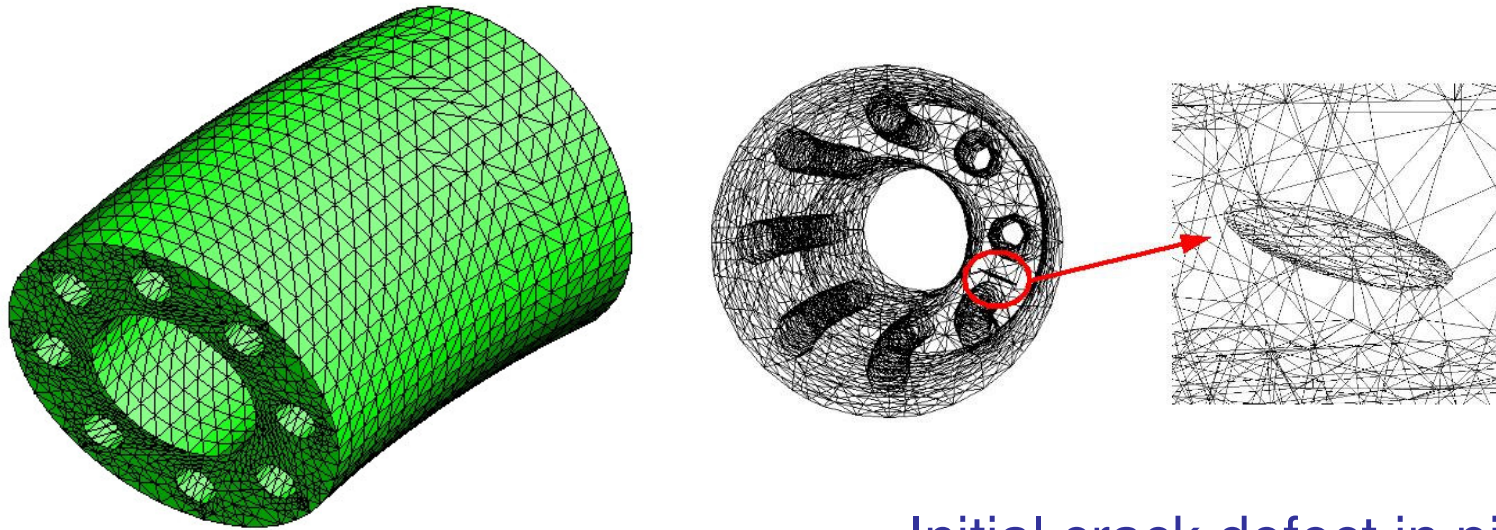


# Focus problem for ASP: Pipe

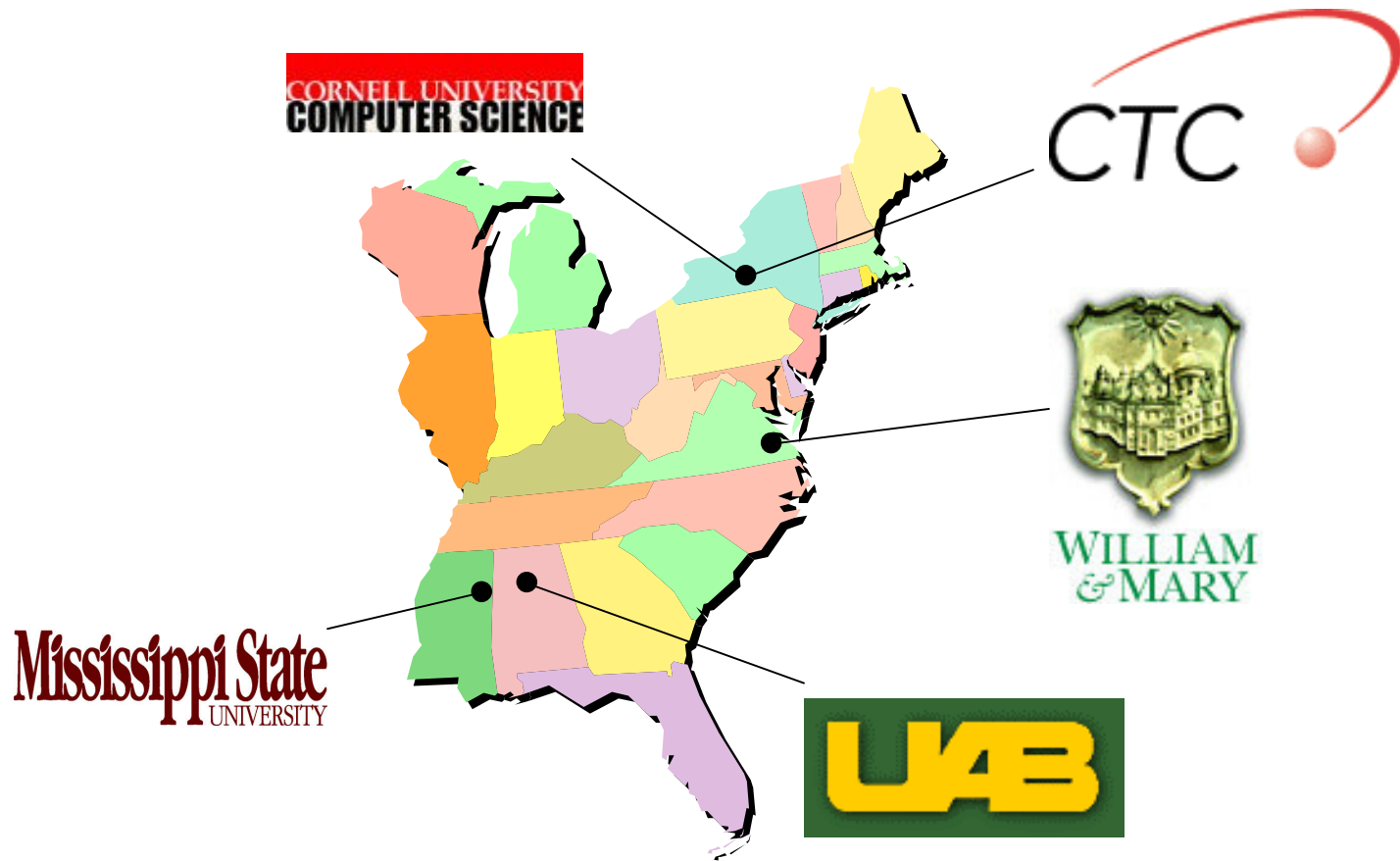


- Idealized rocket engine segment
- Large passage
  - Chemically reacting, high-pressure, high-velocity gas
- Smaller passages
  - Cooling fluid

- Initial crack defect in pipe wall
- Question: How does the flaw propagate?



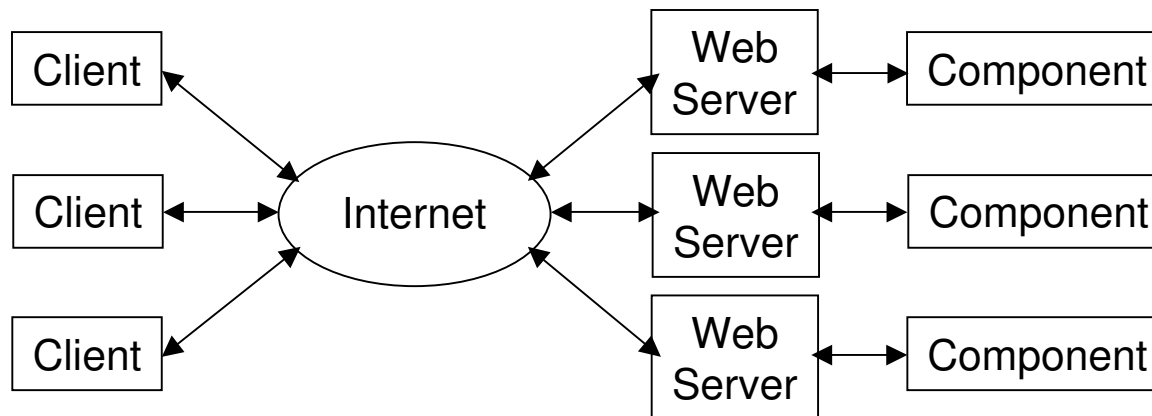
# Pipe Geography



# Software architecture: problems

- Combination of legacy code and new code
- Codes implemented in several languages (C++, Java, Python,...) and on several platforms (Linux, Wintel, IBM SP, ...)
  - Impractical to re-implement legacy code in a single language
  - Impractical to mandate single language for all new code
- Researchers distributed geographically

# Software architecture: ASP solution



- Components
  - exposed as **web services**
  - server located at appropriate developer site
- Workflow defined by client
- Communication
  - SOAP: industry-standard RPC API
  - XML-based data formats for geometry/mesh/...: Vavasis et al

# Why distributed components?

- Why components?
  - Modularity
  - Essential for adaptivity
    - Common, standard interfaces
    - Late binding of modules
- Why distributed?
  - Respect each developer's choice of language, compiler, and platform
  - “Write once, run *from* anywhere”
  - Software maintenance is easier
    - No need to port code to multiple platforms
    - No need to download/re-compile/re-link bug fixes or new releases of software

# Key problem: fault tolerance

- How does client specify parallelism in component invocation?
- How can we get transparent fault-tolerance for components?
- How does client specify what action to take when a component fails?
- How does a client itself survive faults?