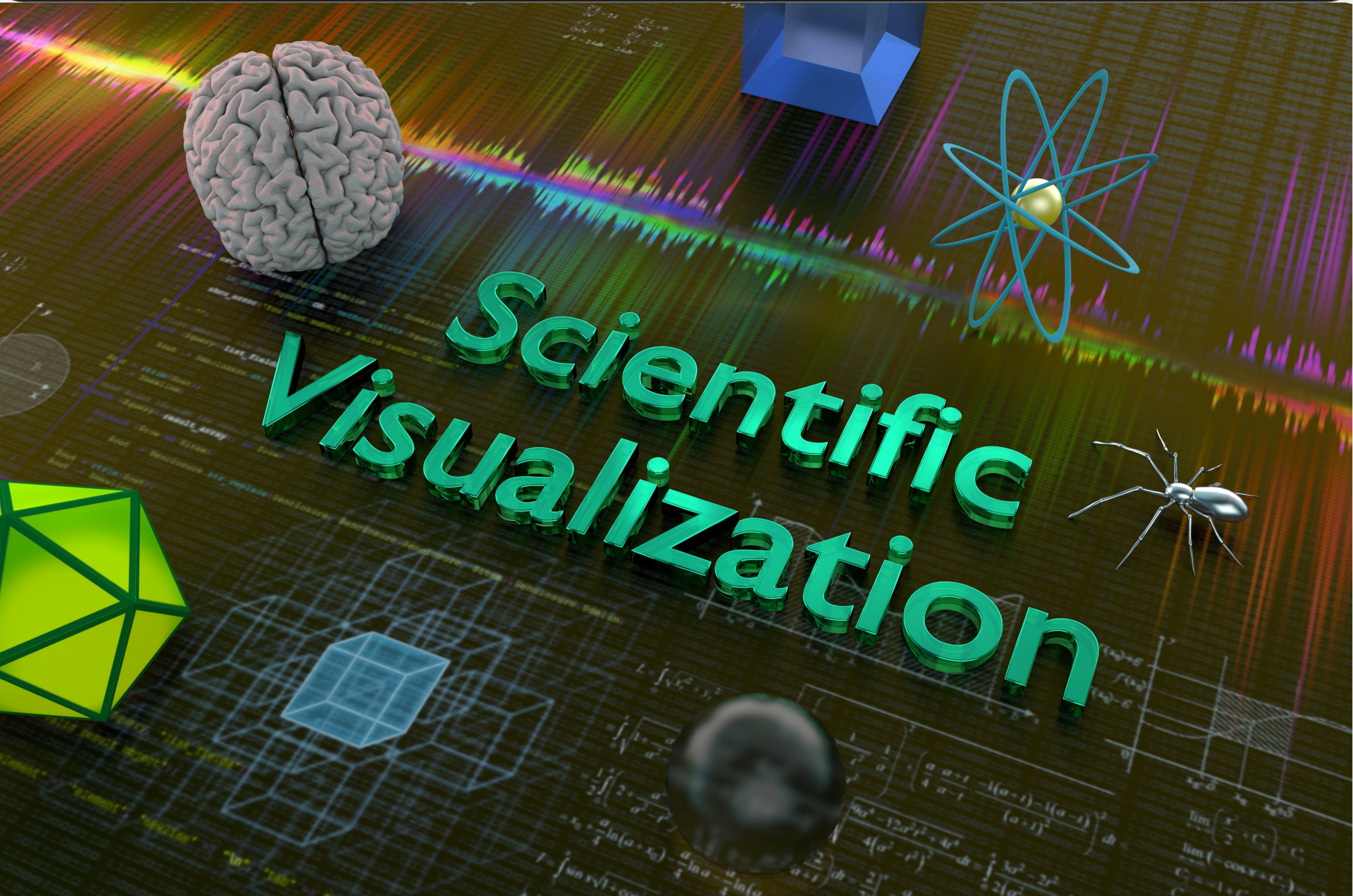
Development of Multiplatform Adaptive Rendering Tools to Visualize Scientific Experiments

Konstantin Ryabinin,
e-mail: kostya.ryabinin@gmail.com

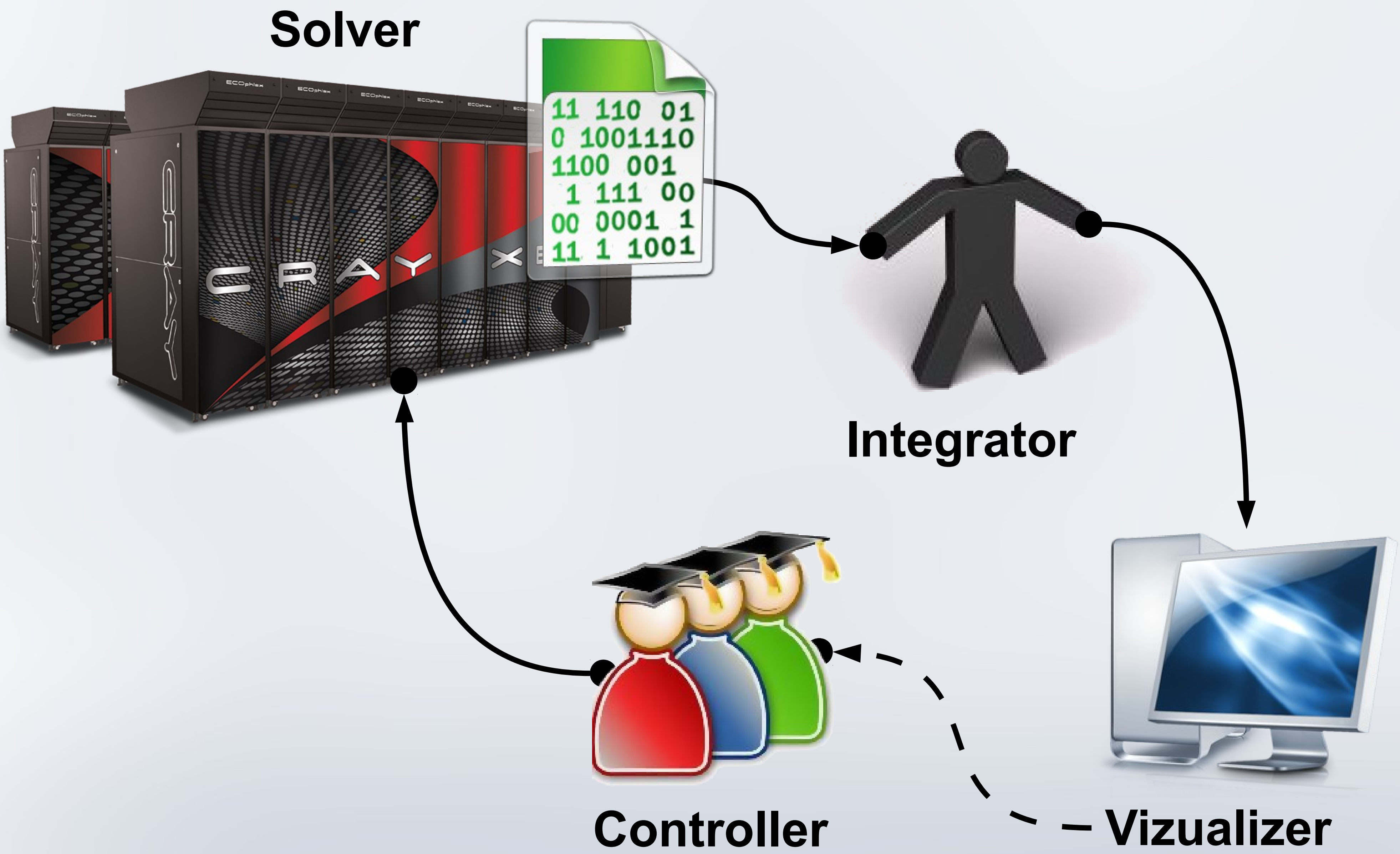
Svetlana Chuprina,
e-mail: chuprinas@inbox.ru

Perm State University,
Bukireva Str. 15, 614990, Perm, Russia

Cairns, 2014


Scientific
Visualization

Typical Visualization Data Flow



Visualization Software

MS Excel

OO Calc

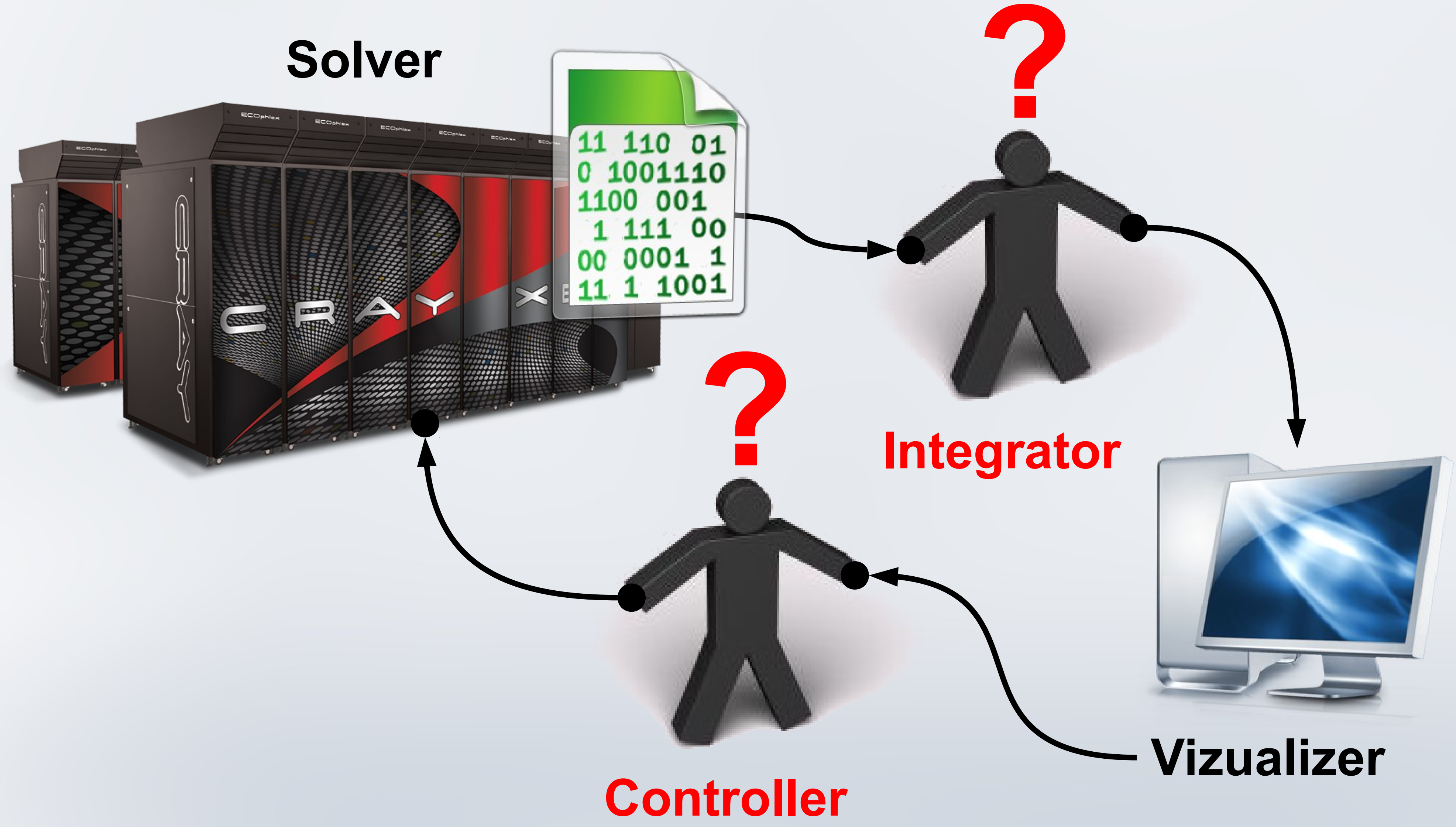
AutoCad

Common Problems (1)

