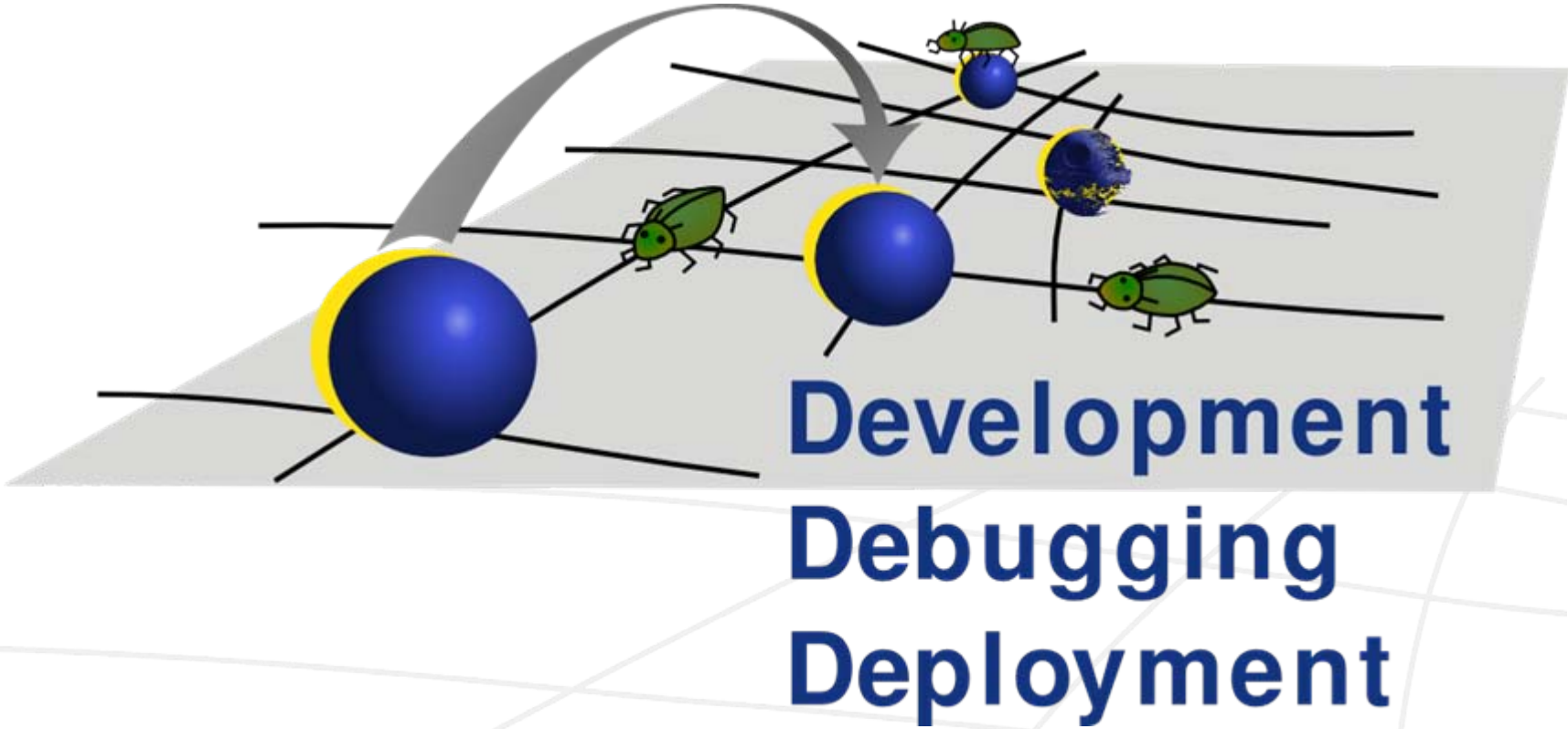


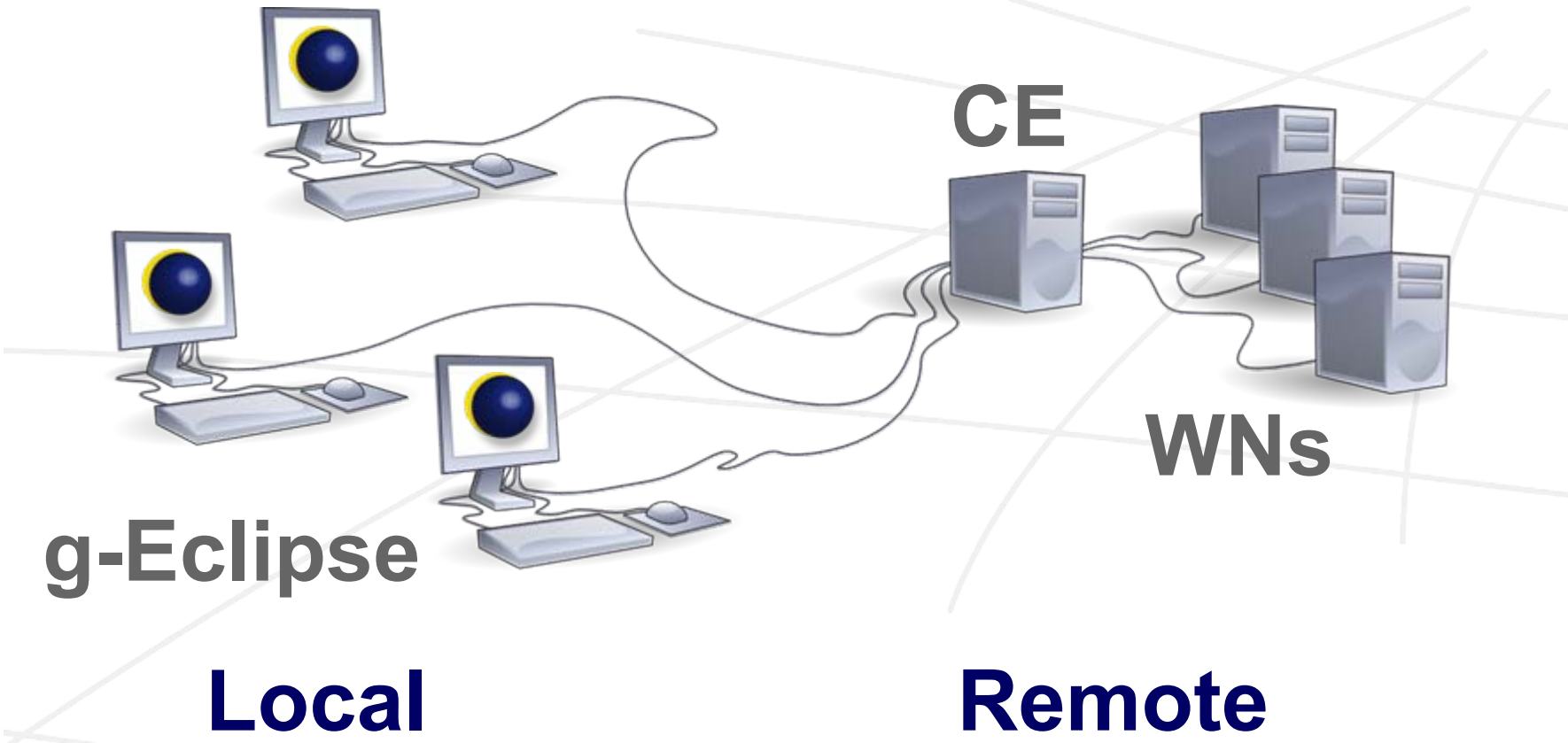
The Grid Developer Perspective

Christof Klausecker (University Linz)

