Distributed Version of Management for Computer Software (DVMS)

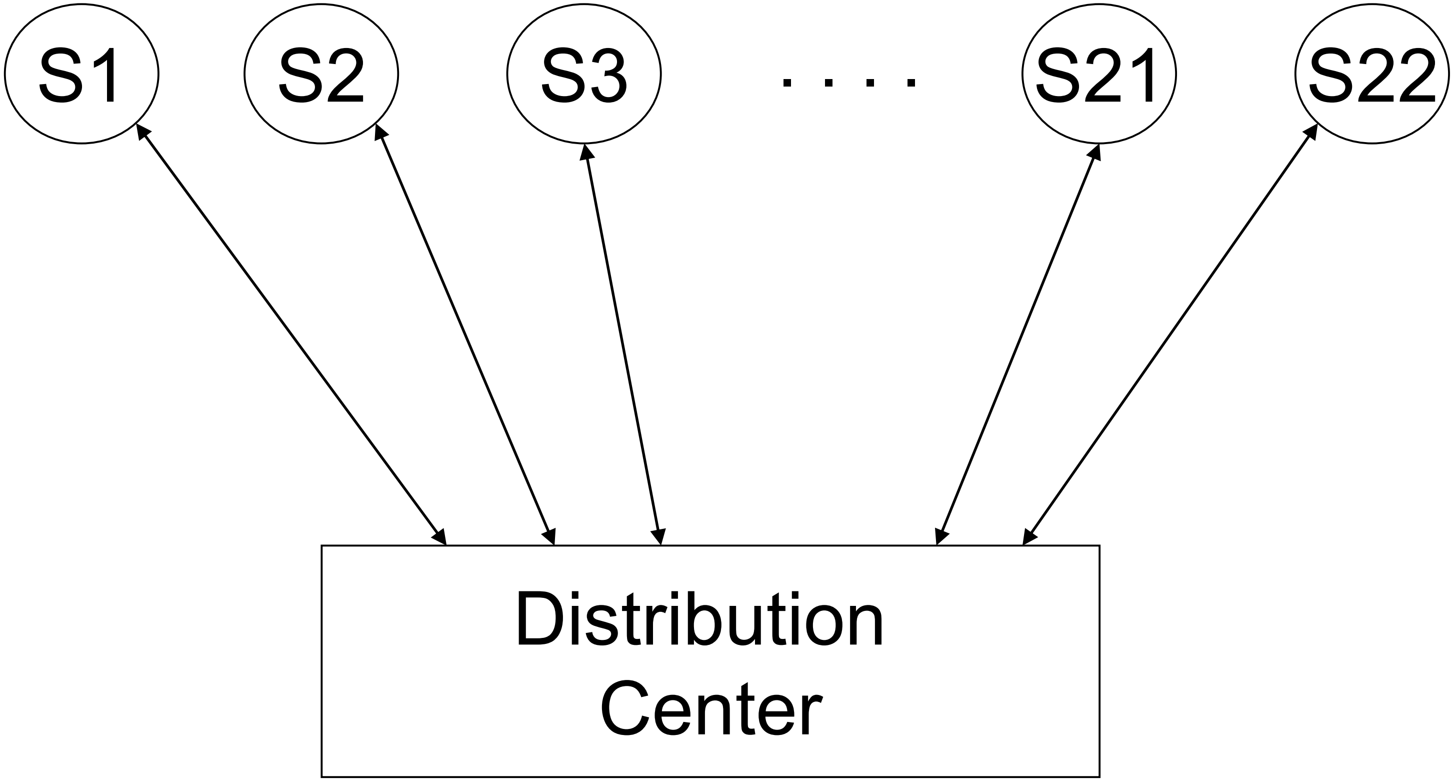
What is the Problem?

- We recently fixed this problem. Why has it reappeared?
- Who originated this modification?
- We got this version from Atlanta. I wonder if they incorporated the changes I sent out last week.
- That site has been going up and down. I hope they’ve received all the updates to the baseline.
- This release is having problems. What version of the system can we roll back to?
- What do I have to recompile to include this modification?
- etc.

Distributed Software Development

Software projects require the development of code by teams
At multiple work stations (sites). Examples:

- Two teams are changing software packages and require each other's changes.
- Maintenance of software which runs at several locations (e.g., enroute air traffic control centers) results in debugging and modifications which need to be transmitted to other centers.
- Delivered software is installed at a customer site while enhancements to the software are done at an industrial site.