- There are no high-level means to integrate solvers and visualizers
- There are no adequate multi-platform portability tools
- Implementation of distributed visualizers is inefficient (visualizers are not adaptive to software infrastructure)
- The quality of images is not high enough because of aliasing (jagged edges of objects)

Common Problems (2)

- There are no high-level means to integrate solvers and visualizers



Solver

Integrator

Vizualizer

Controller

Common Problems (3)

- There are no adequate multi-platform portability tools

Supercomputers

iOS

android

Mobile devices

Desktop computers

GNU / Linux

OS X

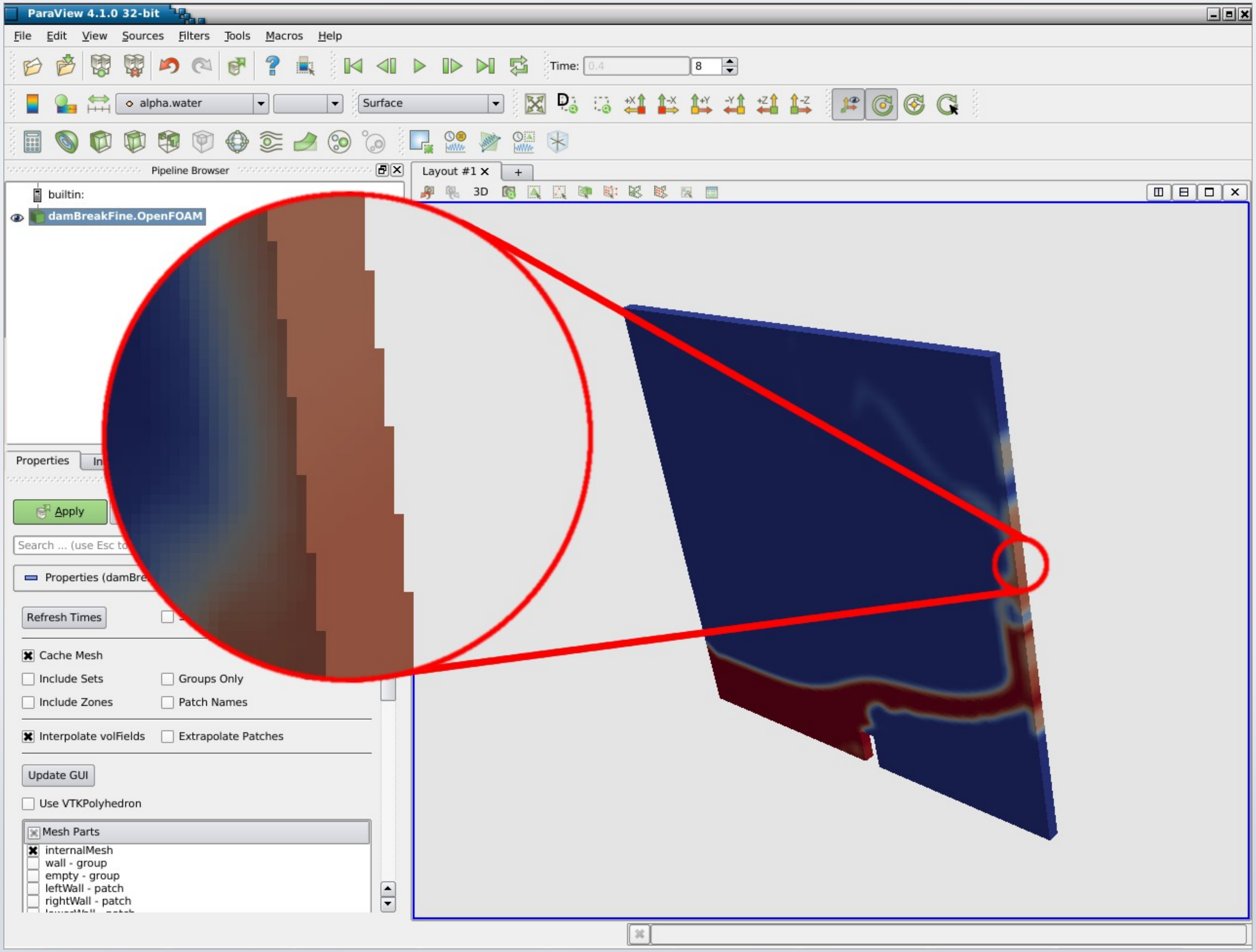
Windows

Common Problems (4)

- **Implementation of distributed visualizers is inefficient (visualizers are not adaptive to software infrastructure)**

Common Problems (5)

- **The quality of images is not high enough because of aliasing (jagged edges of objects)**



Time: 0.4 | 8

alpha.water | Surface

☒ Cache Mesh
☐ Include Sets ☐ Groups Only
☐ Include Zones ☐ Patch Names
☒ Interpolate volFields ☐ Extrapolate Patches
☐ Use VTKPolyhedron

☒ Mesh Parts
☒ internalMesh
☐ wall - group
☐ empty - group
☐ leftWall - patch
☐ rightWall - patch

Purpose

Development of the client-server scientific visualization system

Main features:

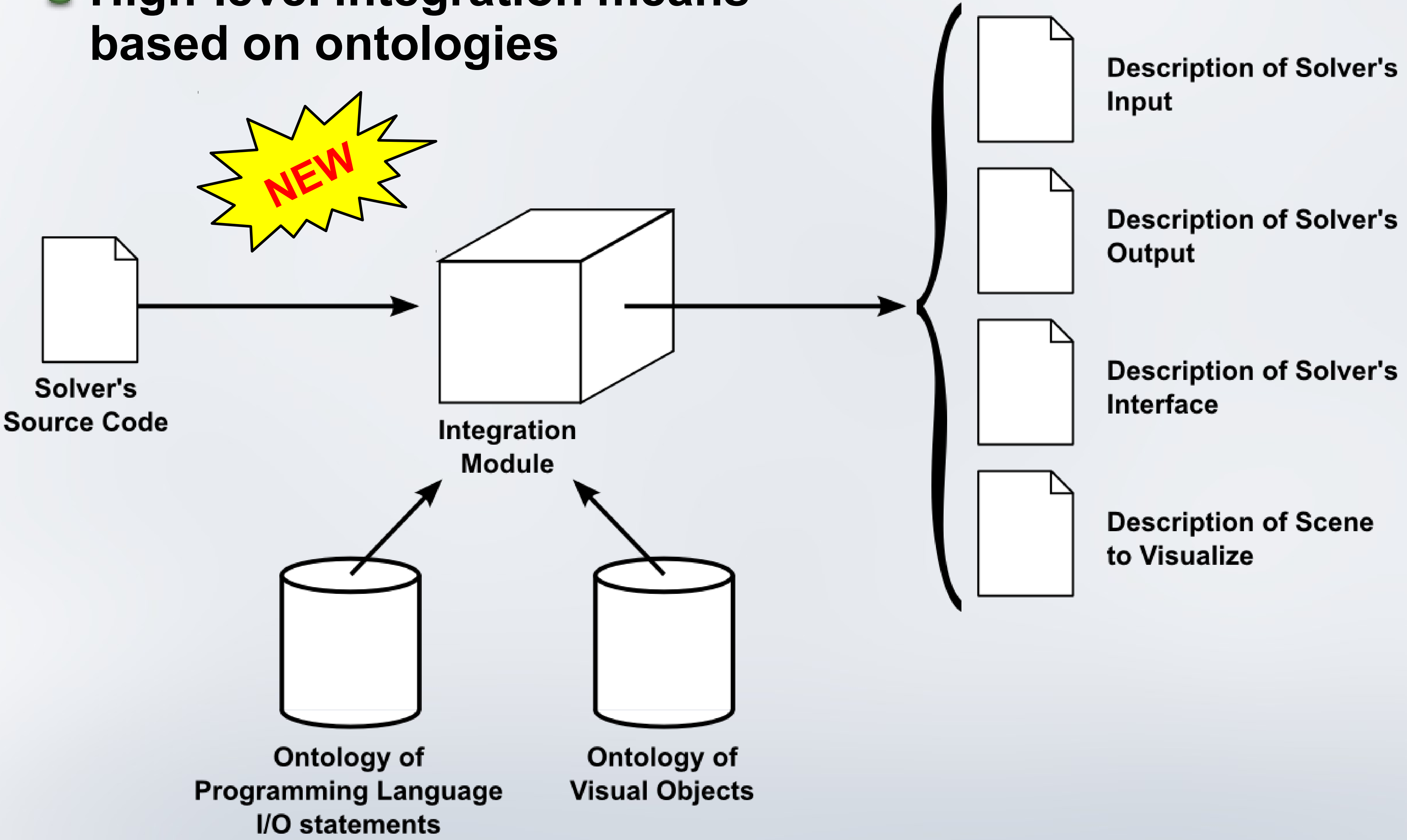
- **Automatic integration with different solvers**
- **Adaptive distribution of rendering between client and server**
- **Multiplatform portability**
- **Adaptive antialiasing**

Objectives

- To suggest new approach based on domain-specific ontologies to integrate different third-party solvers with visualization system
- To develop the frameworks' stack that helps to port visualization system to different platforms
- To design special heuristics for adaptive distribution of rendering between client and server
- To create new multi-platform adaptive anti-aliasing algorithm to increase quality of images
- To test the visualization system to solve real scientific tasks in different application domains

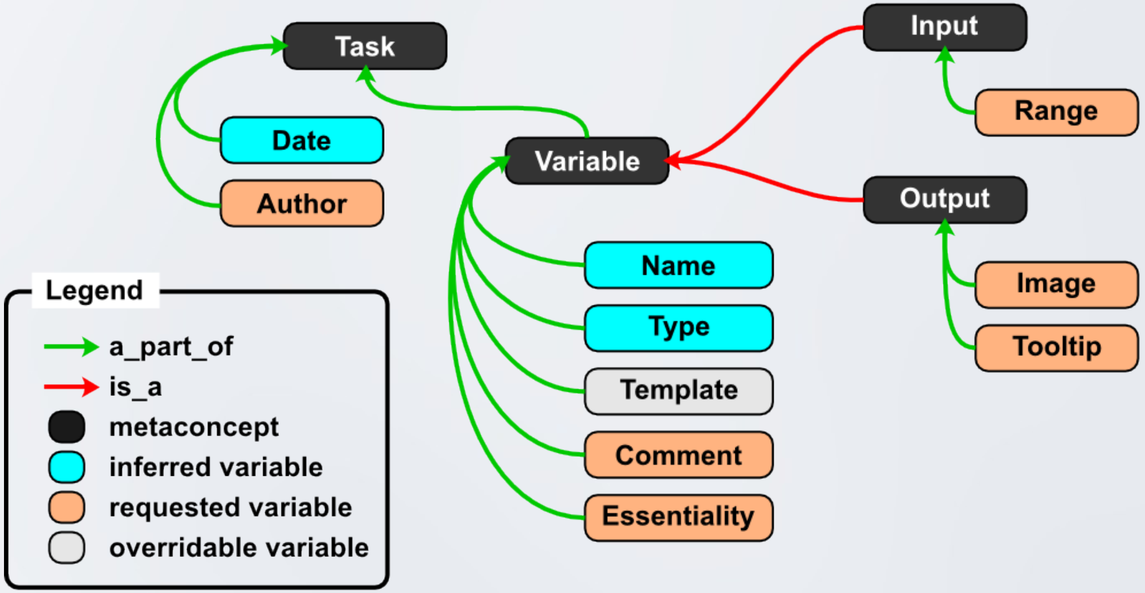
SciVi: Adaptation to Third-Party Solvers (1)

