Recommended Viewing:

Since this is a pictorial presentation, I’ve gone through the effort to type up what I normally say in the “notes” section.

To enjoy the animations, I recommend printing out the “notes” then watching via the slide show viewer.

--Gary Kumfert

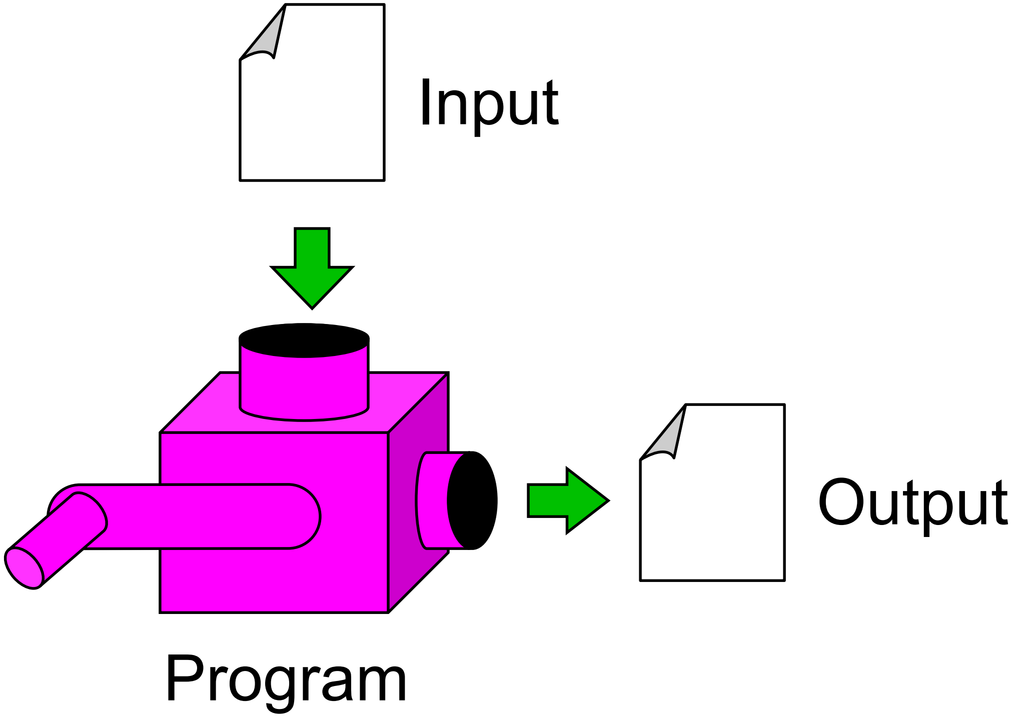
A Pictorial Introduction to Components in Scientific Computing

Gary Kumfert

with

Steve Smith, Scott Kohn,
Tom Epperly, Tammy Dahlgren,
& Bill Bosl

Once upon a time...

