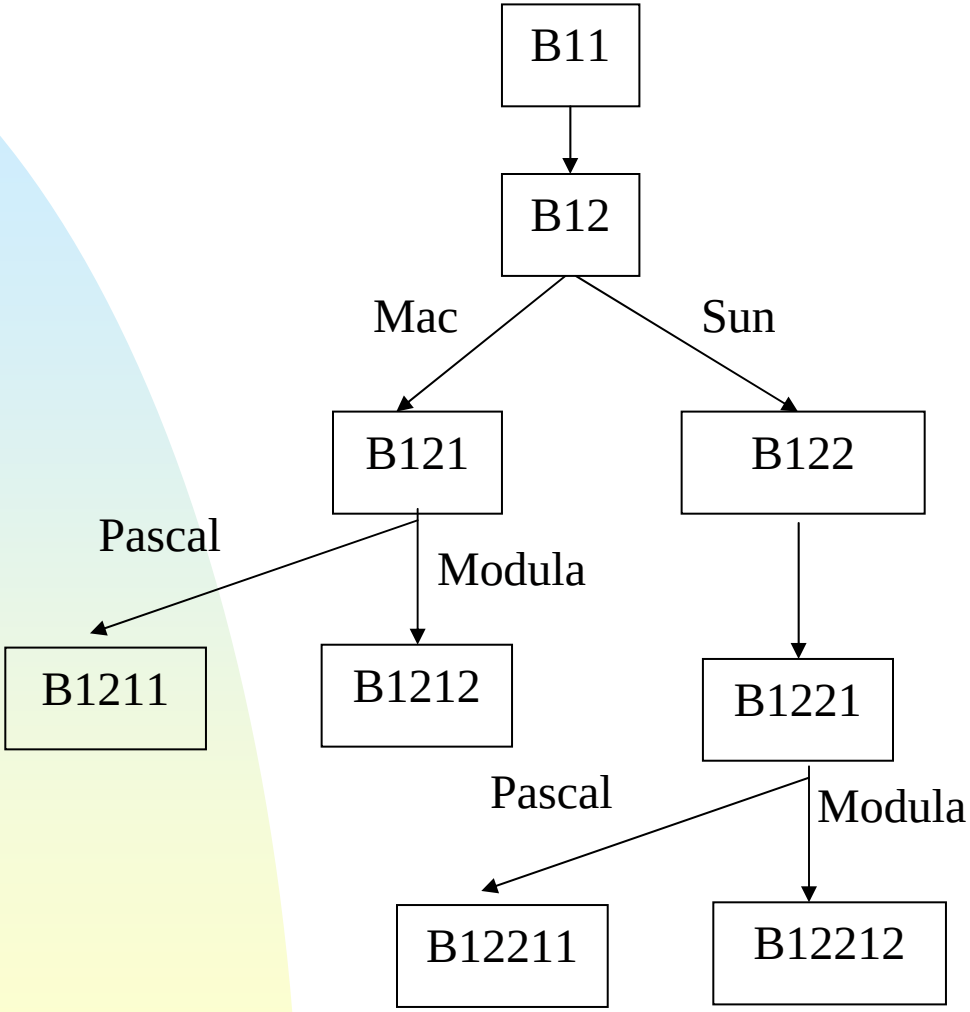
Example

- After development of some revisions, it turns out that component A must be implemented differently on both machines, and component B has to be developed in two variants according to the programming languages.

Component A


A11
A12
A13
Modula
Pascal
A131
A132
Mac
Sun
A1311
A1321
A1322
Mac
Sun
A13111
A13112

Component B


B11
B12
Mac
Sun
B121
B122
Pascal
Modula
B1211
B1212
B1221
Pascal
Modula
B12211
B12212

Important concepts

- **Configuration management** or **configuration control**
  - The management of a system through *control of changes* made to hardware, software, firmware, documentation, test, test fixtures and test documentation of an *automated* information system, throughout the development and operational life of a system.

Important concepts

- **Software configuration management** concerns the storage of the entities produced during the software development project.
- The control of changes--including the recording thereof--that are made to the software and documentation throughout the software system lifecycle.
- The control and adaptation of the evolution of complex software systems

Software Configuration Management

- **SCM** is the discipline of organizing and managing the evolution of large and complex software systems [Conradi and Westfechtel]
- SCM consists activities to control change by identifying the products that are likely to change, defining mechanisms for managing different versions of these products, controlling the changes imposed, establishing relationships among them, and auditing and reporting on the changes made [Pressman]

Software Configuration Management

- SCM is a methodology to control and manage a software development project.
- SCM concerns itself with answering the question: *somebody did something*, *how can one reproduce it?*
- Often the problem involves not only reproducing *identically*, but also with controlled, incremental changes.
- Answering the question will thus become a matter of comparing different results and of analyzing their differences.

Software Configuration Management

- Traditional *SCM* typically focused on controlled *creation* of relatively simple products with only source code.
- Source configuration management
- Nowadays, vendors of SCM face the challenge of dealing with relatively minor increments under their own control, in the context of the complex system being developed.