- **High-level integration means based on ontologies**


NEW
Solver's Source Code
Integration Module
Ontology of Programming Language I/O statements
Ontology of Visual Objects
Description of Solver's Input
Description of Solver's Output
Description of Solver's Interface
Description of Scene to Visualize

SciVi: Adaptation to Third-Party Solvers (2)

- Example of ontology built according to solver's source code:



Source Code Fragment

```
INTEGER  particlesCount,
         timeTickCount,
         timeTick,
         particleID
VEC3 orientation

READ*, particlesCount,
       timeTickCount
. . .
PRINT*, timeTick,
        particleID,
        orientation
```

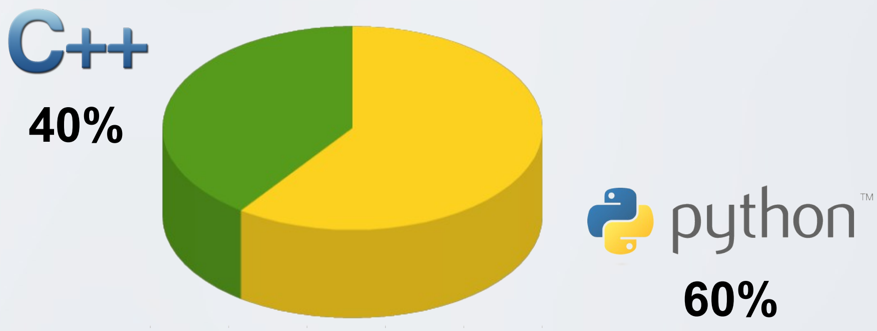
Ontology Fragment

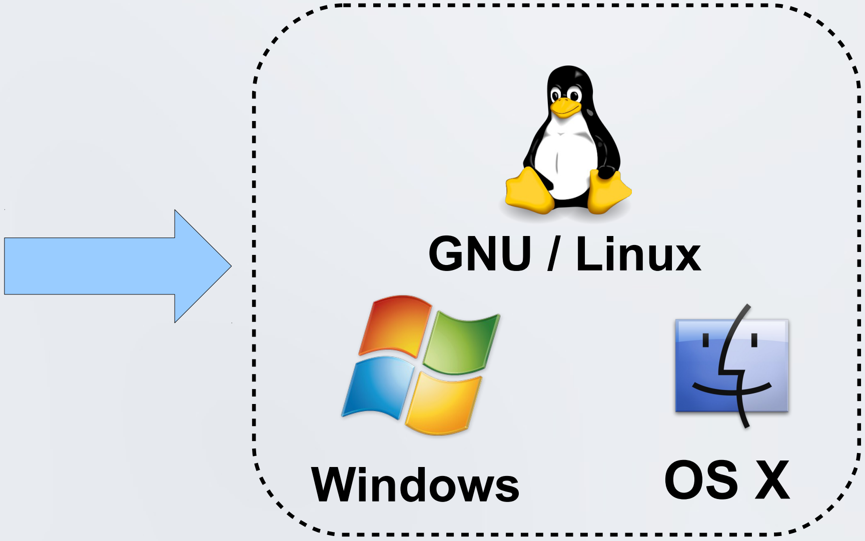
Input
☑ particlesCount
  type: int
  range: [1..100]
☑ timeTickCount
  type: int
  rage: [1..100]
Output
☑ timeTick
  type: int
  image: time
☑ particleID
  type: int
  image: particle.ply
☑ orientation
  type: vec3
  image: rotation

- Requirements that the solver should fulfill:
  - Single input and single output file
  - Type of objects evaluated by solver should not change during the experiment