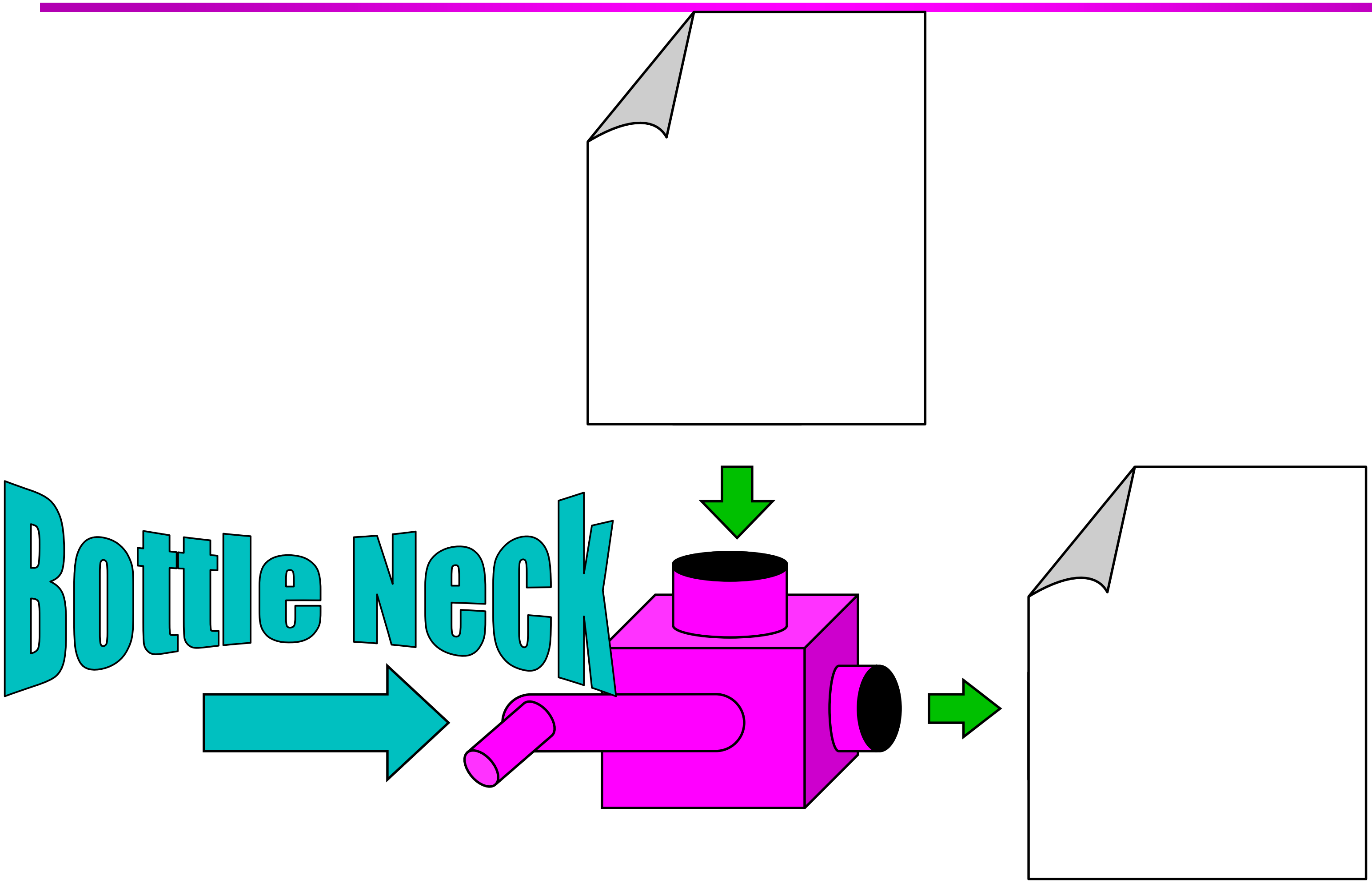


Input

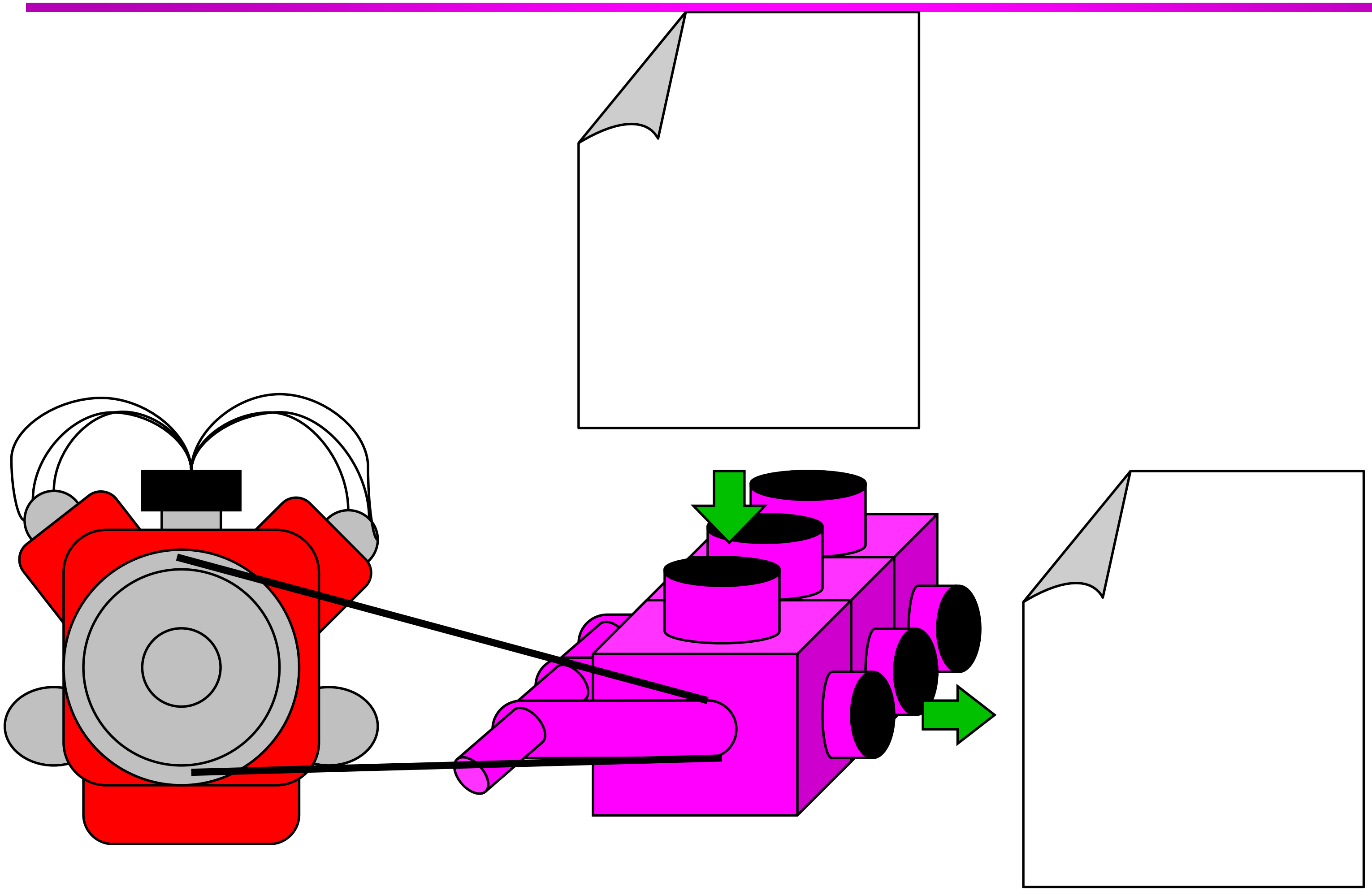
Program

Output

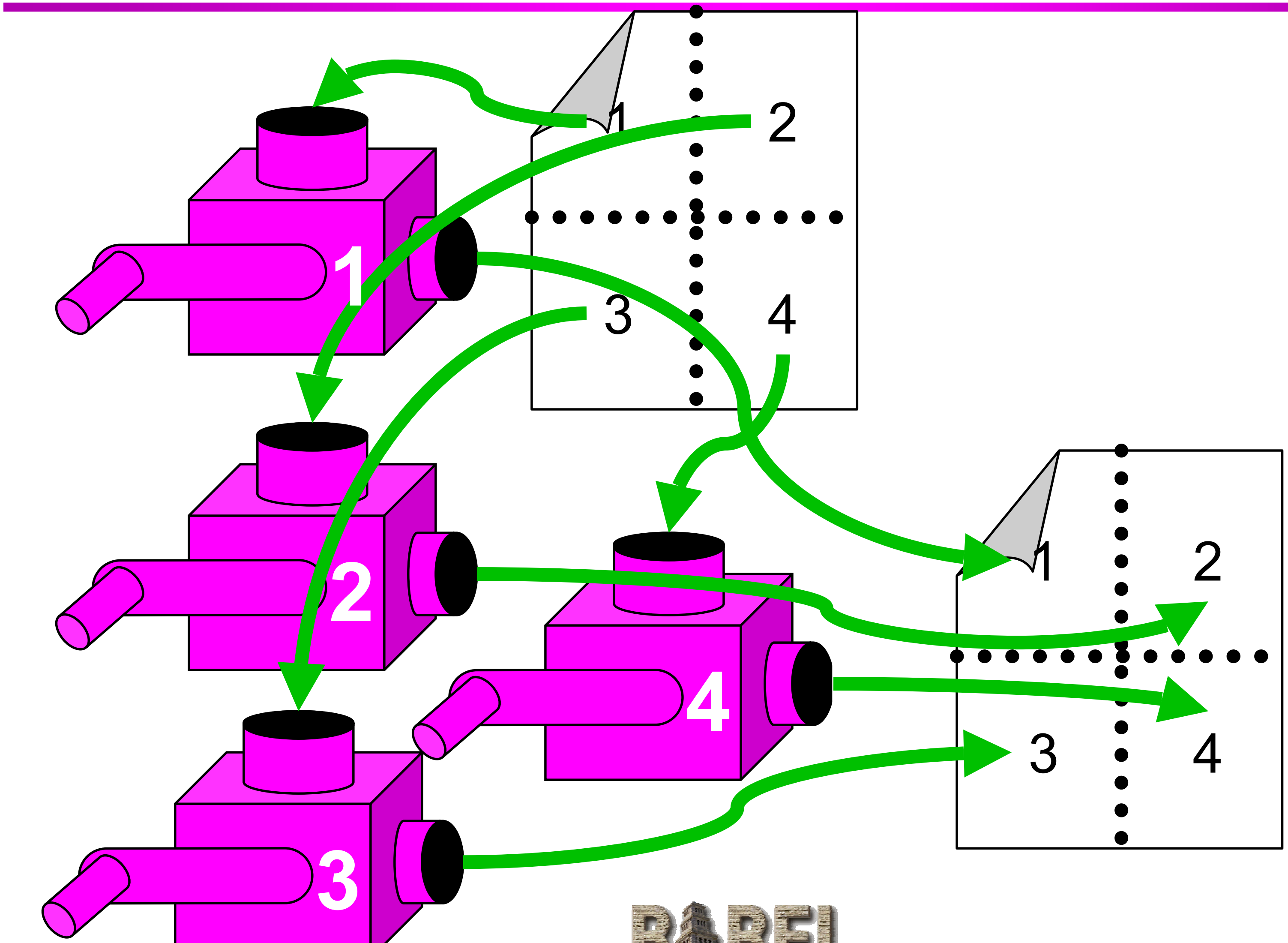
As Scientific Computing grew...



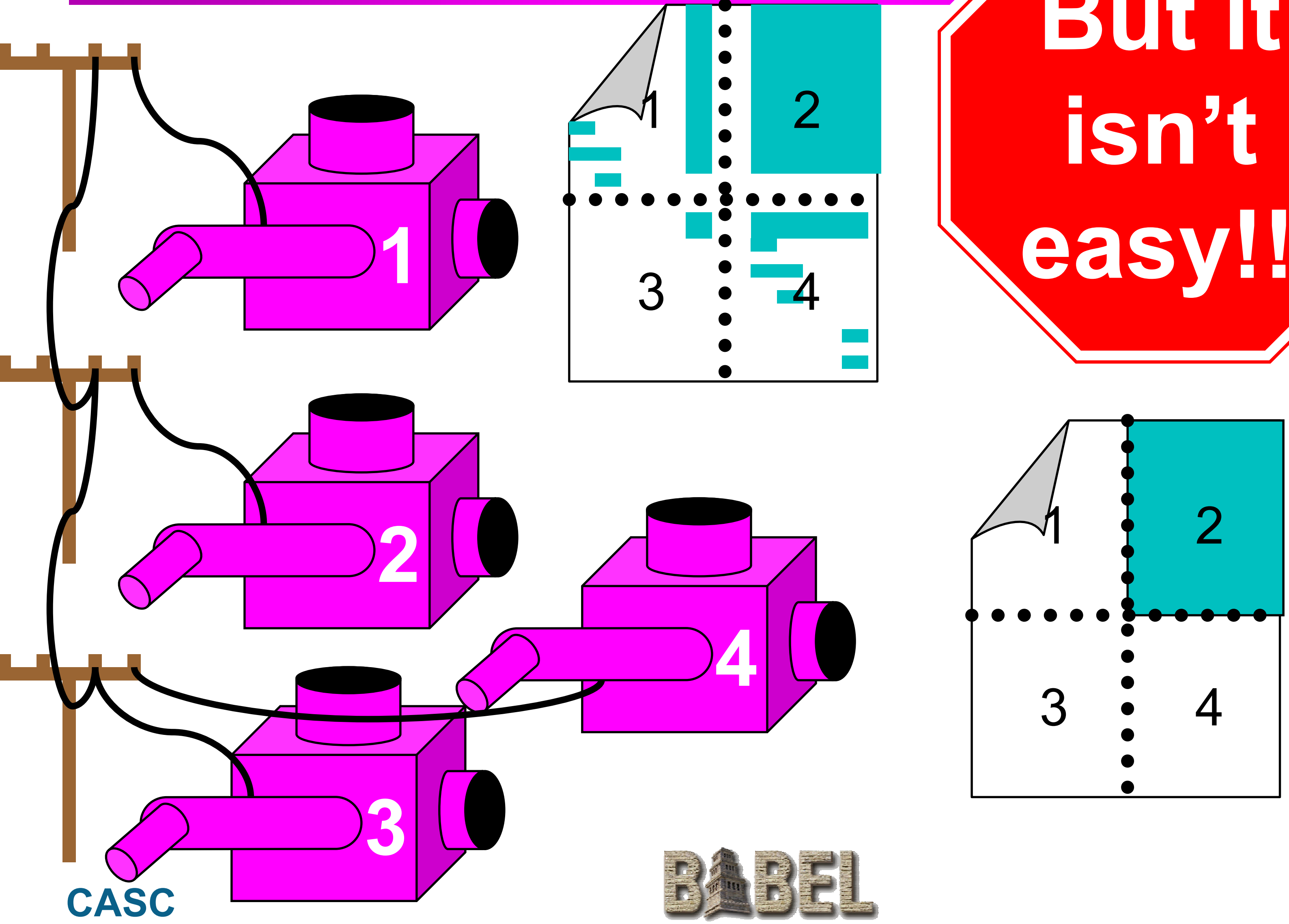
Tried to ease the bottle neck



SPMD was born.

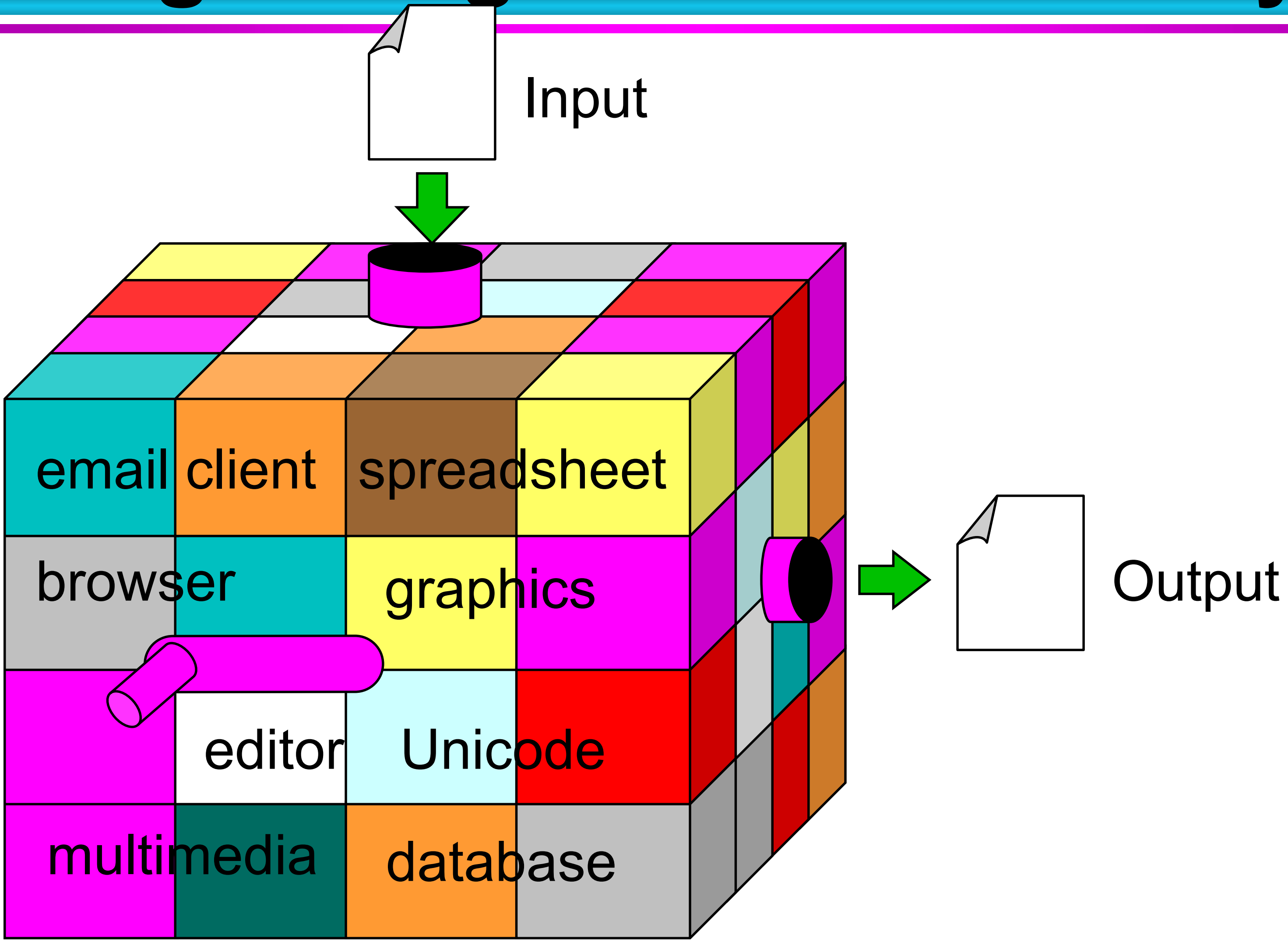


SPMD worked.

