Software Maintenance and Evolution: Background

• **SW Maintenance** - “the modification of a software product after delivery to correct faults (corrective), to improve performance or other attributes (perfective), or to adapt the product to a modified environment (adaptive)”. Other types of modifications are made in anticipation of future faults (preventative).

• **SW Evolution** – lacks formal definition, but is held to be a more informed version of maintenance, where major changes to software products are made in a way to preserve original constructs and meaning

  • Started as a natural occurrence of the complexity of software development (debugging and enhancements are needed), presumably in the 1950’s

  • Lientz and Swanson project estimated of the amount of time and money spent in maintenance phases of development and classify types of maintenance as the four is parentheses in the definition above in 1980.

• **The Problem**: vast amounts of resources spent on keeping software working after initial development and making it better

Software Maintenance and Evolution: Today

• Today we are using staged models of the software maintenance cycle to help predict the amount of resources to be spent on “normal” types of maintenance activities. There are tools that help to isolate changes and enhance information about the systems to be changed, mostly reverse engineering tools. Software process can effect the maintenance phase by the amount of preserved documentation left from initial development. The big issue (still) is changing software easily, quickly, and reliably.

• Most of the ongoing research today is on the following topics:
  • Creating software and software architectures that are more easily adaptable after initial release
    • Robust architectures for change
  • Preserving or recovering design information and rationale for use by the maintenance team
    • See reverse engineering
  • Ways of making changes have less impact on the overall system
    • Separation of concerns (Hyper/J, Aspect/J)