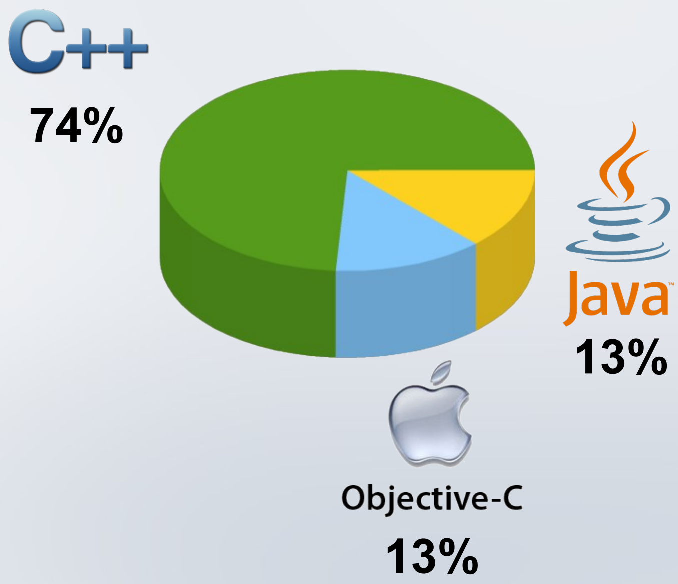
SciVi: Multiplatform Portability

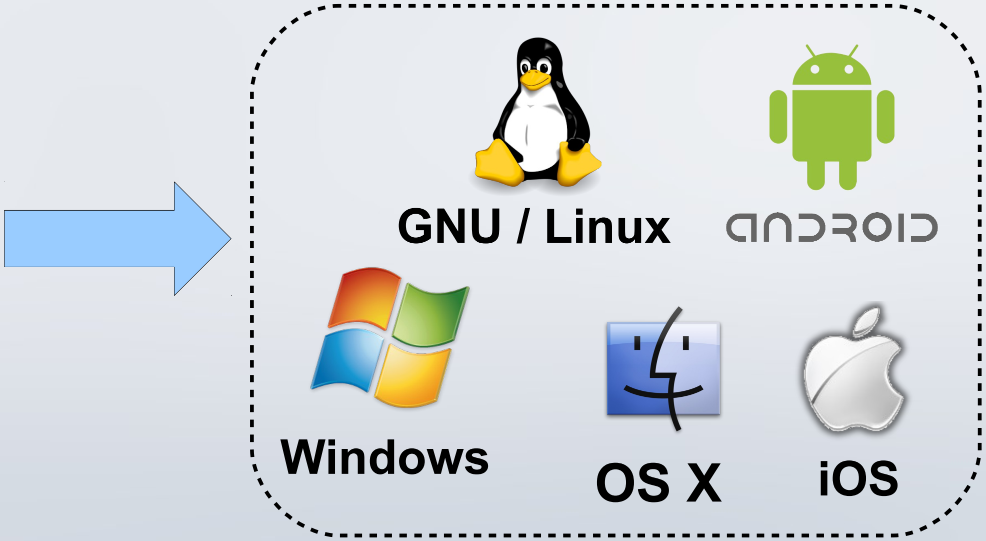
- SciVi server





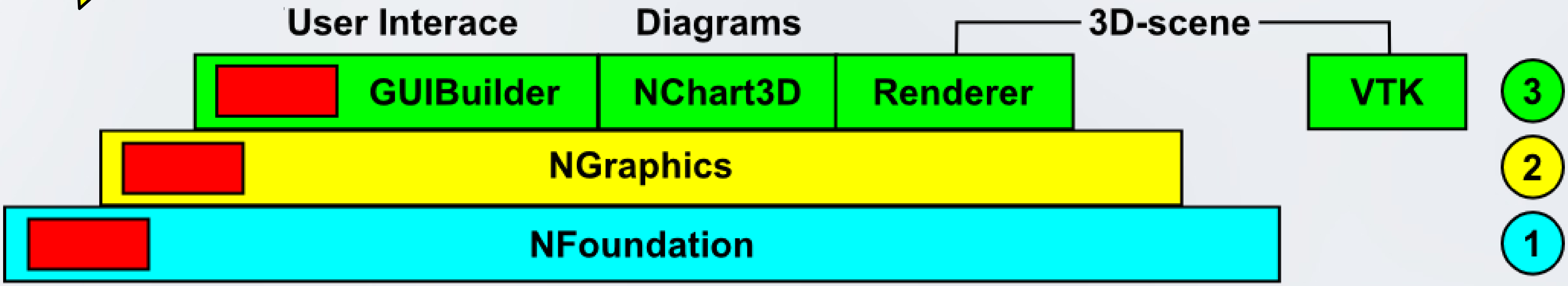
- SciVi client





SciVi: Stack of Frameworks

NEW



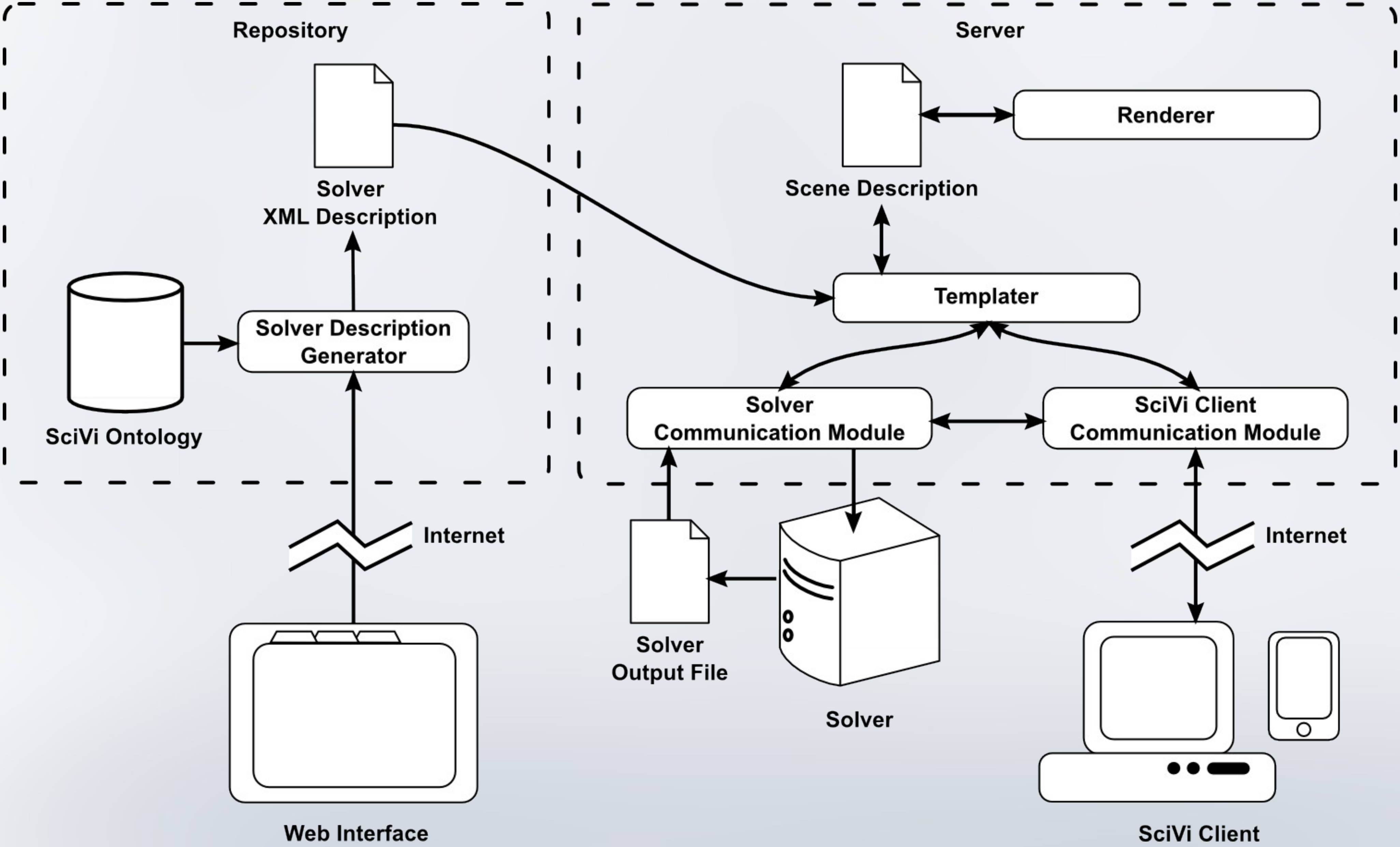
3 High-level visualization means

2 Low-level visualization means

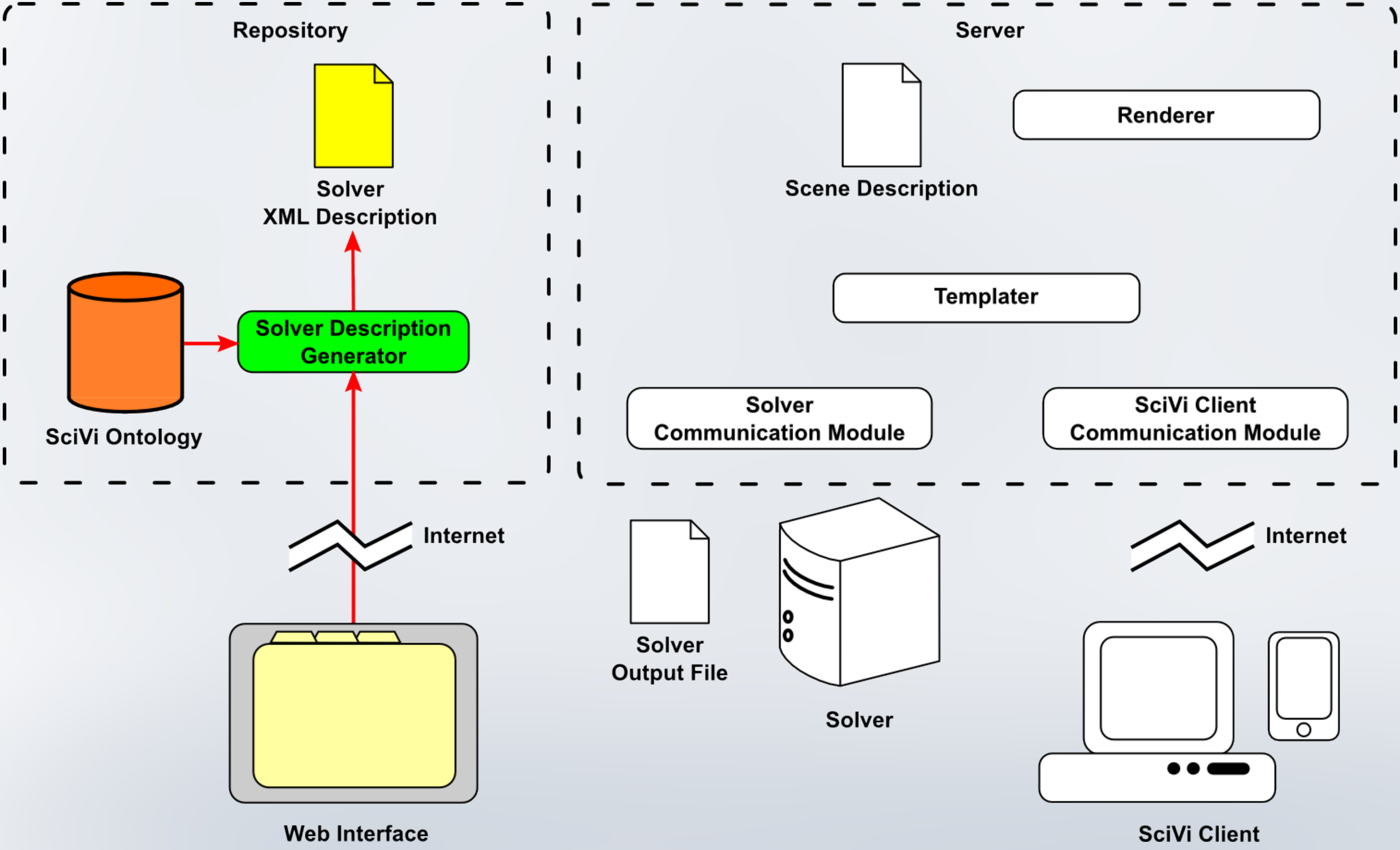
1 OS abstraction layer

🟥 System-dependent part

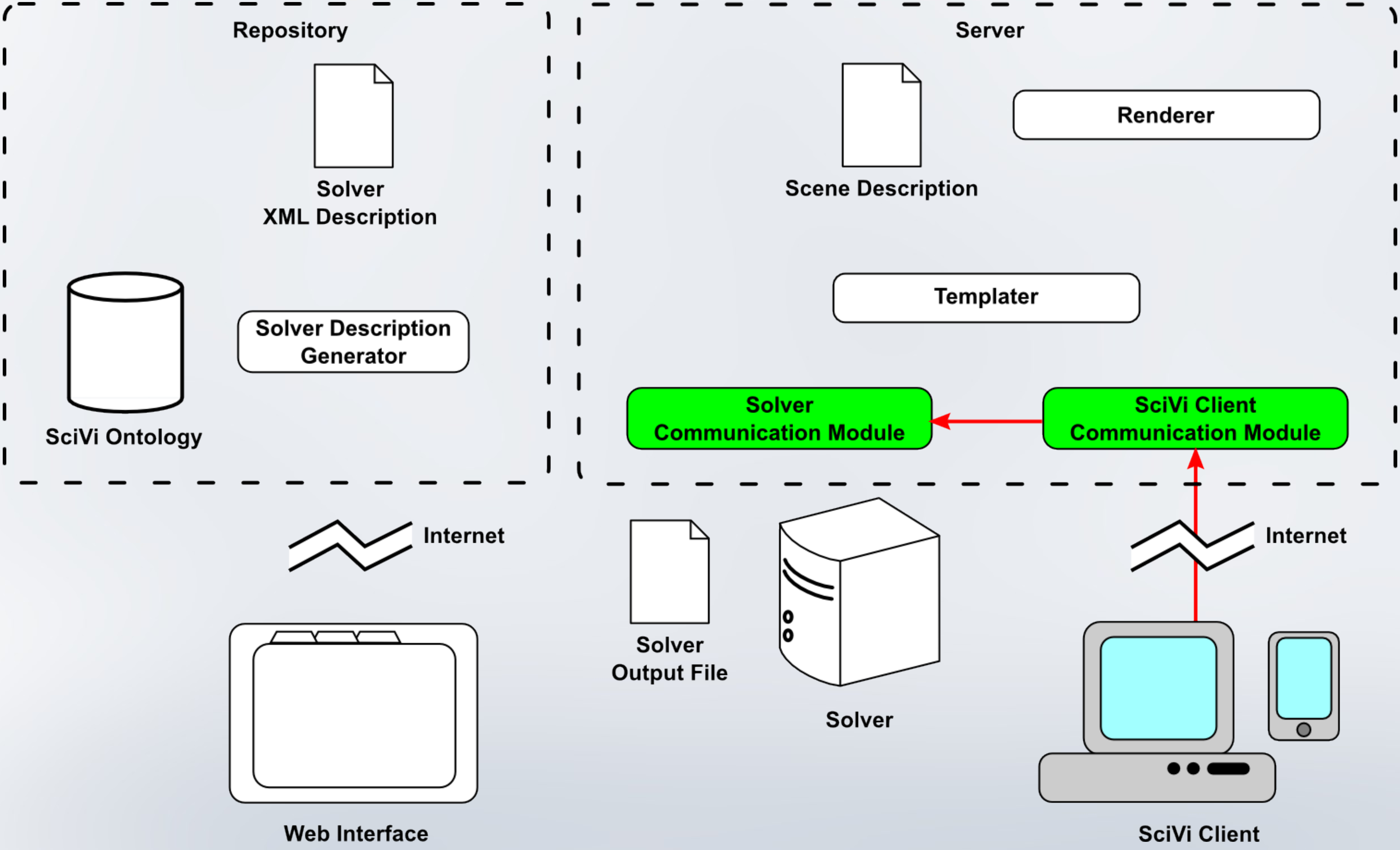
SciVi: Server Architecture (1)



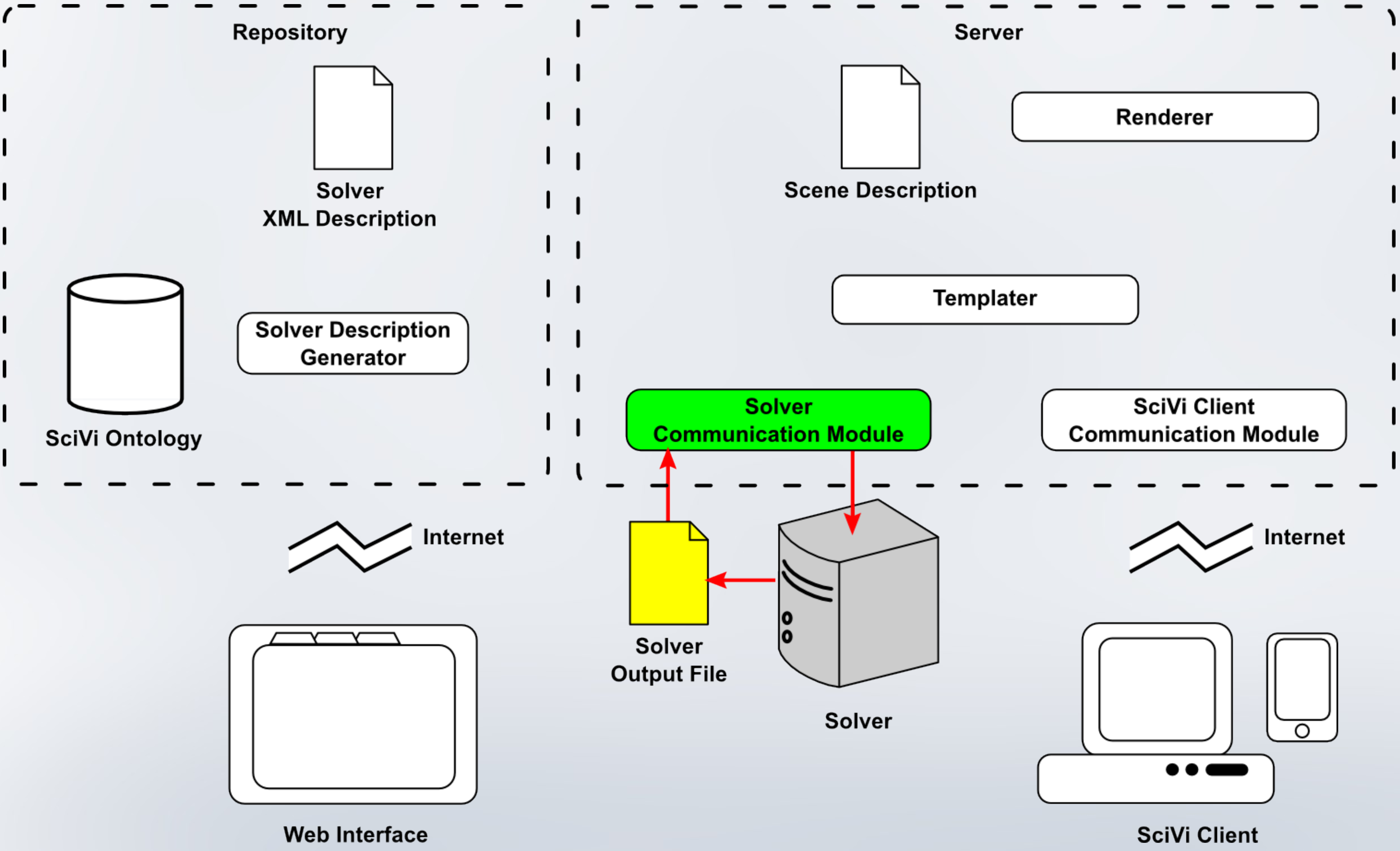
SciVi: Server Architecture (2)