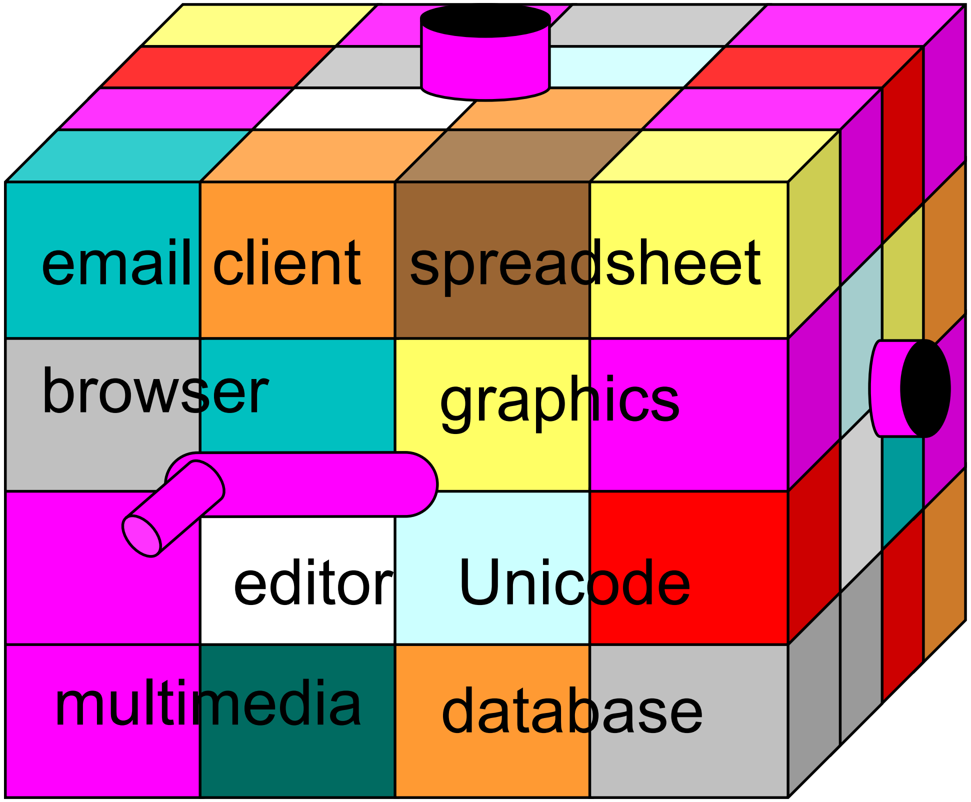


But it isn't easy!!!

Meanwhile, corporate computing was growing in a different way

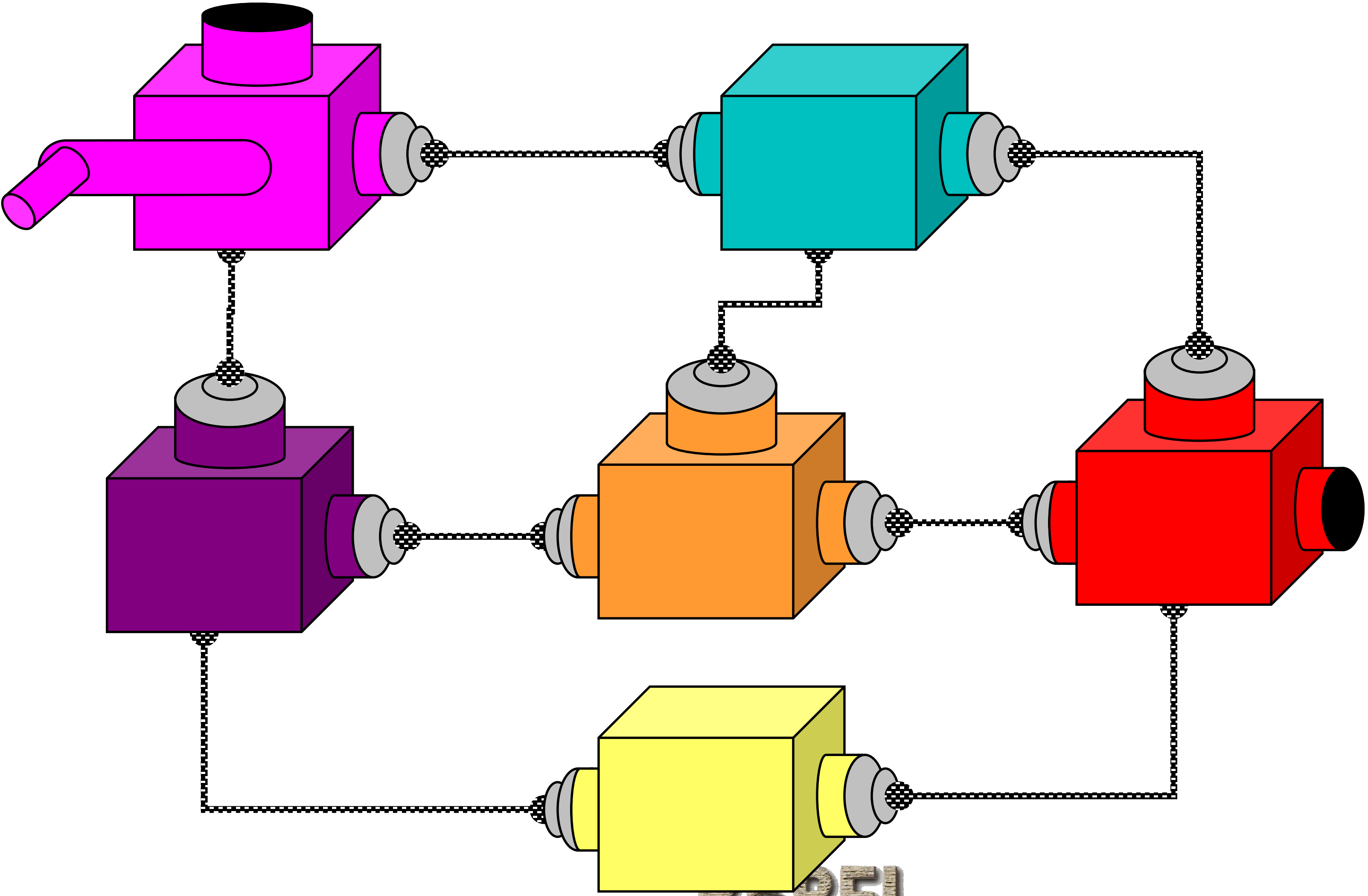


This created a whole new set of problems...

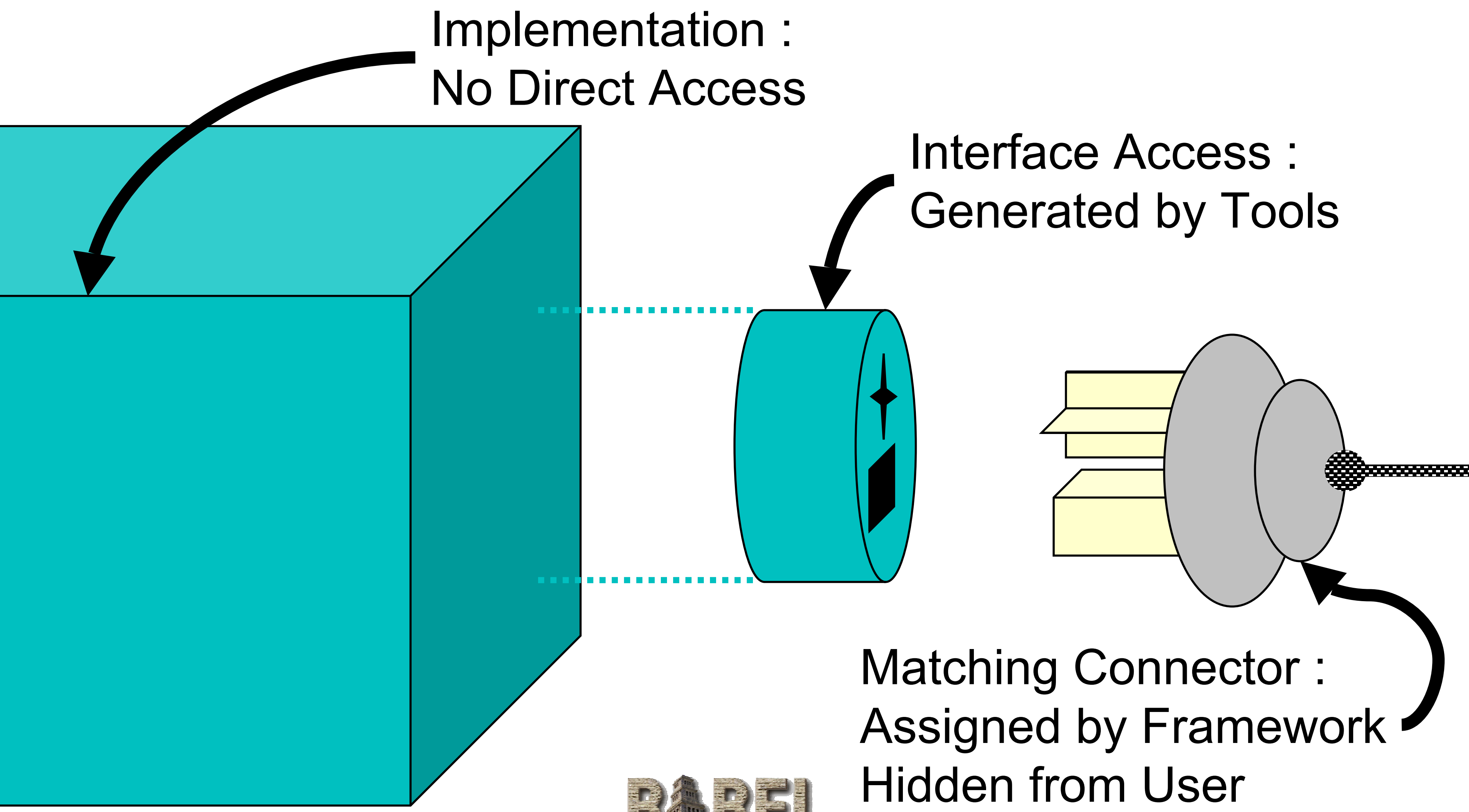


- Interoperability across multiple languages
- Interoperability across multiple platforms
- Incremental evolution of large legacy systems (esp. w/ multiple 3rd party software)

Component Technology addresses these problems



So what’s a component ???