Example from FAA enroute air traffic control system

Revision Control Tools

- Source code Control System (SCCS) – AT&T
- Code Management System (CMS) – DEC
- CLEAR/CASTER – IBM
- DSEE – Apollo
- Revision Control System (RCS) – Tichy (Purdue)

Characteristics

- Centralized control
- No identification of special configurations (except DSEE)
- No automatic distribution of changes

Goals

- Study the problems associated with distributed software development in industrial environments

  [Need input from industrial affiliates]

- Build prototype system for DVMS to run in conjunction with current version management techniques

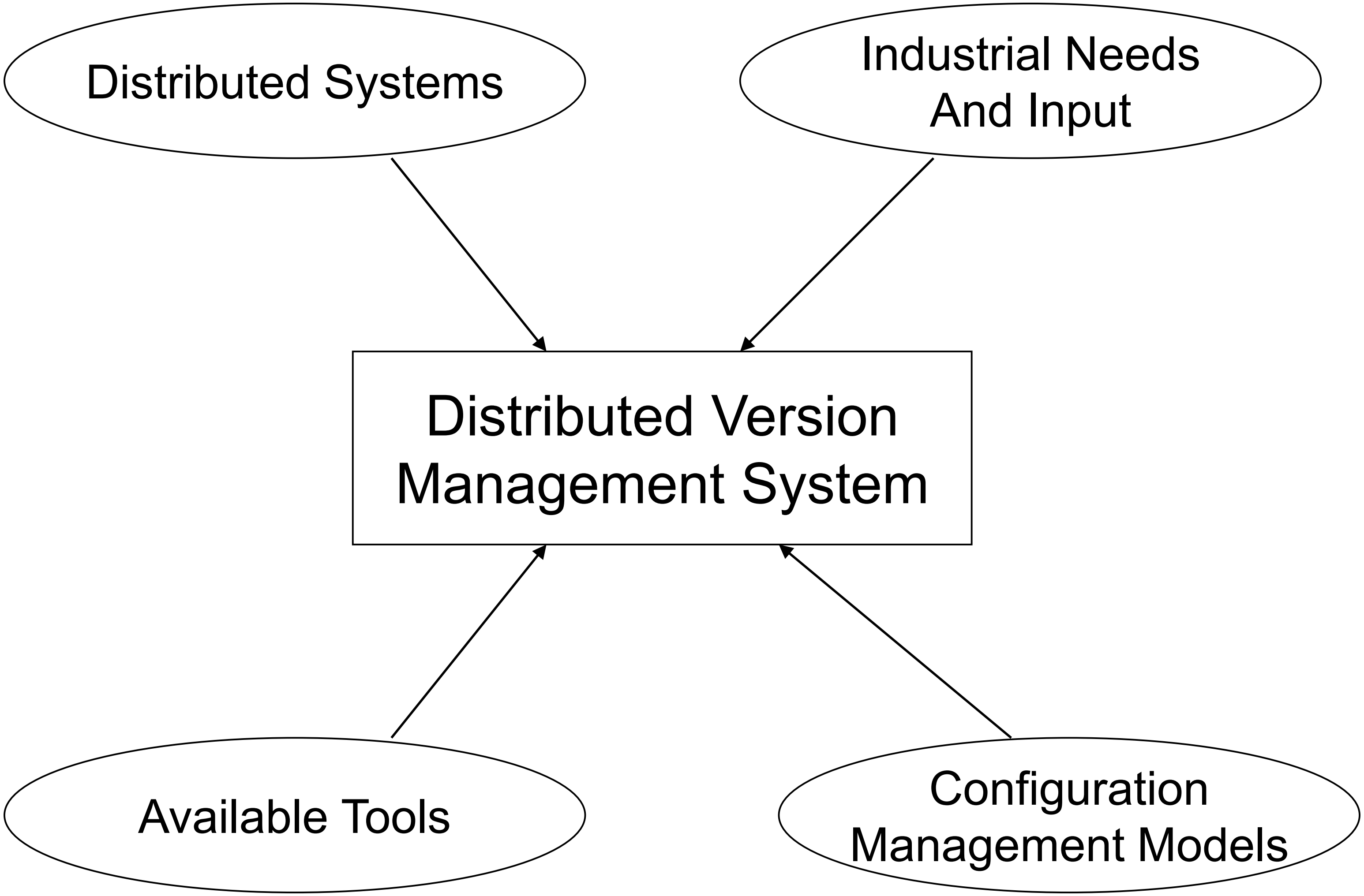
  [Need research from consistency control, version control, replication control]

- Produce and tune DVMS software package for use in an industrial setting

  [Need to work closely with affiliates]

Requirements of DVMS

- Track changes to software packages
- Manage the simultaneous updating of software packages
- Determine a set of consistent versions of software packages to run a system (version selection and baselining)
- Construct an environment from component packages which is suitable for system execution
- Allow modifications to a software package while some of its copies on other sites are unavailable (dealing with site failure and network partitioning)