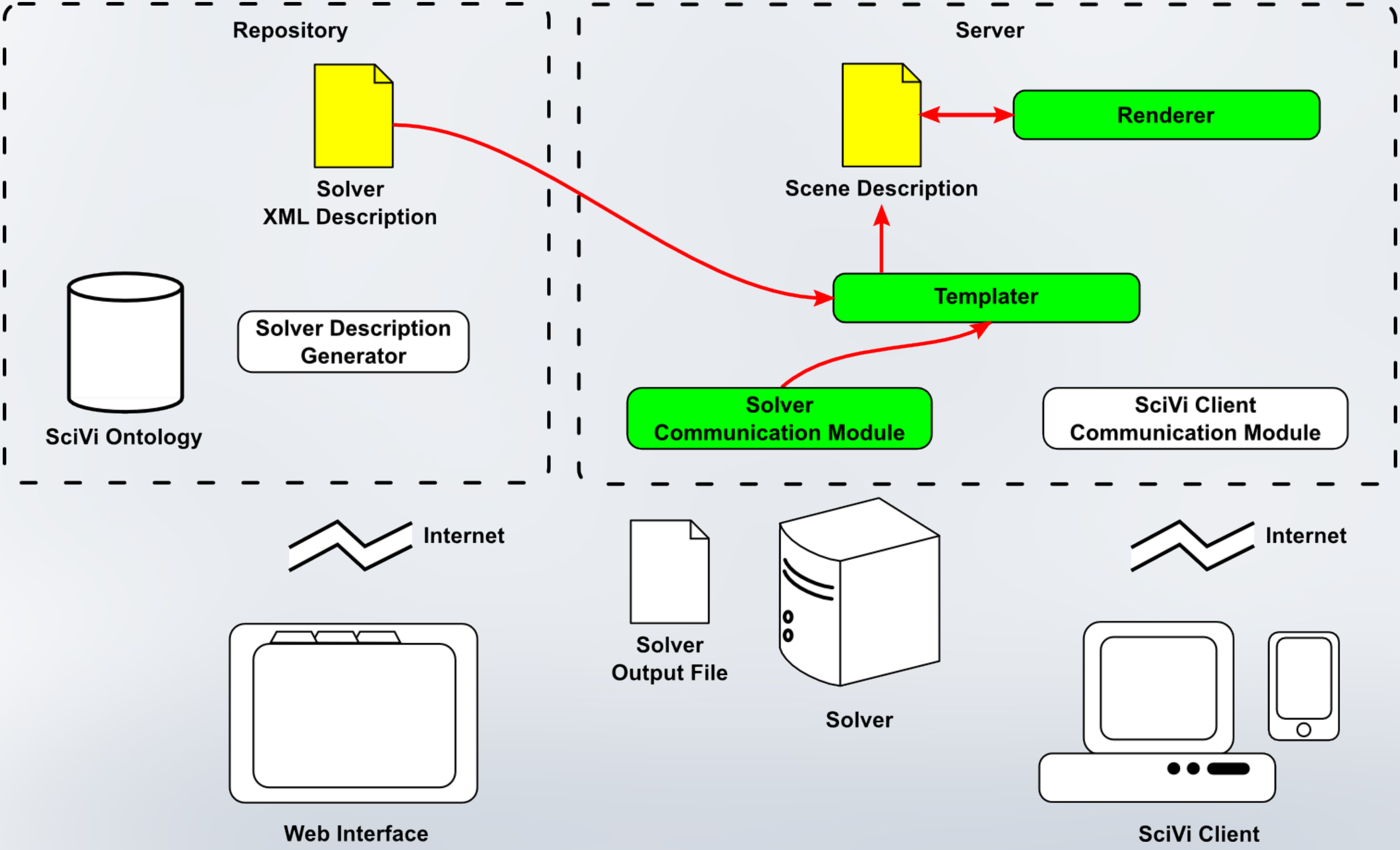
SciVi: Server Architecture (3)



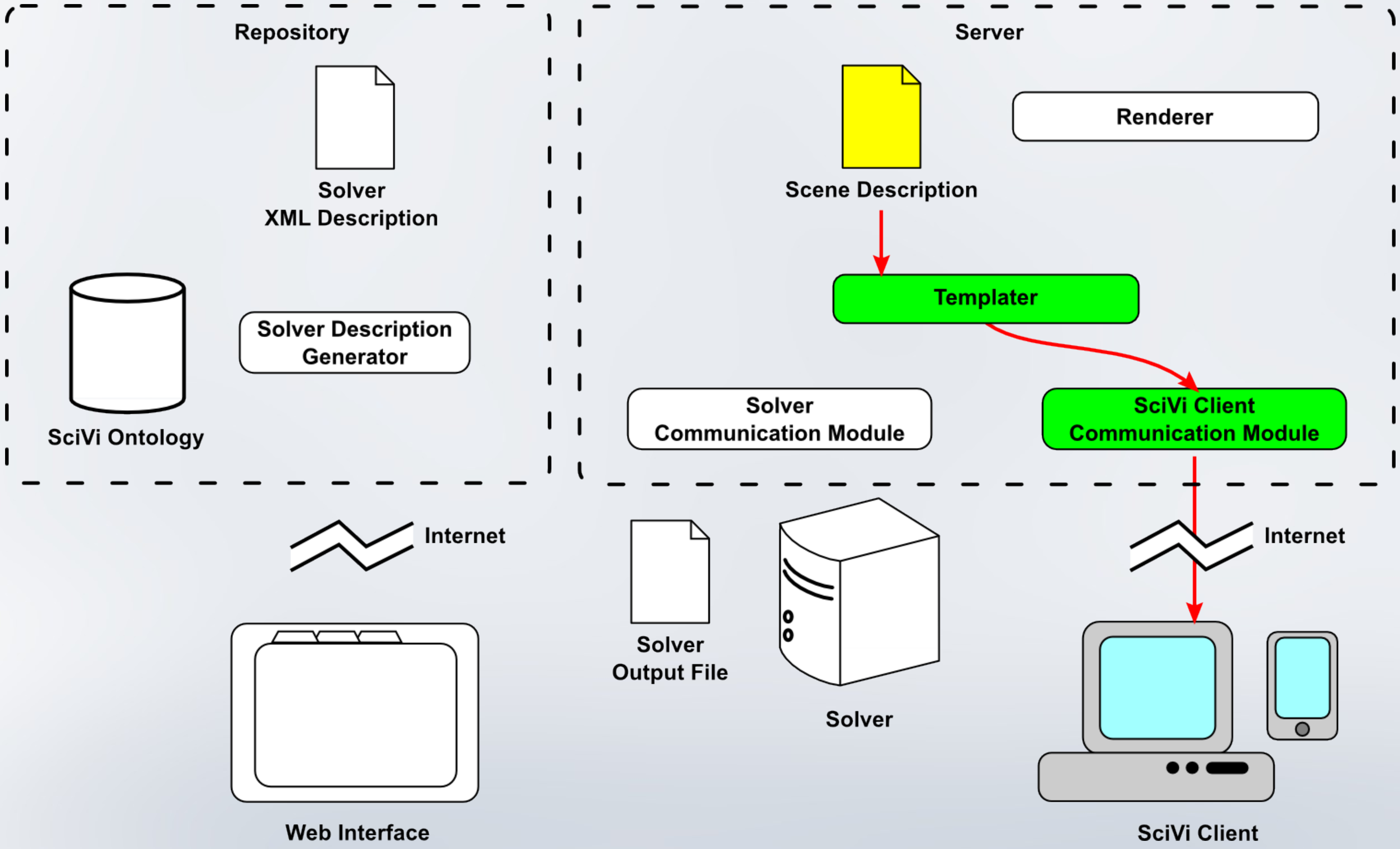
SciVi: Server Architecture (4)



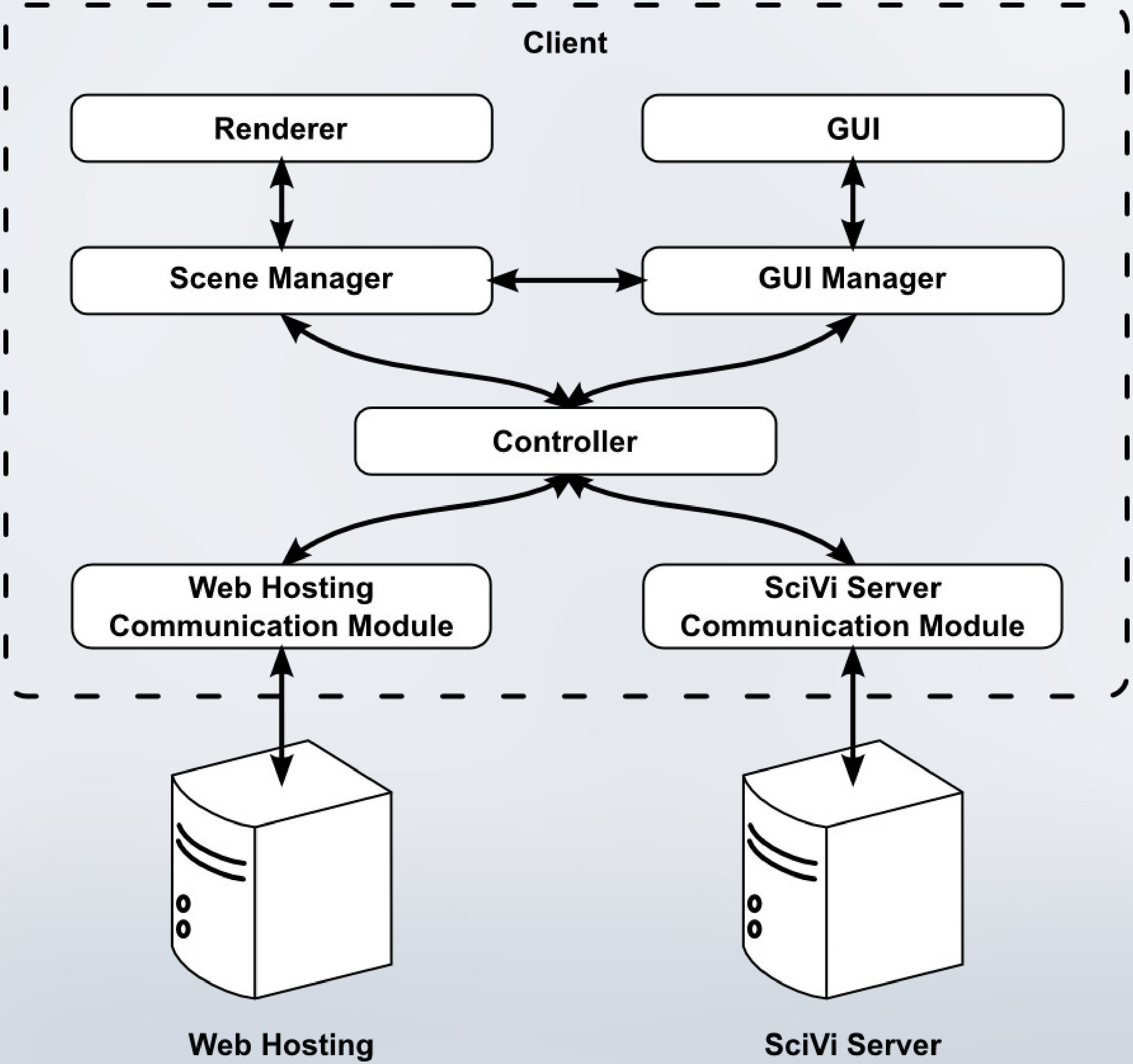
SciVi: Server Architecture (5)