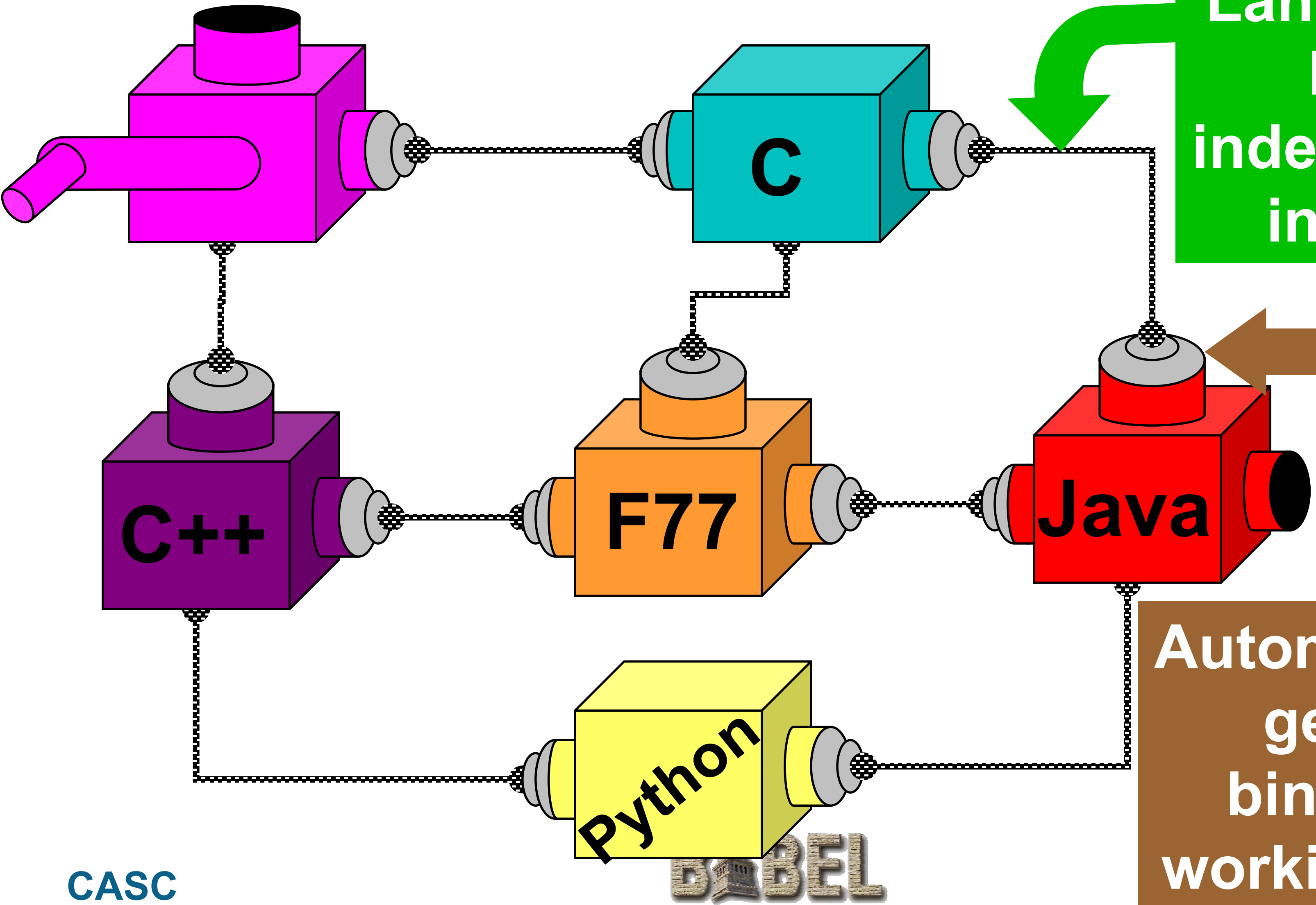


Implementation :
No Direct Access

Interface Access :
Generated by Tools

Matching Connector :
Assigned by Framework
Hidden from User

1. Interoperability across multiple languages



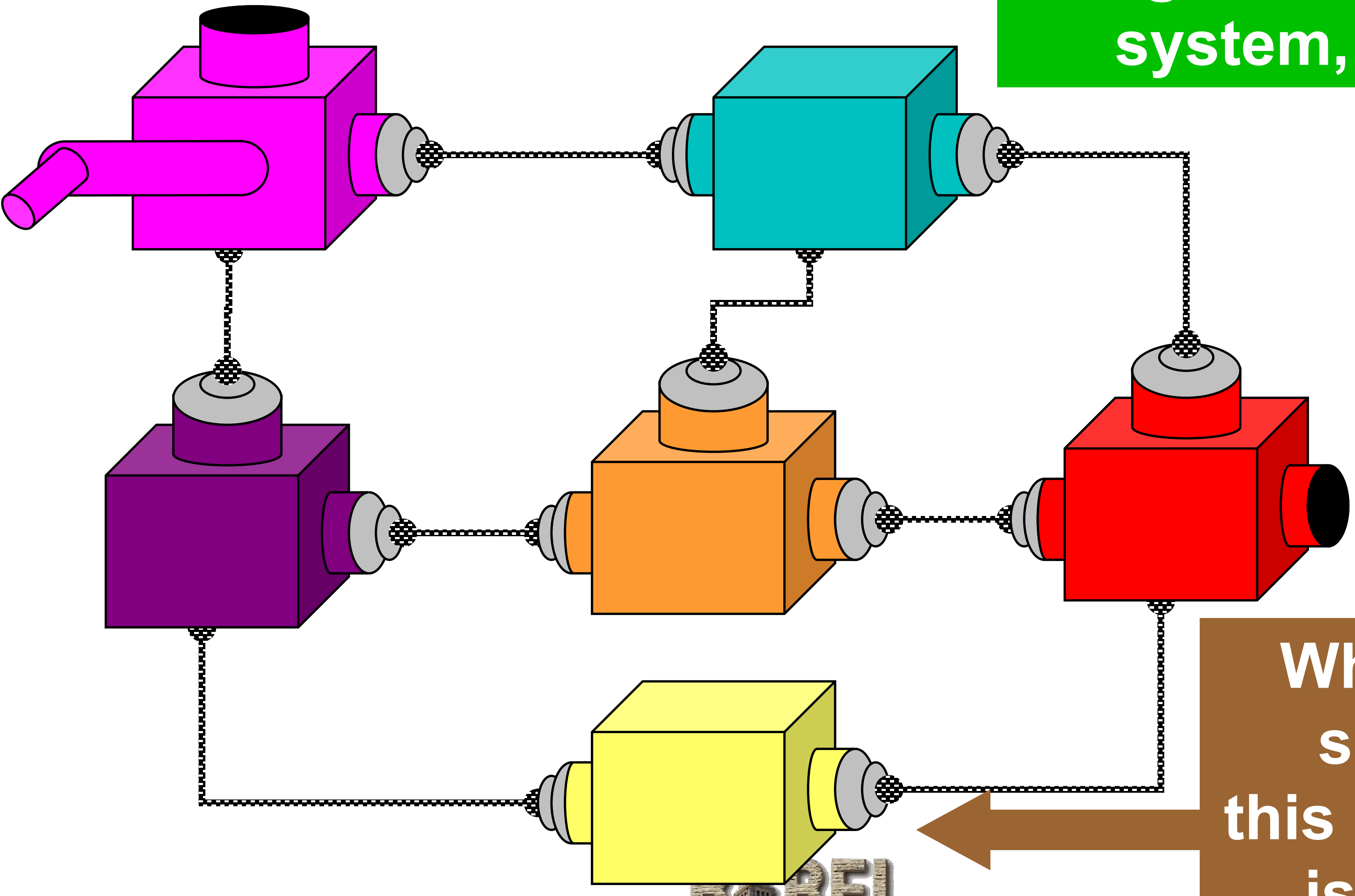
Language & Platform independent interfaces

Automatically generated bindings to working code

CASC

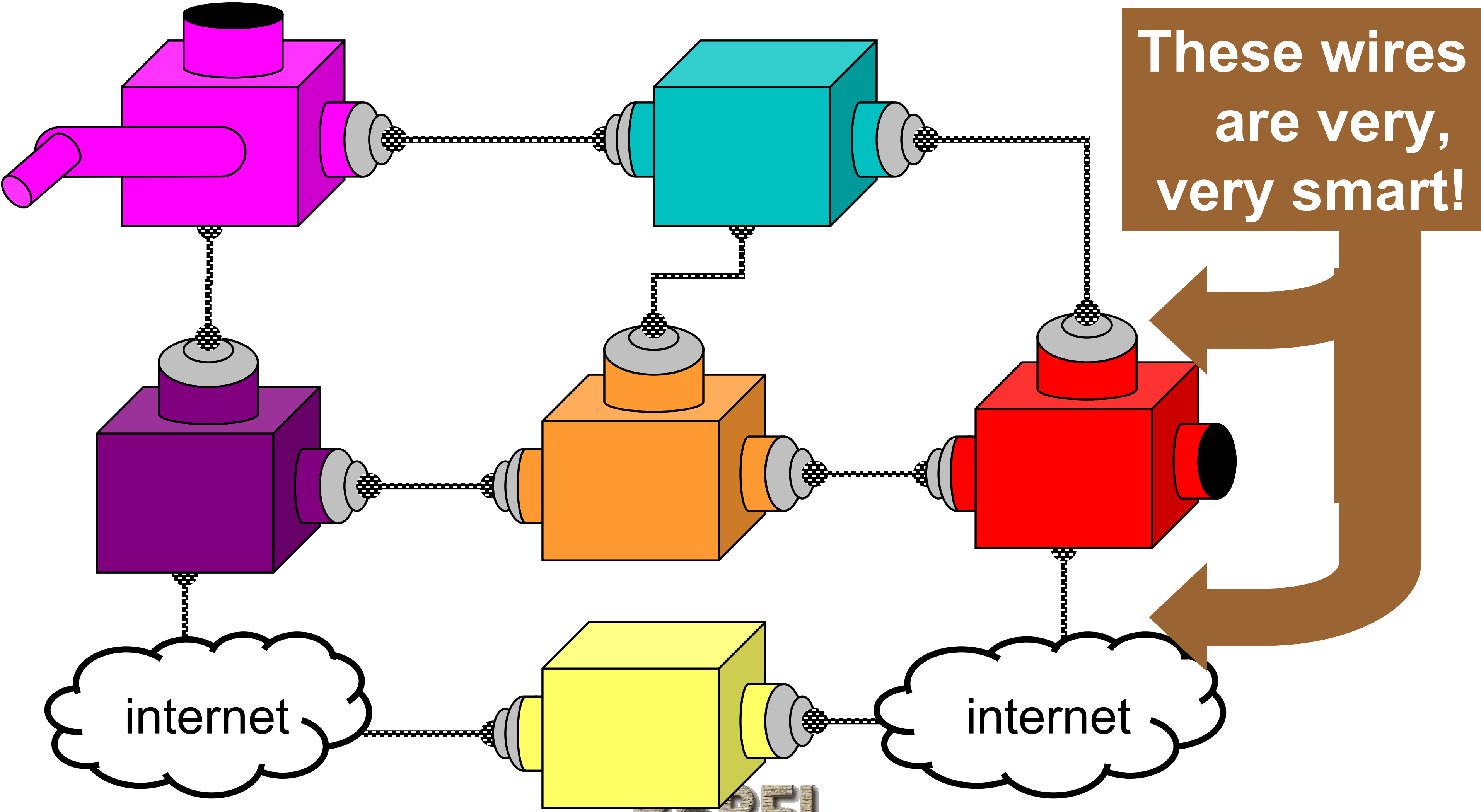
2. Interoperability Across Multiple Platforms

Imagine a company migrates to a new system, OS, etc.