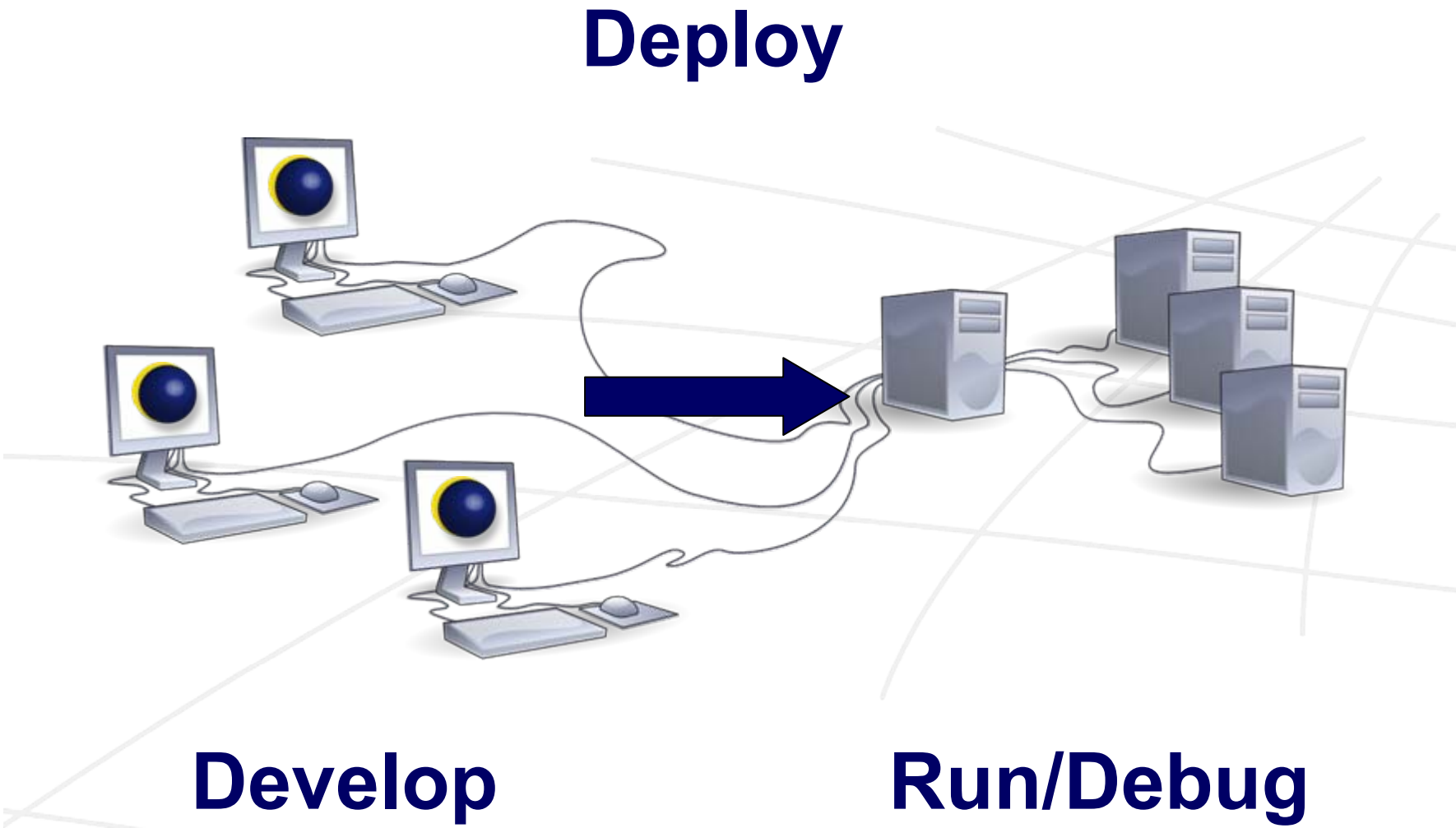
Outline

- Overview
- Arising Problems and their Impact
- Goals
- Used Components and Offered Features
- Preview