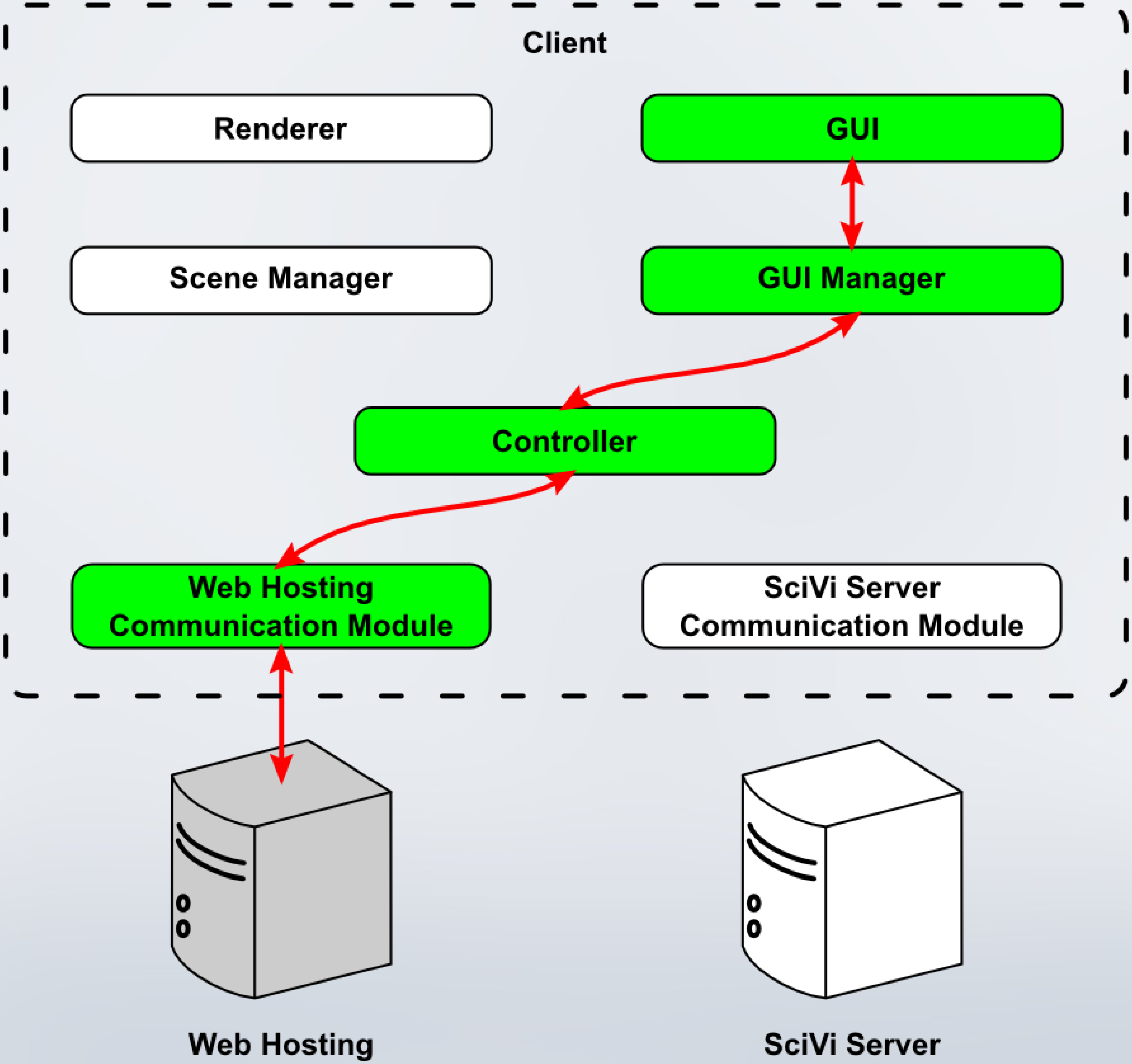
SciVi: Server Architecture (6)



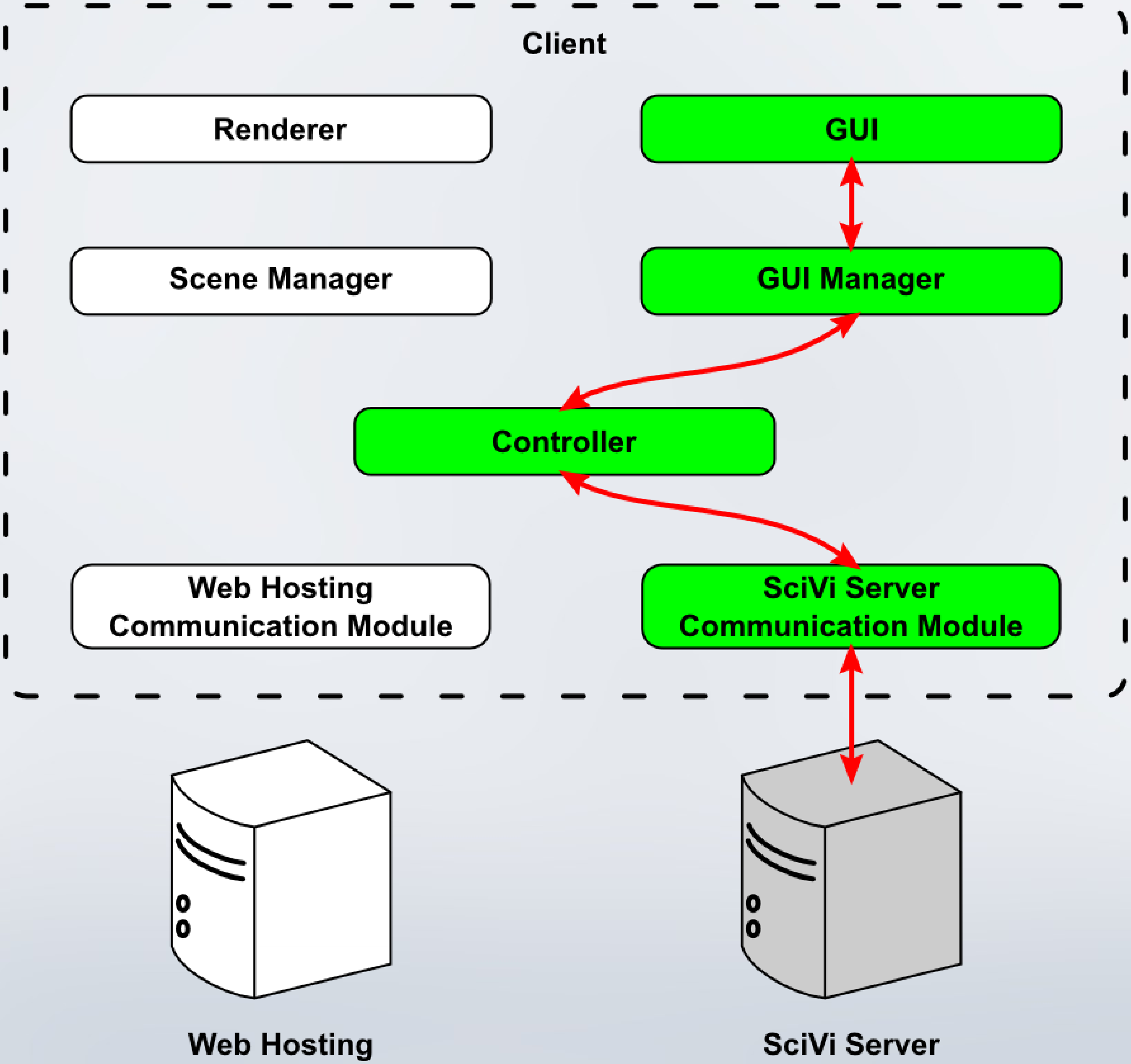
SciVi: Client Architecture (1)



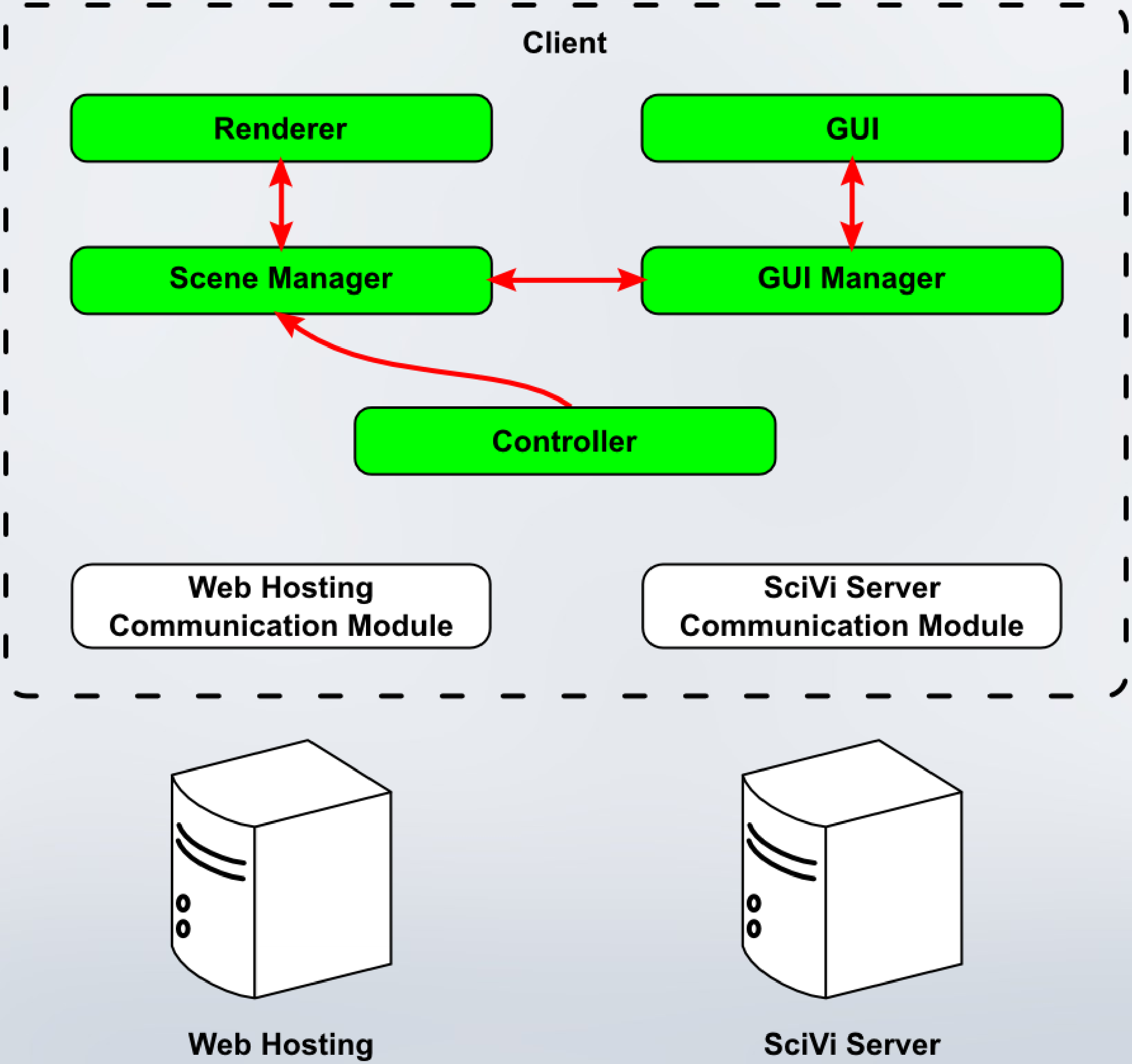
SciVi: Client Architecture (2)



SciVi: Client Architecture (3)