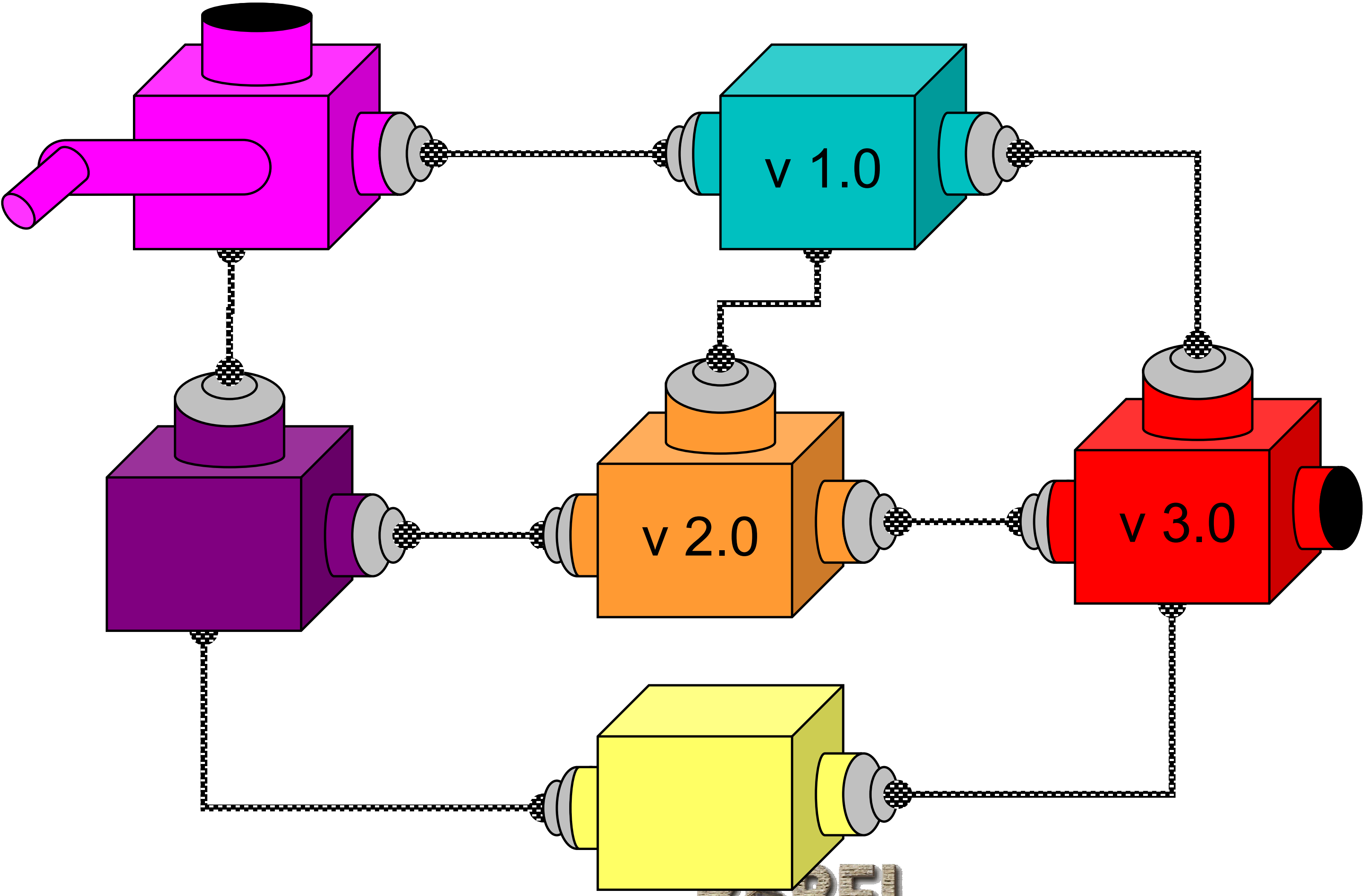
What if the source to this one part is lost???

Transparent Distributed Computing

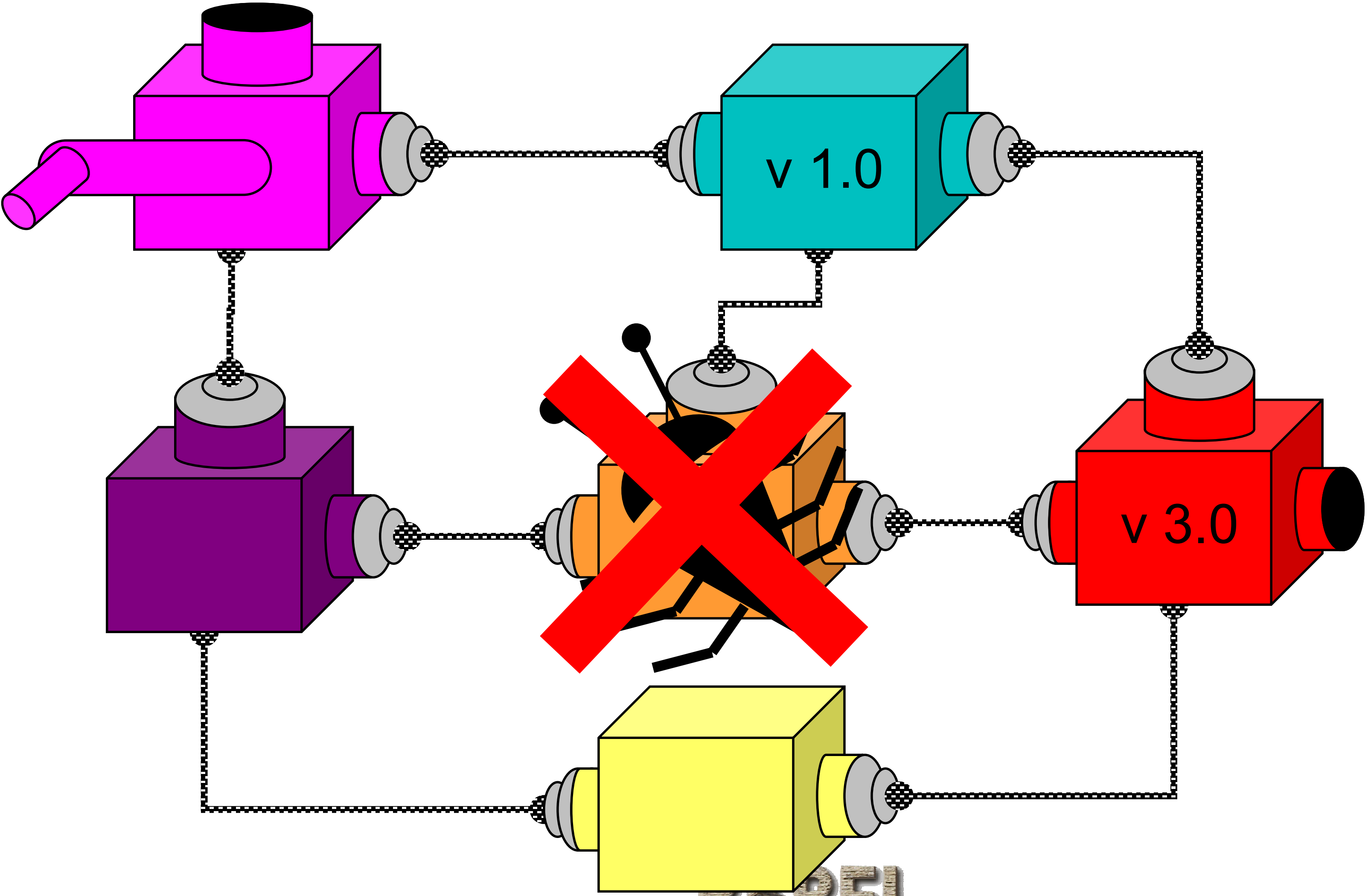


These wires are very, very smart!

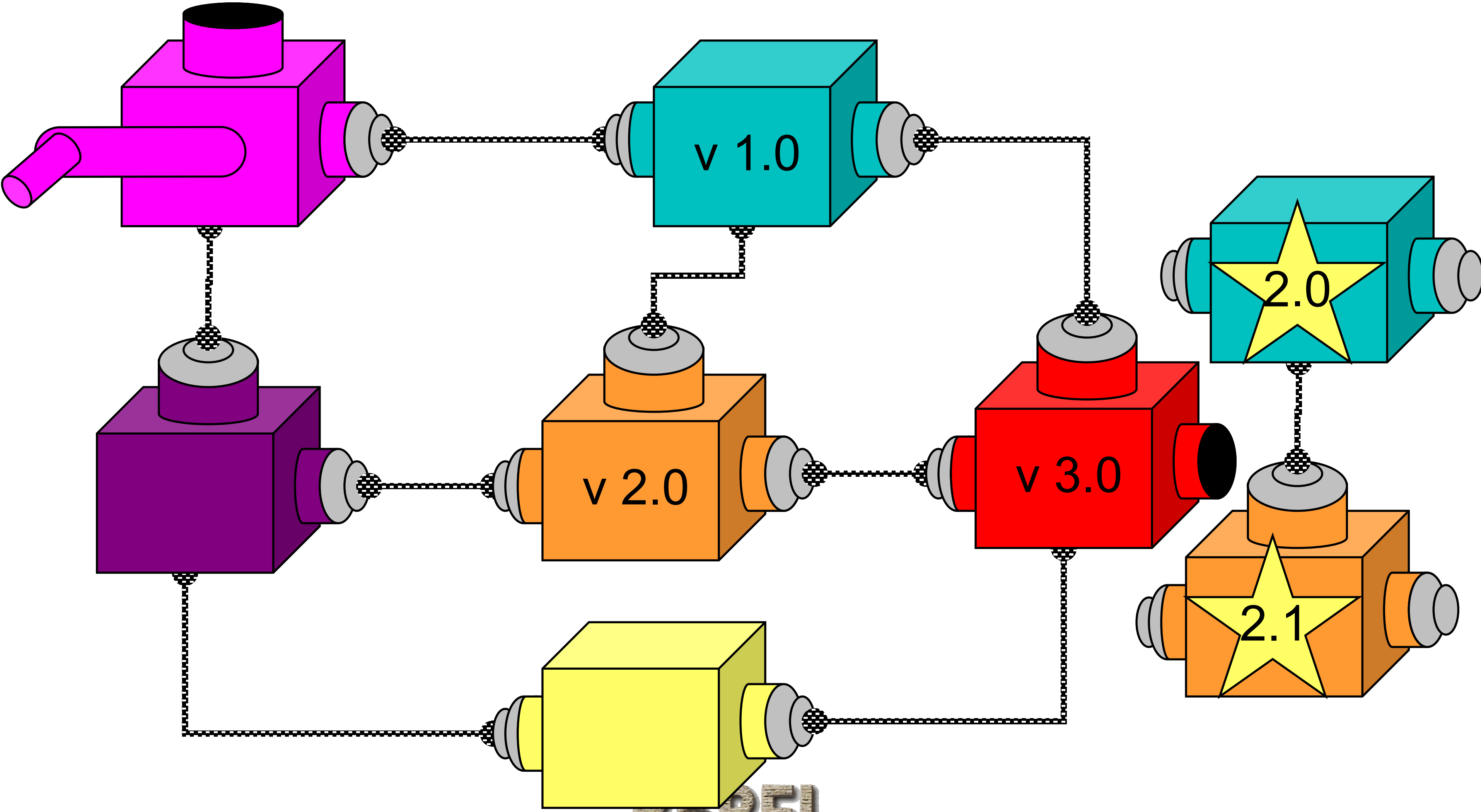
3. Incremental Evolution With Multiple 3rd party software