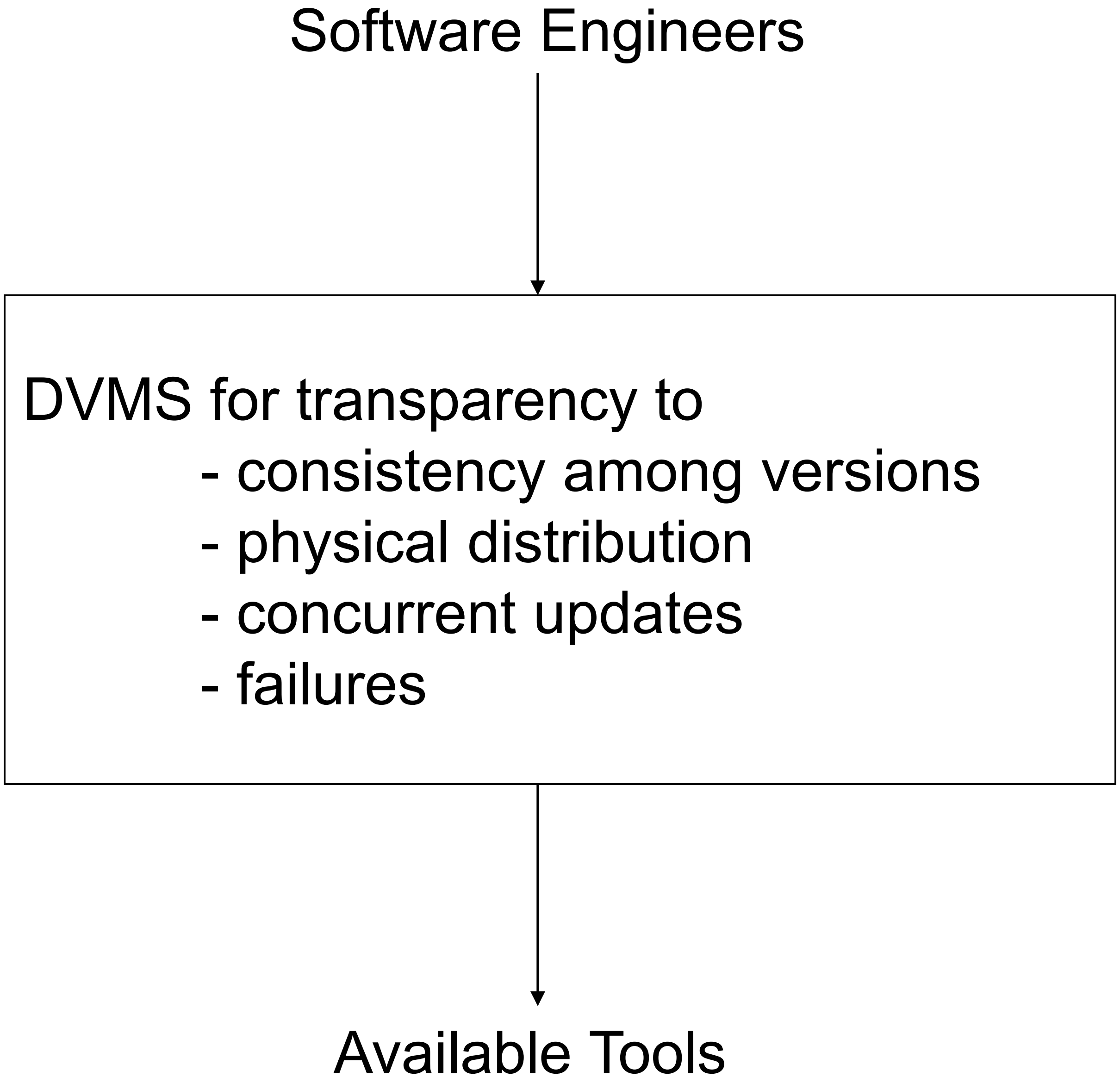
Distributed Systems

Industrial Needs
And Input

Distributed Version
Management System

Available Tools

Configuration
Management Models



Software Engineers

DVMS for transparency to

- consistency among versions
- physical distribution
- concurrent updates
- failures

Available Tools

Towards a Model of DVMS

- Objects
  - software object
  - source object
  - derived object
  - distribution
- Multiple versions
  - version groups
  - y revision of x
  - y variant of x
- Communications
  - amount required for distributed development
  - set of primitive functions

[We need input from industrial affiliates[1] to better define the parameters of a model]

[1] Have begin interactions with Dr. Adnan Onart of Racal-Milgo Inc., and Dr. Phil Koltun of Harris Corporation

Implementing DVMS Software Package

- Define a communications library
- Use delta storage for conserving space
- Use smart recompilation techniques to make configuration processing more efficient
- Provide the ability to retrieve older versions of software
- Include the concept of revision control to track changes to source objects

[Further market study and investigation of available tools is Planned]