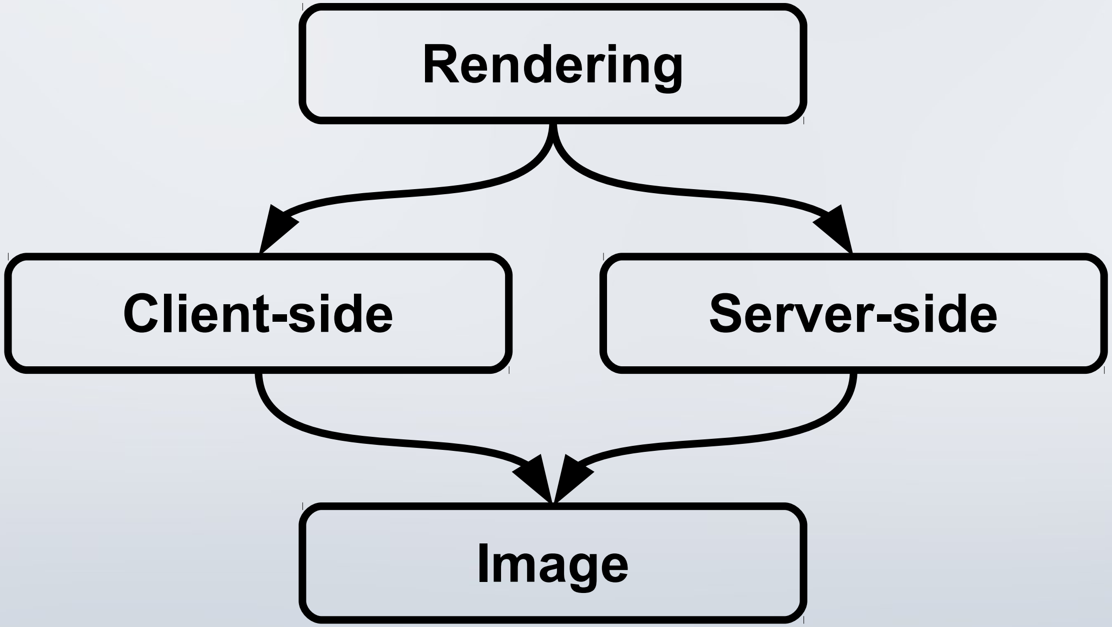


SciVi: Client Architecture (4)



SciVi: Adaptive Rendering Distribution (1)

- Server performs the adaptive distribution based on heuristics according to
  - Type and performance of the client
  - Speed of the network connection
  - Load of the server

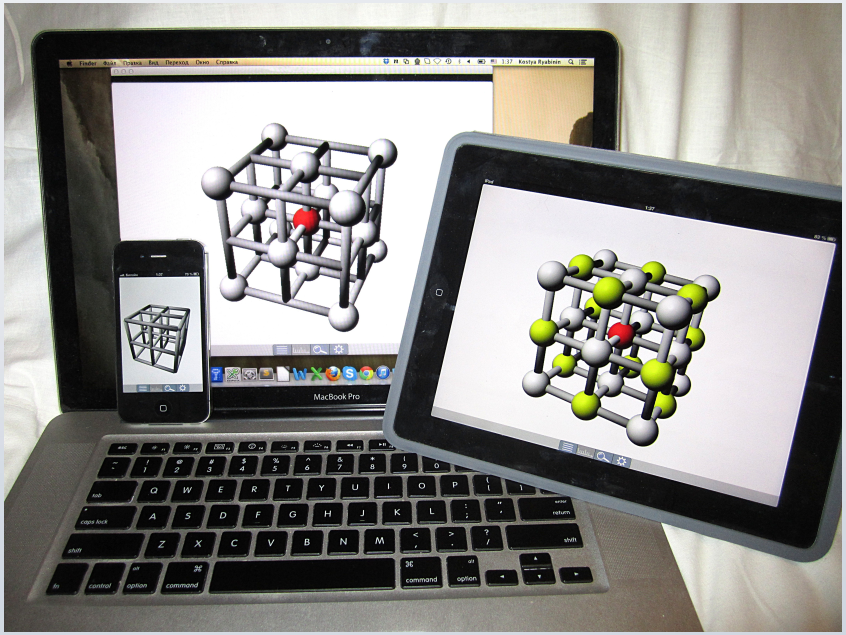


SciVi: Adaptive Rendering Distribution (2)

- Server's tasks:
  - Preparing the scene description according to the solver-specific template generated by integration module
  - Partial visualization of the data by pVTK library
  - Preprocessing and simplification of the data
  - Transfer the data to client
- Client's tasks:
  - Building interface for solver according to the description generated by integration module
  - Rendering of the final image

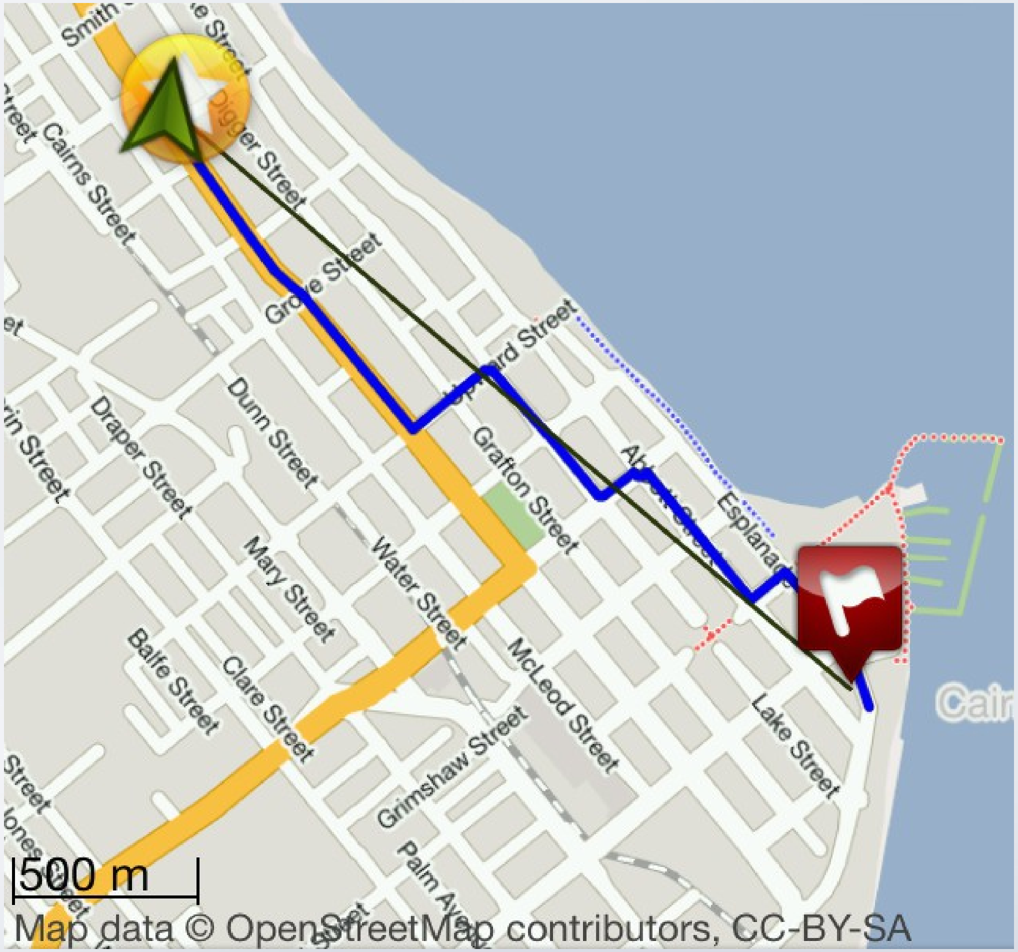
SciVi: Adaptive Rendering Distribution (3)

- Adaptive rendering distribution allows to
  - Enhance scene interactivity on the client-side
  - Optimize loading of heterogeneous computing system with
    - Different kinds of devices (desktop, tablet, smartphone)
    - Different types of network connections (Ethernet, WiFi, 3G, EDGE)