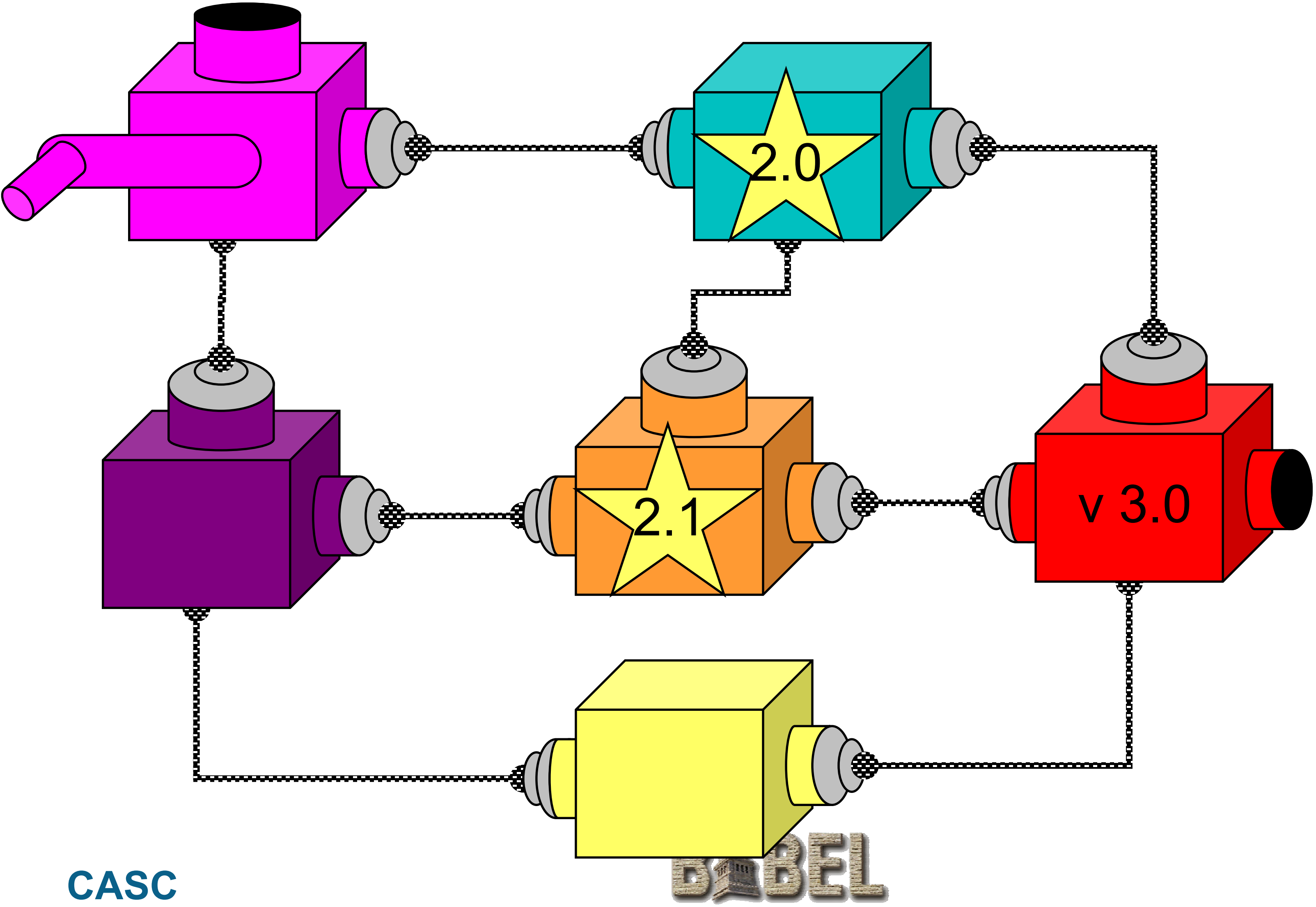
Now suppose you find this bug...



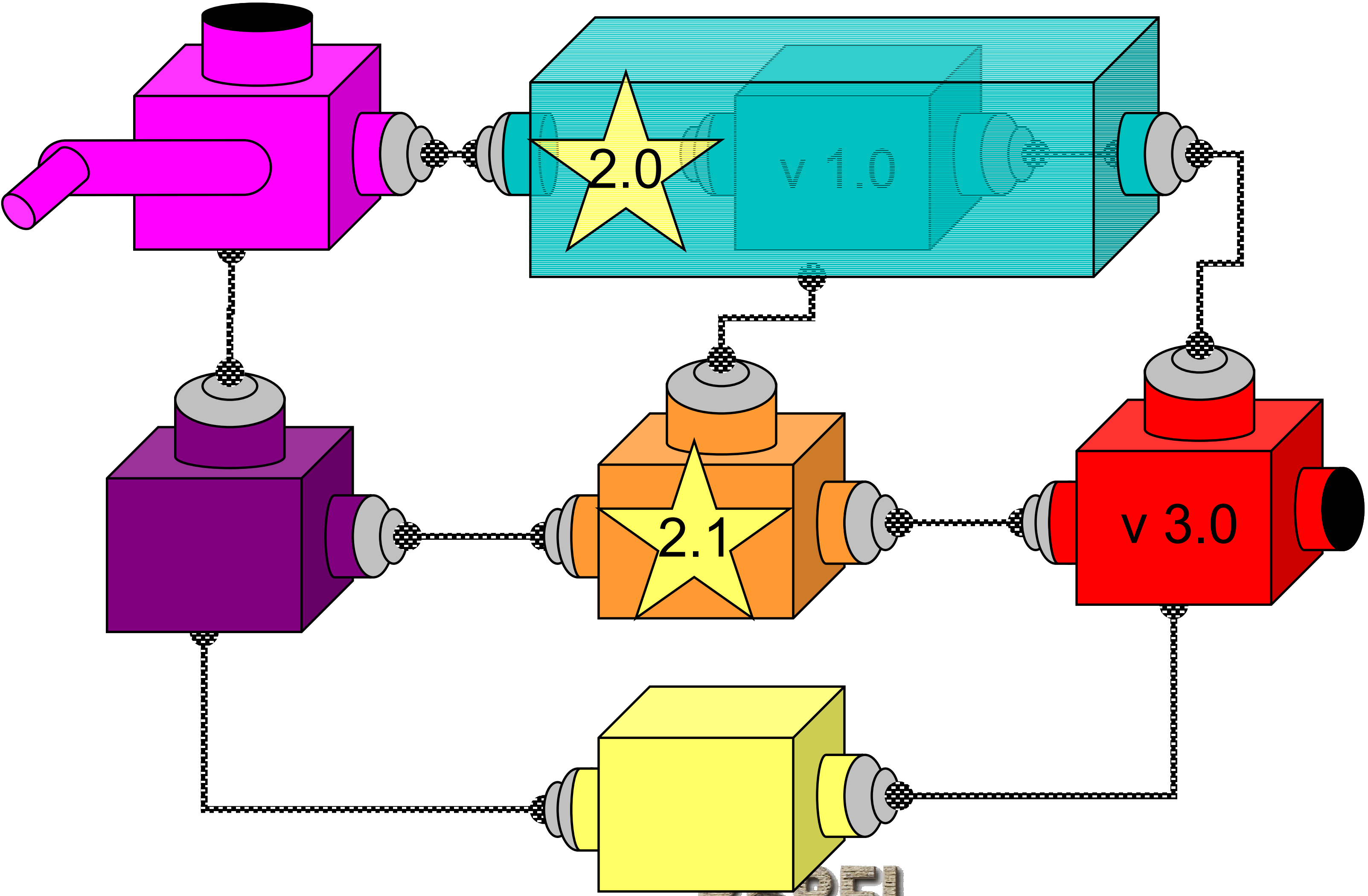
Good news: an upgrade available
Bad news: there’s a dependency