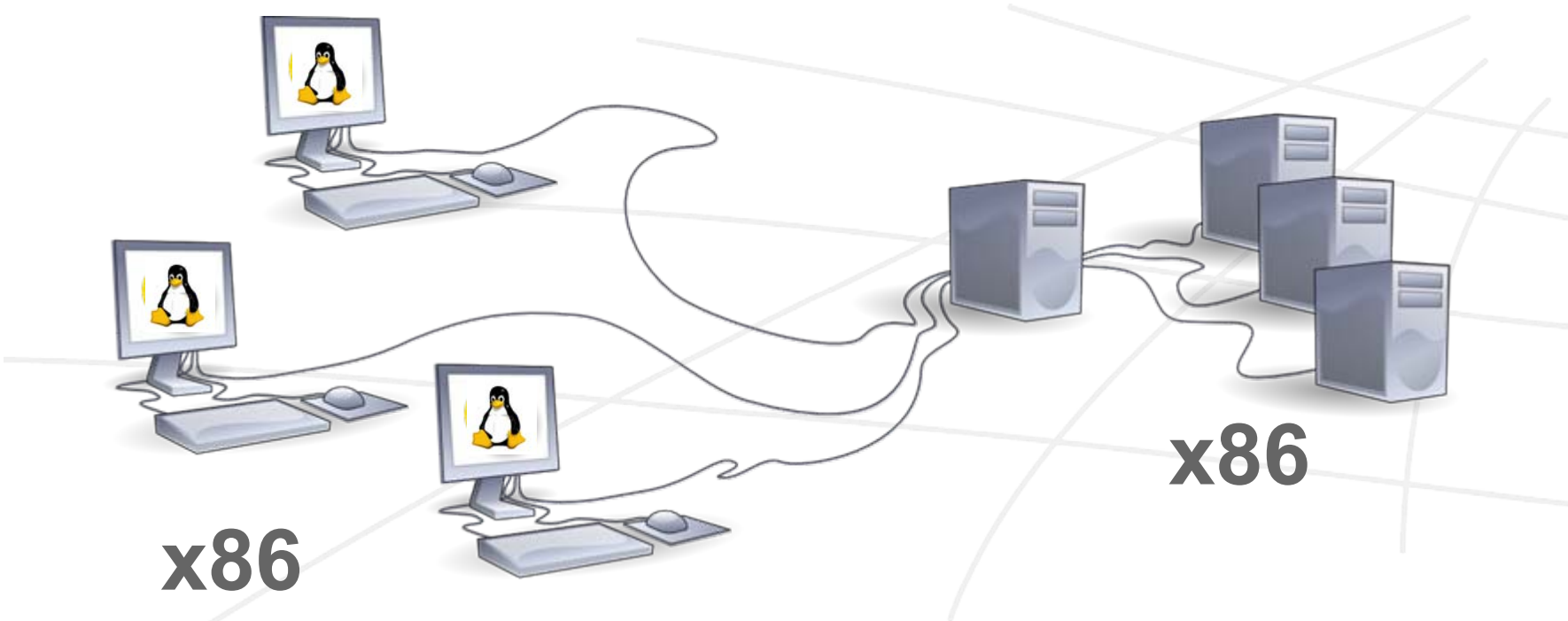
Overview



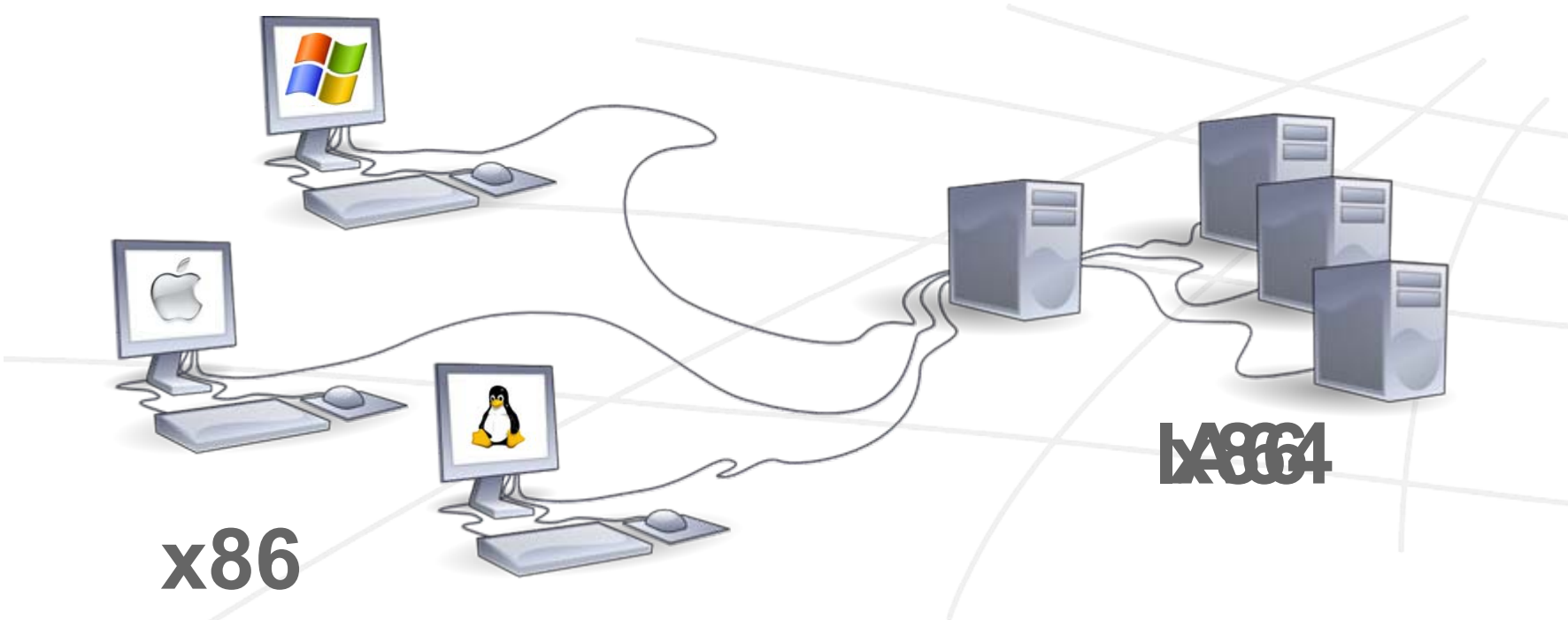
Objective



Wishful Thinking

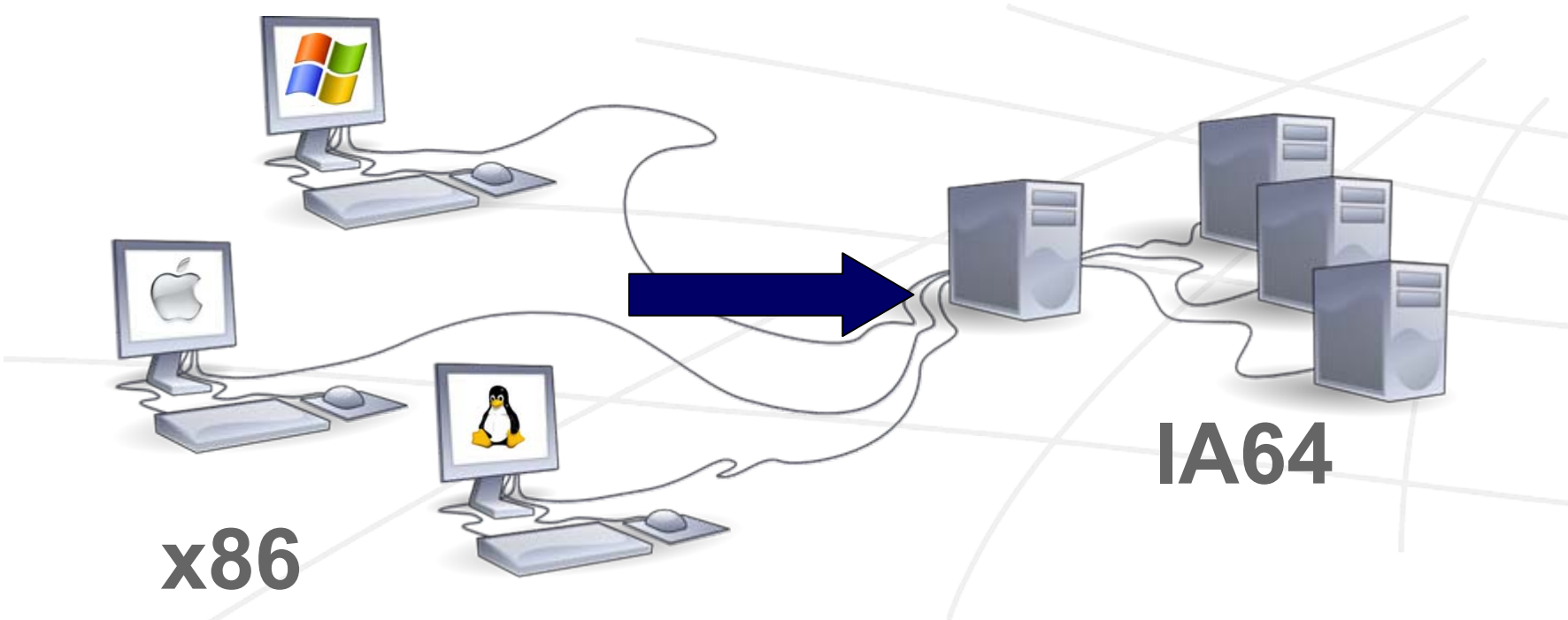


Reality



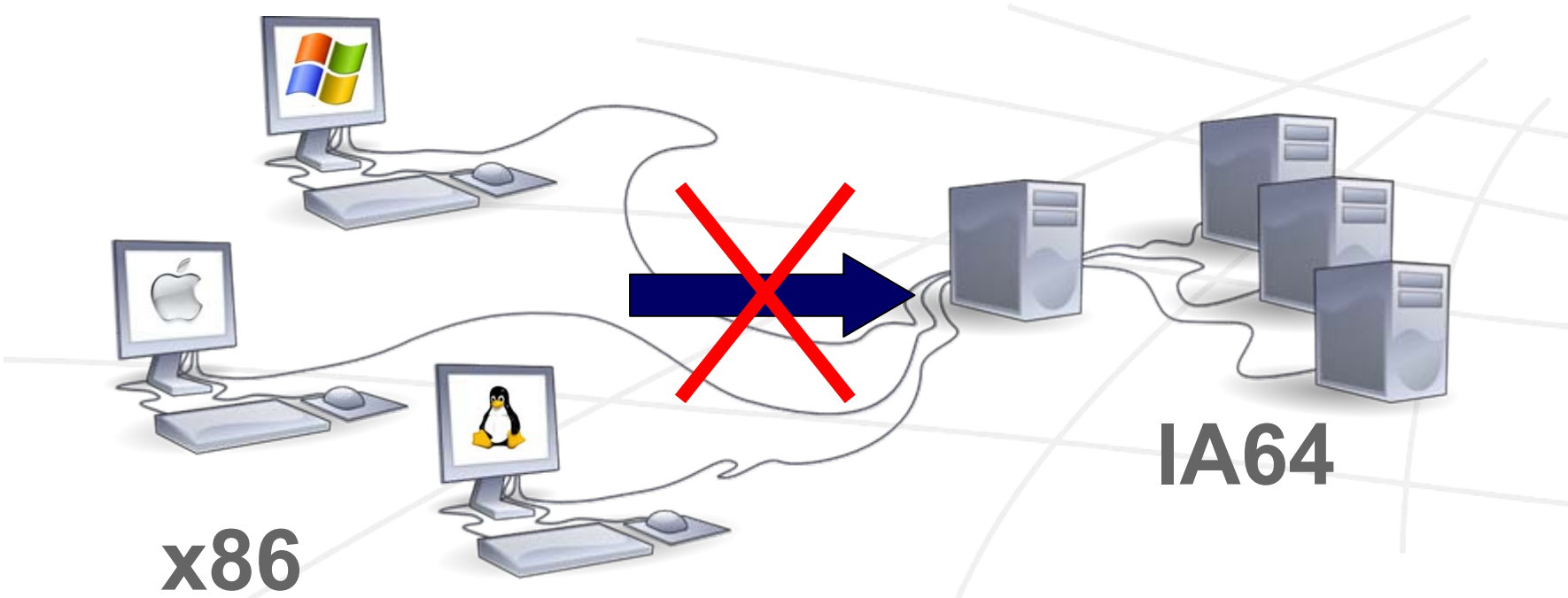
Impact

- distributed nature
  - data staging
  - remote execution
  - remote debugging
- heterogeneity
  - distinguish between
    - Java
    - C/C++



- developed applications can be compiled locally, staged and started/debugged remotely afterwards
- perhaps need to be built for specific java version

C/C++



- locally compiled applications won't run remotely (different architecture, operating system, libraries ...)
- need for cross compiling or building remotely

Goals



- provide the similar tooling as if developing for your Desktop Machine
- minimize learning efforts
- develop applications using the powerful IDE features of the Eclipse JDT and CDT

Used Components

- didn't reinvent the wheel, but used existing components

- local side
  - JDT
  - CDT
- remote side
  - GDB
  - JAVA VM

- connection between the local and remote side
  - GridFTP for data transfer
  - glogin for communication

glogin

- used for interactive connections between the local computer and the remote CE / WNs
- offers functionality similar to SSH
  - standard shell
  - port forwarding
  - uses proxy certificates for authentication
- java version integrated into g-Eclipse (needs a public ip and open local ports for incoming connection)
- native version required to be installed on the remote side

