What is Eclipse?

• **Eclipse is:**
  
an open source community whose projects are focused on providing a vendor-neutral open development platform and application frameworks for building software.

• **The Eclipse Foundation is:**
  
a not-for-profit corporation formed to advance the creation, evolution, promotion, and support of the Eclipse Platform and to cultivate both an open source community and an ecosystem of complementary products, capabilities, and services.

• **Variety (!) of big-name players:**
  
  – IBM, Intel, Borland, CA, BEA, Sybase, Nokia, Wind River, HP, SAP
Best known face of Eclipse: IDE

- Code editor
- PyLint
- Module content
- CVS tree
- Interactive prompt
- __doc__ string
Looks good ... usable?

• Need recent hardware (OS/JVM dep.)
  – Ongoing dev. to improve performance
  – Eclipse itself is nicely responsive
    • Editors, wizards, code-completion, etc.
  – Code compilation slower than shell
  – Memory hungry (as is Atlas Software)
    • E.g. can't index/build ROOT on a 4GB box

• Redundancy: yet another Java ...
  – GUI toolkit (SWT)
  – Component model (Equinox)
Atlas problems to solve

- **Software life cycle**
  - Not part of average physicist's toolkit
  - Neither intuitive, nor explorable

- **Tool integration**
  - Multi-language environment
  - Individual tool interactions
  - Setup/environment difficulties

- **Beginners sandbox**
  - More sophisticated "undo" than "rm -rf"
IDE is just one application ...
Middleware

• Not a GUI, but an *application platform*
  – Tools integration
    • Multi-language, -platform, and -vendor
    • Adaptation, distribution (install/updates)
• Component model implements OSGi
  – [http://osgi.org](http://osgi.org) (version 4)
  – Plugins/Bundles w/ life- and runtime mgmt
    • Only connect through extension points
    • Each their own “execution space” (i.e. class loader and process environment)
• Complexity solvable by modular sw
  – Only possible if dependencies managable
    • 100% automatic <=> 100% strict (no backdoors)
    • Limited human input ok, if always verified
  – Shared libs, .class files != modular
    • More like a grab bag: access is not controlled

• Lifetime control starts at installation
  – Eclipse: from Eclipse startup

• Connection control through broker
  – Not (yet) implemented in Eclipse
Tools integration

- Languages
  - Java
  - Python
  - GCC

- Eclipse Platform
  - ASK plugin
  - JEP
  - .project
  - .cdtproject

- External Tools
  - Pacman plugin
  - Atlantis plugin

- Workflow
  - Tutorials
  - How-to's

- CVS

- User Code
  - Athena

- CMT

- Office of Science
Athena Startup Kit (ASK)

- **User-space set of tools**
  - Fills voids left by other applications

- **Code generation for Atlas Framework**
  - Algorithms, AlgTools, DataObjects

- **RunTime management**
  - Site abstractions (CERN, BNL, local, etc.)
  - CMT requirements, config, setup
  - POOL file catalog maintainance

- **Module, fully scriptable, CLI, GUI**
End-user Workflow

Code creation, build, run, and check-in
Wizard for new package

New CMT Package
This wizard creates a new Atlas-style cmt package.

Name: MyPackage
Release: 11.0.4
Version: 00-00-01
Location: /home/wlav/Eclipse/target/workspace/Example

Next > Finish Cancel
Build with external tool
Add to repository
Conclusions and Outlook

• Eclipse has lots of potential uses
  – Impressive, full-featured IDE
  – Individual tool execution environments
  – Allows interactive how-to's

• Plugin development is easy and clean
  – Rewarding for post-doc/student

• Maybe too top-heavy (esp. GUI)
  – Atlas sw itself has high requirements