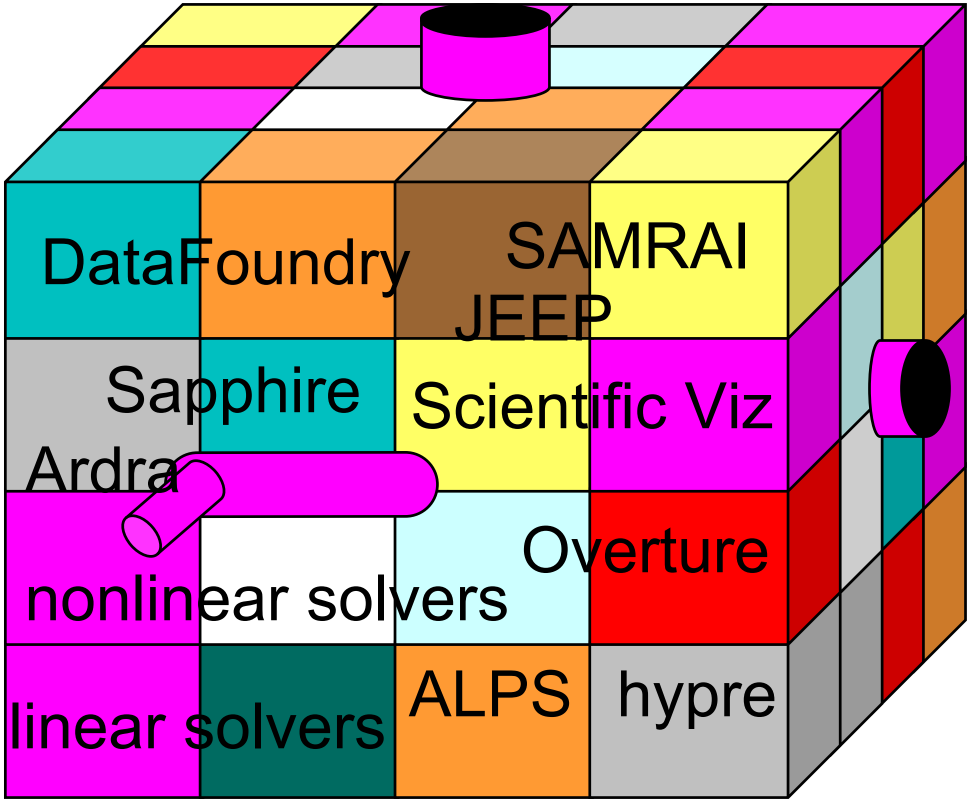
Great News: Solvable with Components



Great News:
Solvable with Components

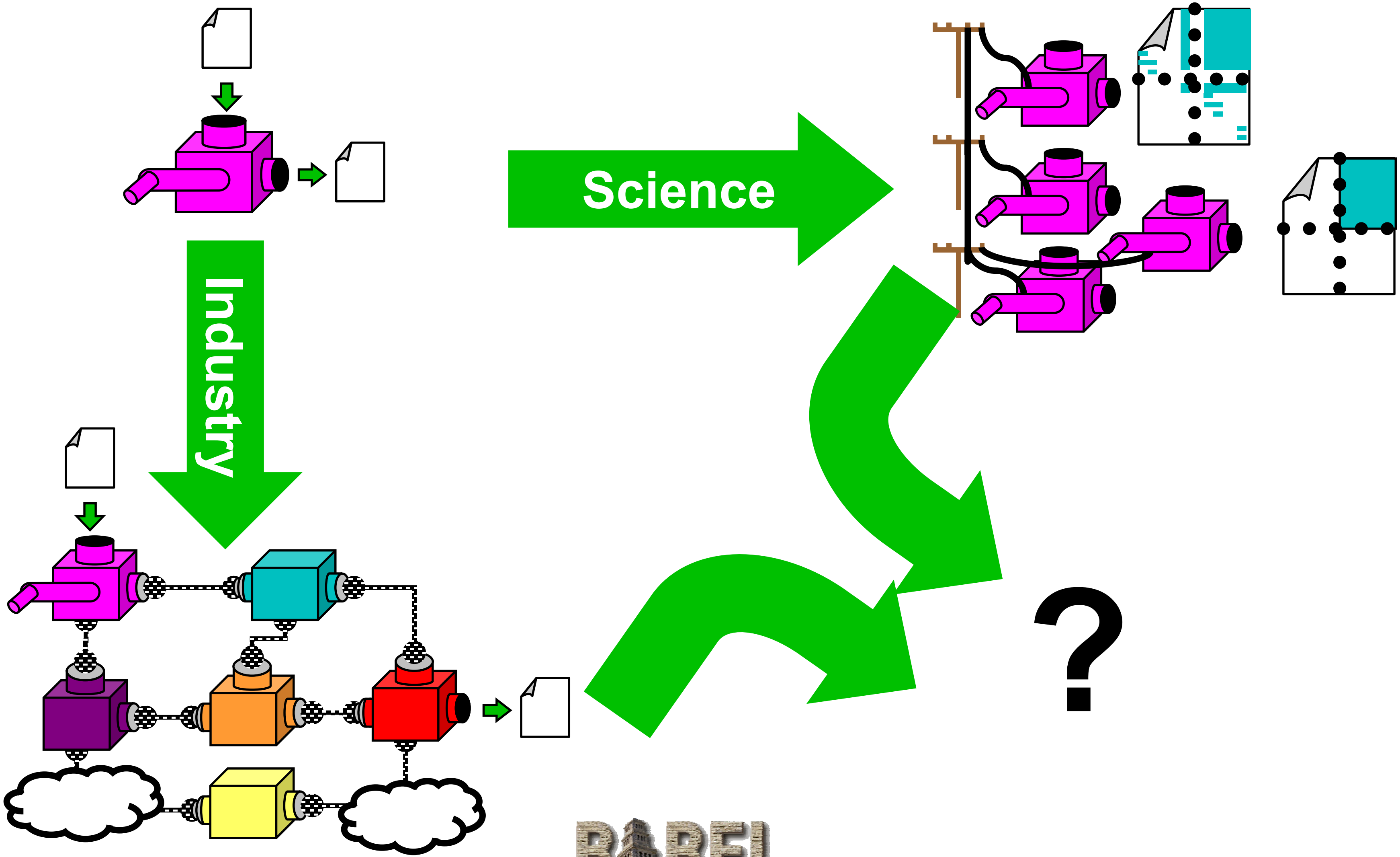


Why Components for Scientific Computing?



- Interoperability across multiple languages
- Interoperability across multiple platforms
- Incremental evolution of large legacy systems (esp. w/ multiple 3rd party software)

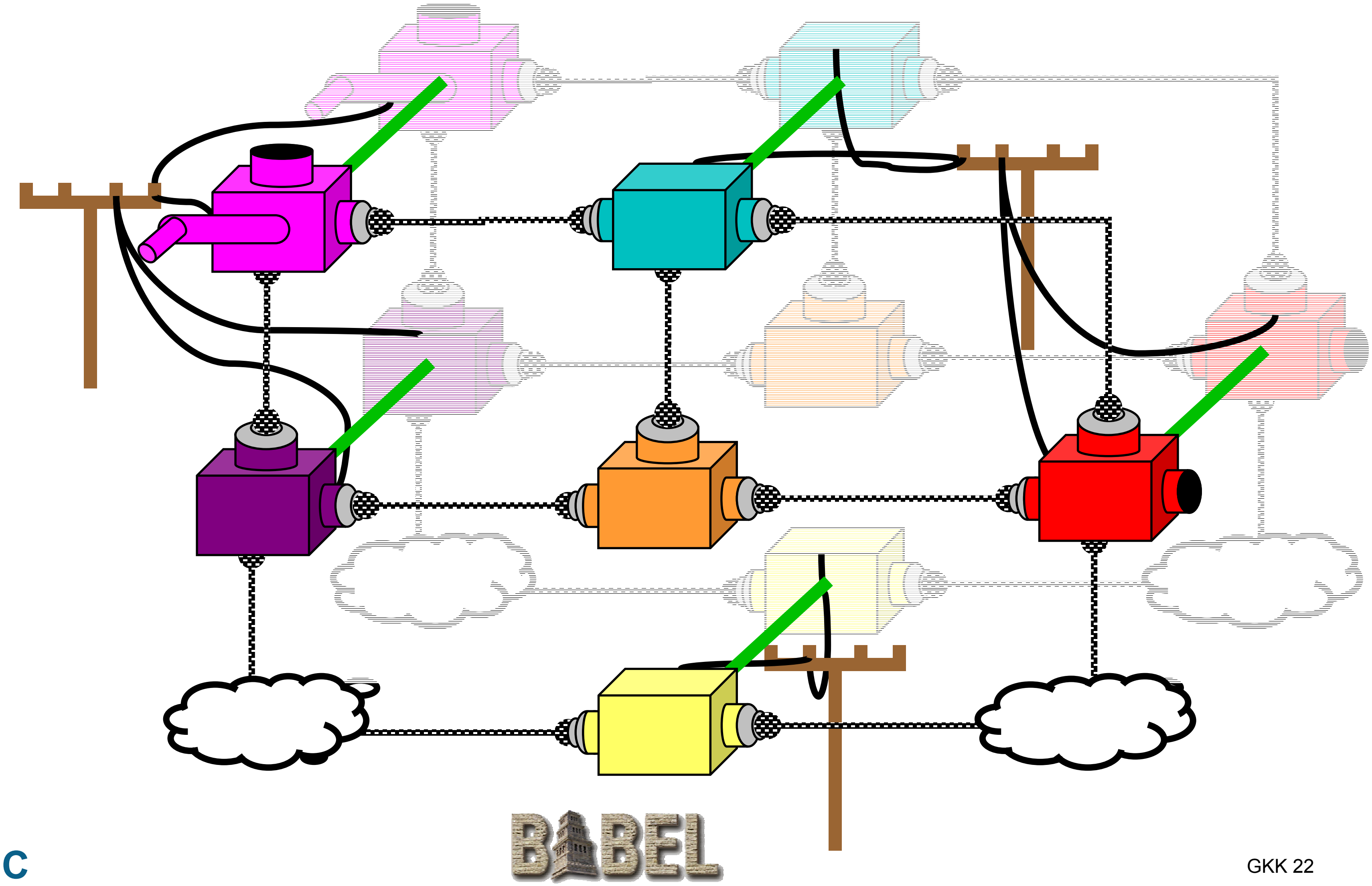
The Model for Scientific Component Programming



CASC

BABEL

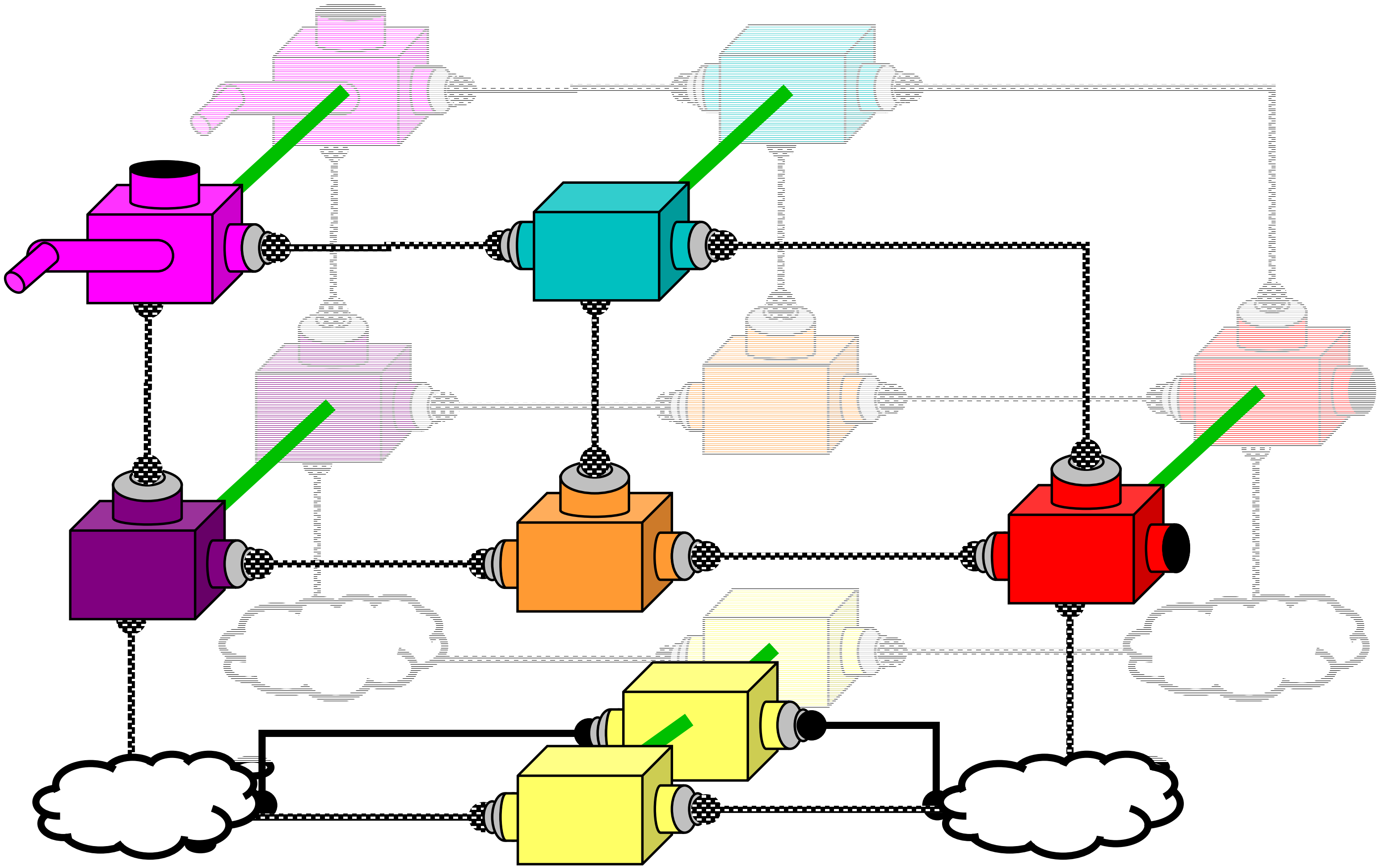
Parallel Distributed Component-Based Application