Offered Features

- Remote Builder to compile and link C/C++ applications remotely
- Launch Configurations to run and debug applications on the grid
- Ways to deploy the developed application in order to make them available for grid users

Remote Builder

- copies the source files incrementally to the Grid
- invokes make on the remote machine
- forwards the output and marks the errors in the local source files
- copies the compiled executable back to the local machine for deployment and debugging.
- no need for a local compiler

Launch Configurations



- run and debug applications on the grid
- offer possibility to stage executables
- offer interactivity by connecting the remote std in/err/out to local eclipse console
- connect the remote debugger to local eclipse
- support
  - Java Applications
  - C/C++ Applications
  - preliminary support for MPI Applications

Debugging

- uses standard Eclipse debugging interface
- Java
  - connects the JAVA VMs Debugger via port forwarding to our local g-Eclipse and the JDT
- C/C++ (2 versions)
  1. Remote Builder
     - the files are compiled on the remote machine
     - compiled for the right architecture
  2. Local Builder
     - The files are compiled locally
     - Remote libraries are needed to be available locally
     - May not work if the architectures are different

Application Deployment

- after developing an application that is
  - free of bugs
  - usable for others

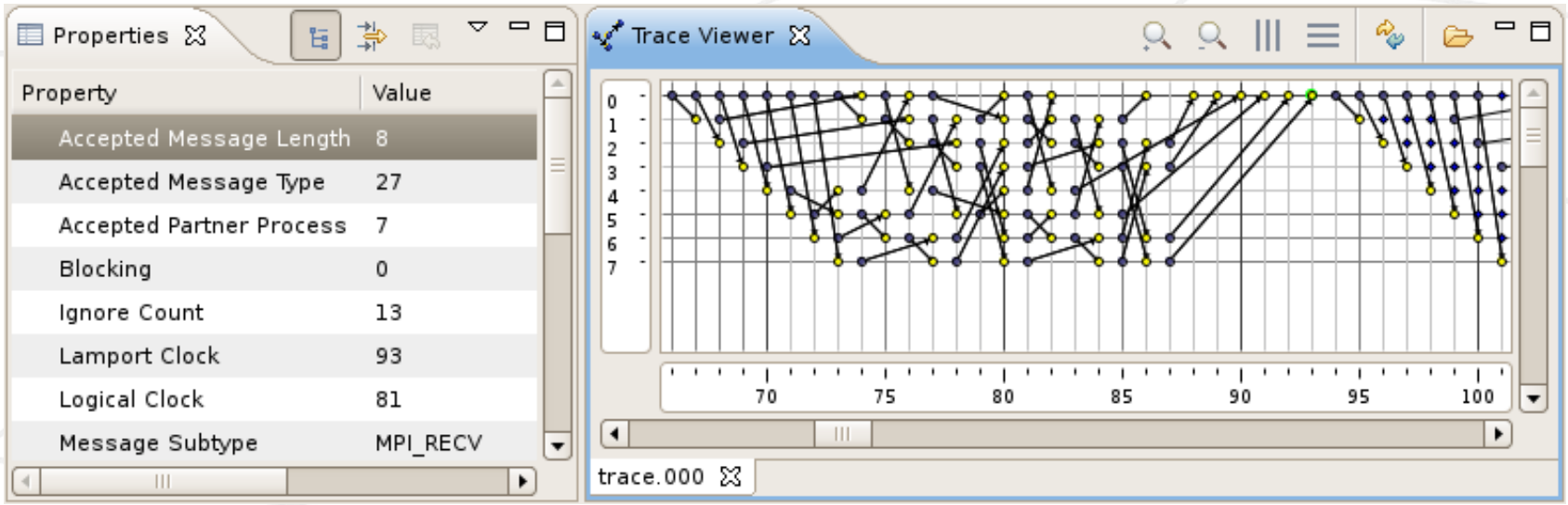
- deploy the application using g-Eclipse
  - save the stage-in step for future job runs
  - supports different methods

Preview

- Parallel Debugging
  - debug MPI Applications running on the Grid
  - for each process separately
    - view stdout
    - view/change variables content
    - step/set breakpoint (using the CDT process filter)
    - ...
  - support different MPI Implementations
  - Process View to visualize the state of the processes

Event Graph

- support the developer
  - analyse communication
  - search for bottlenecks
  - discover application errors
  - set breakpoints in the graph



Putting it all together