General goals

- to maximize productivity by automating tasks
- minimizing mistakes
- maintaining software integrity, traceability, and accountability

SCM and Capability Maturity Model

- Capability Maturity Model (CMM) defines levels of maturity in order to access software development processes in organizations.

- SCM is seen as a key element from “initial” (undefined process) to “repeatable” (project management, SCM, and quality assurance).

SCM in different angles

- SCM is seen in different angles
- SCM: as a development support discipline
  - SCM provides functions that assist developers in performing coordinated changes to software products and components
- SCM: as a management support discipline
  - SCM is concerned with controlling changes (modifications) to software products (and their software artifacts)

SCM: As a development support discipline

- Functionalities
  - SCM provides functions that assist developers in performing coordinated changes to software products and components
  - accurately recording the composition of versioned software products evolving into many revisions and variants
  - Reconstructing previously recorded software component versions and configurations

SCM: As a development support discipline (2)

- Maintaining consistency between interdependent components
- Building derived objects
- Constructing new configurations based on descriptions of their properties

SCM: as management support discipline

- SCM covers functionalities
  - identifications of product components,
  - change control (establishing procedures to be followed when performing a change),
  - status accounting (recording and reporting the status of components and change requests, etc)
  - audit and review (quality assurance)

Specific goals

- Configuration Identification - What code are we working with?
- Configuration Control - Controlling the release of a product and its changes.
- Status Accounting - Recording and reporting the status of components.
- Review - Ensuring completeness and consistency among components.

Specific goals

- Build Management - Managing the process and tools used for builds.
- Process Management - Ensuring adherence to the organizations development process.
- Environment Management - Managing the software and hardware that host our system.
- Teamwork - Facilitate team interactions related to the process.

Version model

- **Revision control** (also known as **version control**) is the management of multiple revisions of the same unit of information.
- A core component of a SCM system
- A version model defines
  - The objects to be versioned
  - Version identification and organization
  - Operations for retrieving existing versions and constructing new versions

Multiple versions

- As software is developed and deployed, it is extremely common for multiple versions of the same software exist.
  - Multiple developers
  - Parallel development paths
    - Bug fixing, new functionality
  - At the simplest level, developers can simply retain multiple copies of the different versions of the program.

Version control software

- Free software or open source
  - Older, not widely used any more: SCCS, RCS
  - Software using a non-distributed approach: CVS, Subversion, Stellation for Eclipse, Vesta, Superversion
  - Software using a distributed approach: GNU Arch, Bazaar, ArX

SCM software

- ClearCase IBM/Rational
- Visual SourceSafe, Microsoft
- Perforce
- Serena Version Manager — previously Merant PVCS
- TrueChange
- BitKeeper (was used in Linux kernel development December 1999 - April 2005)
- Aldon Lifecycle Manager
- More on CM Crossroads: www.cmcrossroads.com

Version control tools

- It is most commonly used in engineering and software development
  - to manage ongoing evolution of digital documents like application source code, art resources such as blueprints or electronic models and other critical information that may be worked on by a team of people.
- In practice, tools for revision control have rarely been used outside software development cycle (though they could actually be of benefit in many other areas)

Aspects

- Variant management
- Differences and differencing
- Software merging
- (Physical or virtual) workspace management
- Awareness
- User-interfaces
- Build and release management

SCM in other contexts

- Versioning for hypertext systems
- SCM and content change management for Web-based systems
  - TeamSite, StoryServer, DynaBase, WebDAV, ChangeMan
- Product data management and SCM
  - CAD
- Temporal databases
- Change impact management
- architectural configuration management

Configuration items

- **A configuration item** is a unit of configuration that can be individually managed and versioned.
  - An aggregation of hardware or software or both that is designated for configuration management and treated as a single entity in the configuration management process.
  - A work product that is placed under configuration management and treated as a single entity.

Configuration items

- A configuration item (a component) : files, directories, objects in OO database, entities, relationships and attributes in EER databases,… [Conradi]
- Any software products produced during a software lifecycle.
- SCM: “umbrella activity” [Pressman]

A configuration

- The units themselves can be considered configuration items, or they may be combined into a bigger collection that is also managed.
- A *configuration*: a version of a complex object, which is composed of versions of components [Conradi].
- A configuration of a software project is composed of versions of the requirements definition, the software architecture, the program source code, etc.

Baseline

- A specification or product that has been formally reviewed and agreed upon, thereafter serves as the basis for further development, and can be changed only through formal change control procedures.
- A document or a set of such documents formally designated and fixed at a specific time during the lifecycle of a configuration item.

Versions, revisions and variants

- Version: An instance of a configuration item. Once a version is baselined it cannot be changed without creating a new version.
- *Revisions* and *variants*
  - *Revisions*: Sequential versions that evolve along the time dimension
  - *Variants*: Parallel versions coexisting at a given time

Versions, revisions and variants

- *Revision: a* version that supersedes an earlier version, typically, to correct errors as opposed to a version that is an alternative version.
- *Variant*: a version that is an alternative of another version. For example, variants allow a configuration item to meet conflicting requirements
- *Delta*: the difference between two versions
- *Release*: a configuration management action whereby a particular version of software is made available for a specific purpose.