BABEL

CASC

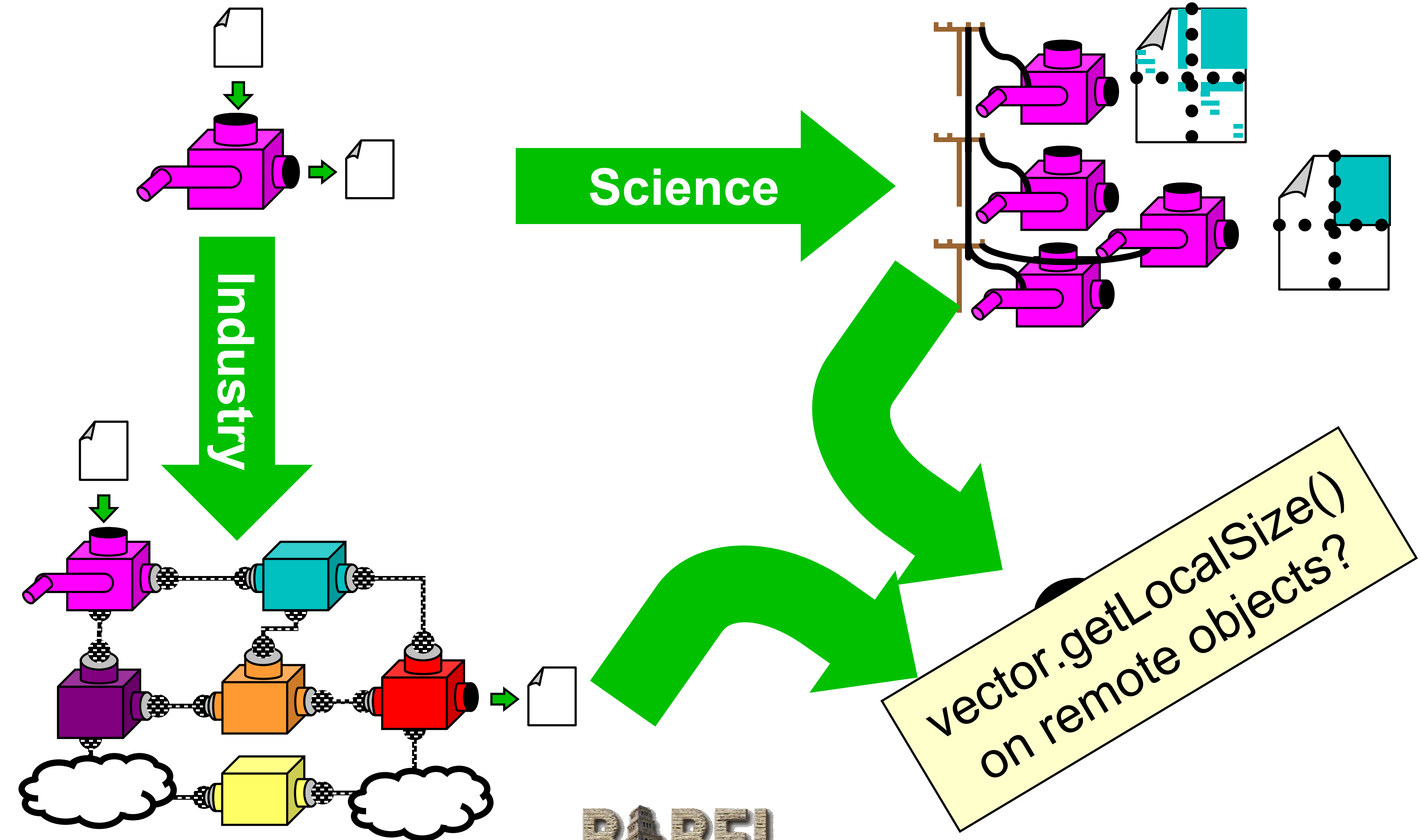
Research Issues: #1. The "MxN Problem"



CASC

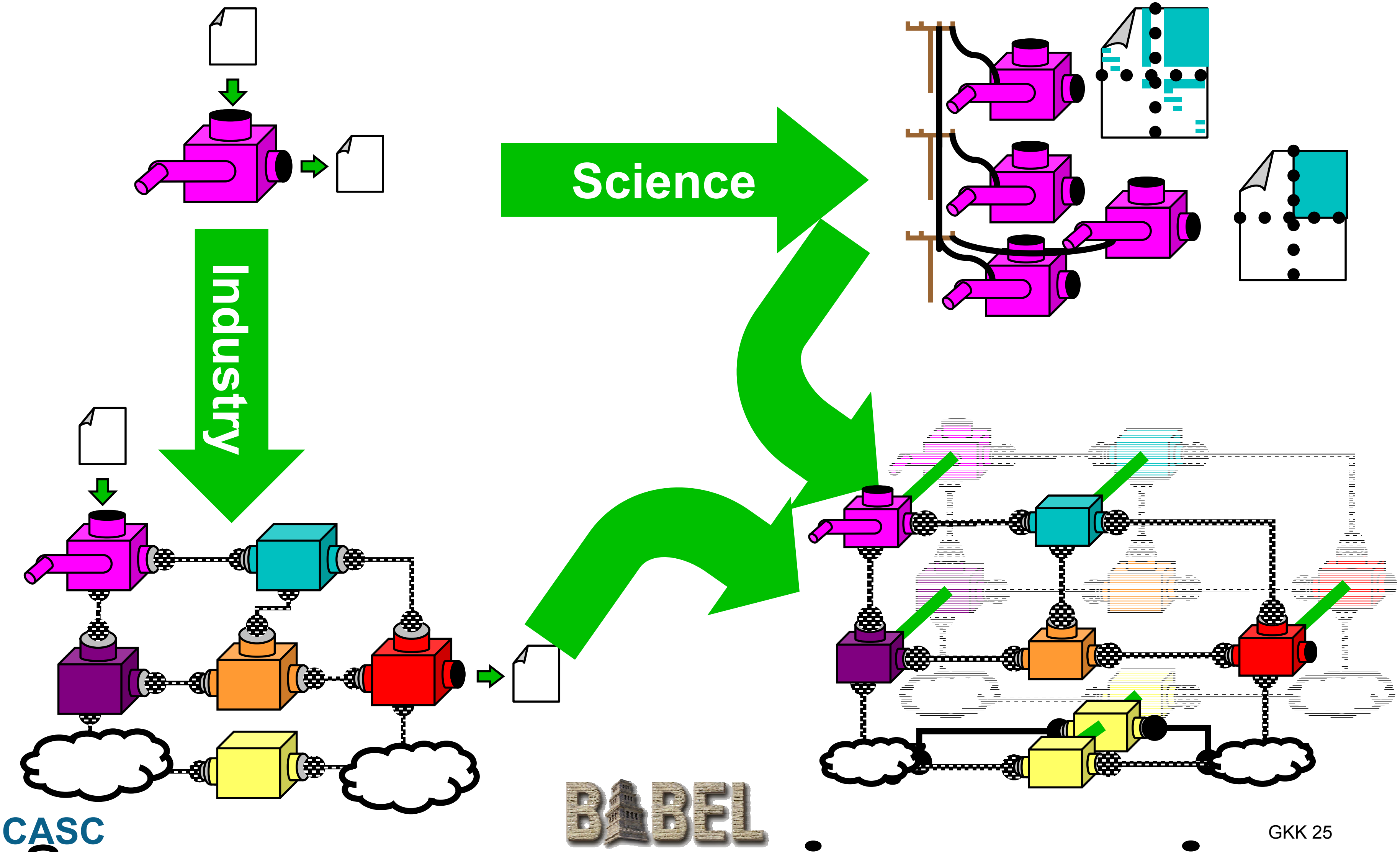
BABEL

Research Issues:
#2: Programming Model



vector.getLocalSize()
on remote objects?

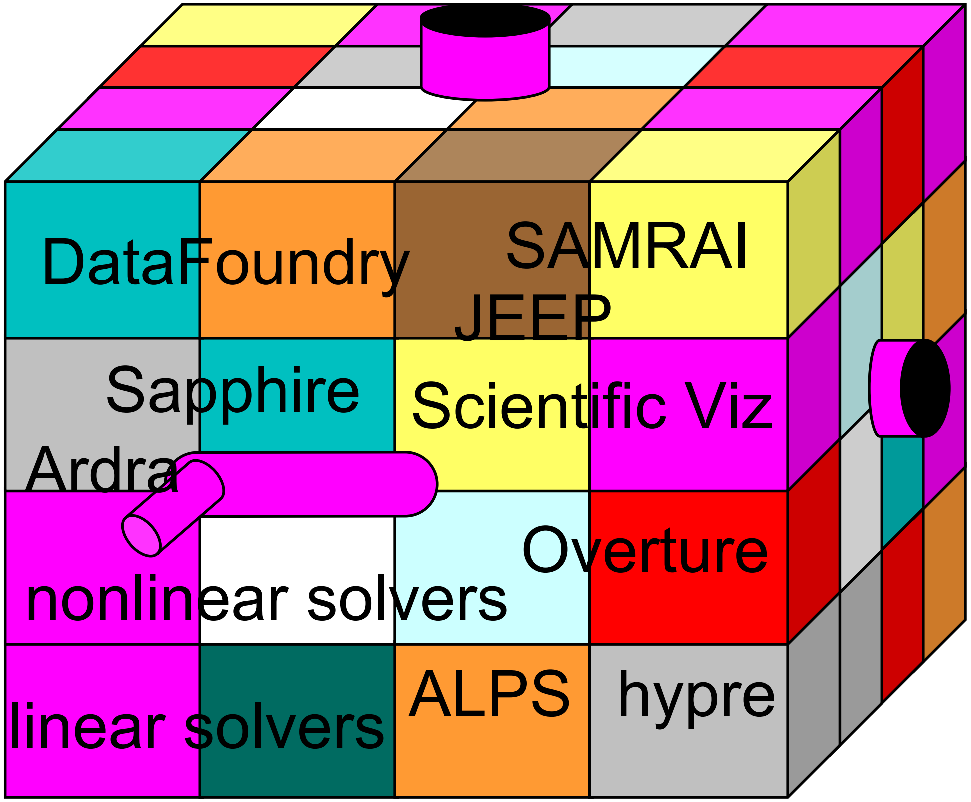
Is This Still SPMD?



Is This Still SPMD?

- No
  - Each “component” may be an entire legacy SPMD code
  - Multiple components (possibly distributed) working together on a single problem
    - MPMD, MCMD, DPMD???
- But
  - Will look like SPMD to application developer
  - Business components look like serial code.

Why Components for Scientific Computing?



- Interoperability across multiple languages
- Interoperability across multiple platforms
- Incremental evolution of large legacy systems (esp. w/ multiple 3rd party software)

BABEL

The End

UCRL-VG-143384

25 Apr 2001

Work performed under the auspices of the U. S. Department of Energy by the University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48