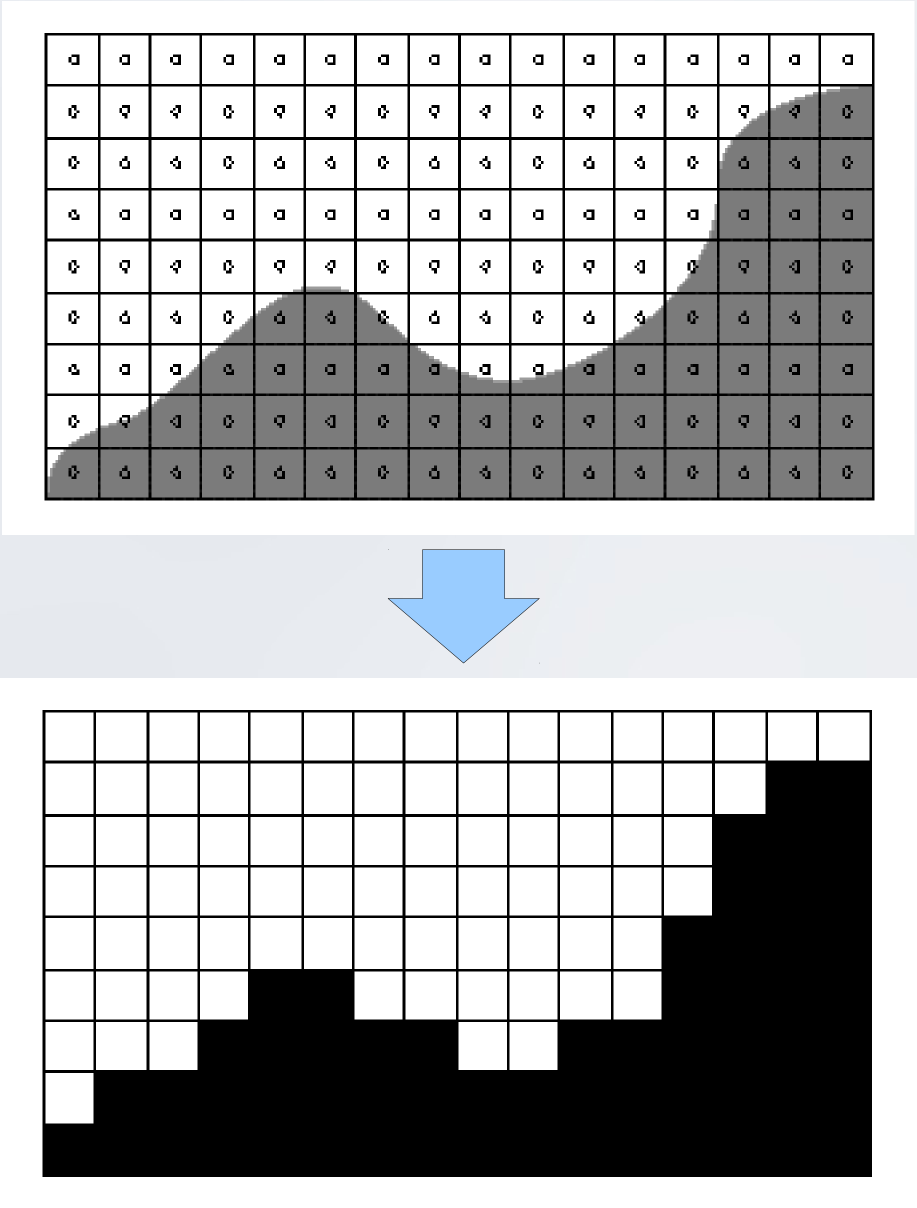


Aliasing Problem

- **Rendering is a discrete process => aliasing is inevitable**



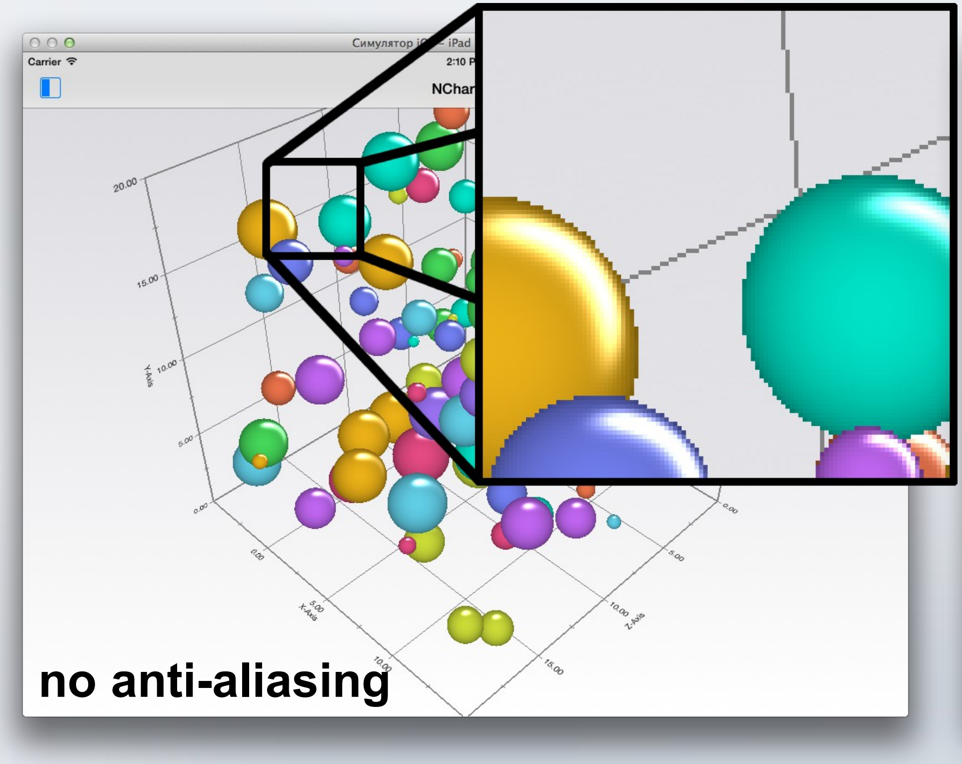
Map data © OpenStreetMap contributors, CC-BY-SA



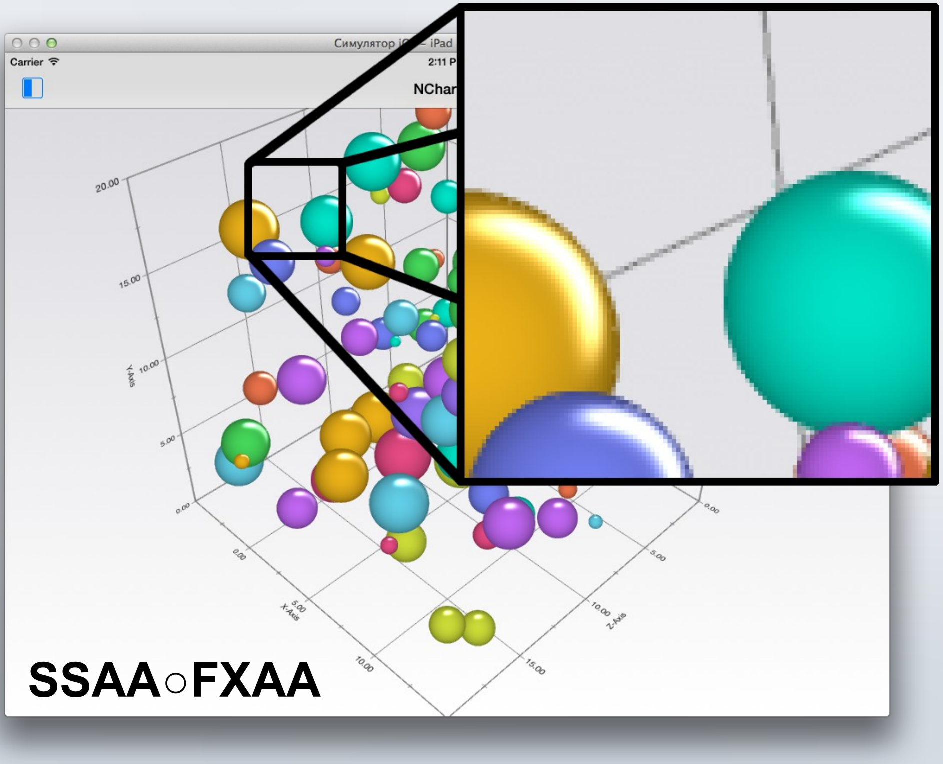
SciVi: Adaptive Anti-aliasing (1)

- Using our own algorithm to smooth jagged edges on the images:
  $A = SSAA \circ FXAA$
  $T(A) \sim O(n + (S^2 + 1) \cdot w \cdot h)$,
  - n – count of vertices on the scene
  - S – supersampling ratio
  - w, h – width and height of the screen

NEW



no anti-aliasing



SSAA○FXAA

SciVi: Adaptive Anti-aliasing (2)

- **Adaptivity of algorithm:**
  - **Supersampling ratio (*S*) adjustment to adapt to software and hardware properties**

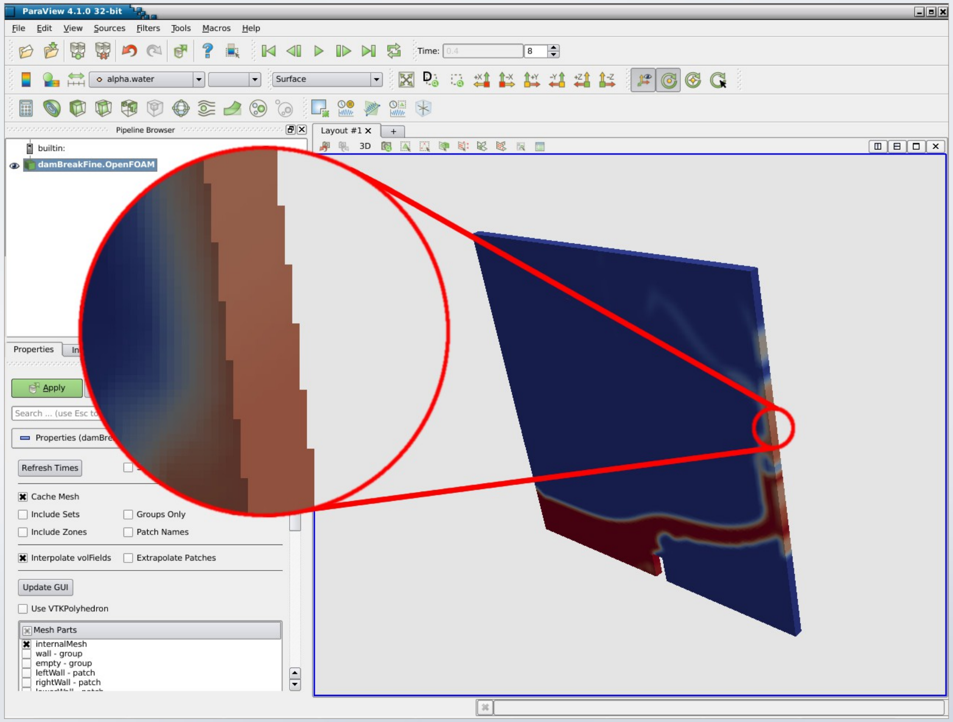
$$S = \begin{cases} \dfrac{w_{texture}}{w_{screen}}, w_{screen} > h_{screen} \\ \dfrac{h_{texture}}{h_{screen}}, w_{screen} \le h_{screen} \end{cases}, \quad S = \min\{\max\{S, 1\}, 2\}$$

  - **Automatic anti-aliasing on/off switching to ensure high performance during time-critical periods**
  - **Layered rendering to combine objects with and without anti-aliasing on the single scene**

SciVi: Adaptive Anti-aliasing (3)

- Implemented anti-aliasing algorithm allows to increase:
  - Visual quality of images
  - Performance of rendering

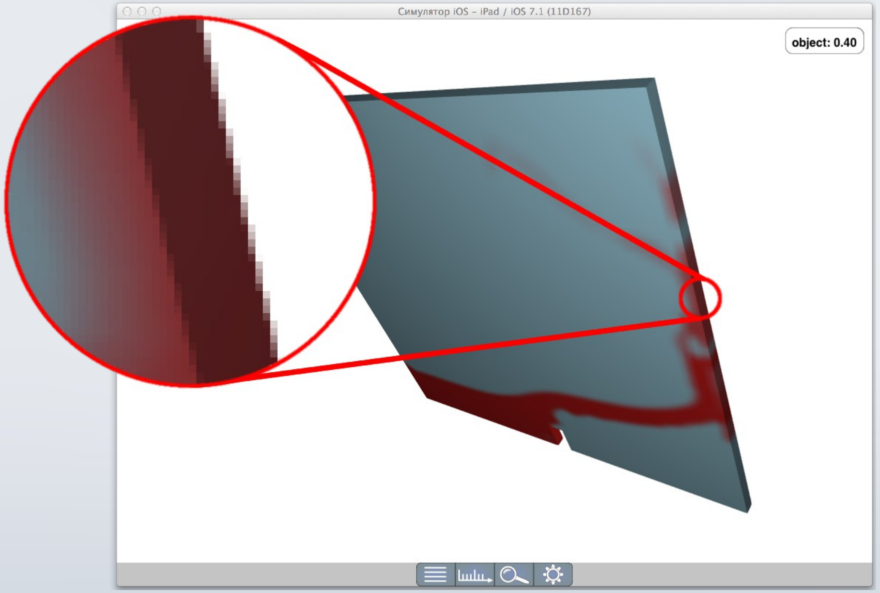
ParaView



☒ Cache Mesh
☐ Include Sets
☐ Groups Only
☐ Include Zones
☐ Patch Names
☒ Interpolate volFields
☐ Extrapolate Patches
☐ Use VTKPolyhedron

Mesh Parts
☒ internalMesh
☐ wall - group
☐ empty - group
☐ leftWall - patch
☐ rightWall - patch

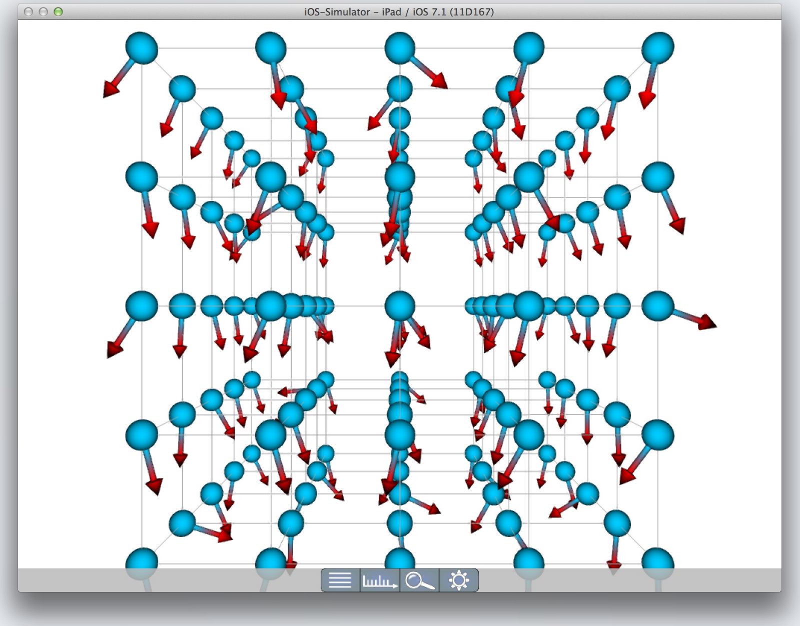
SciVi

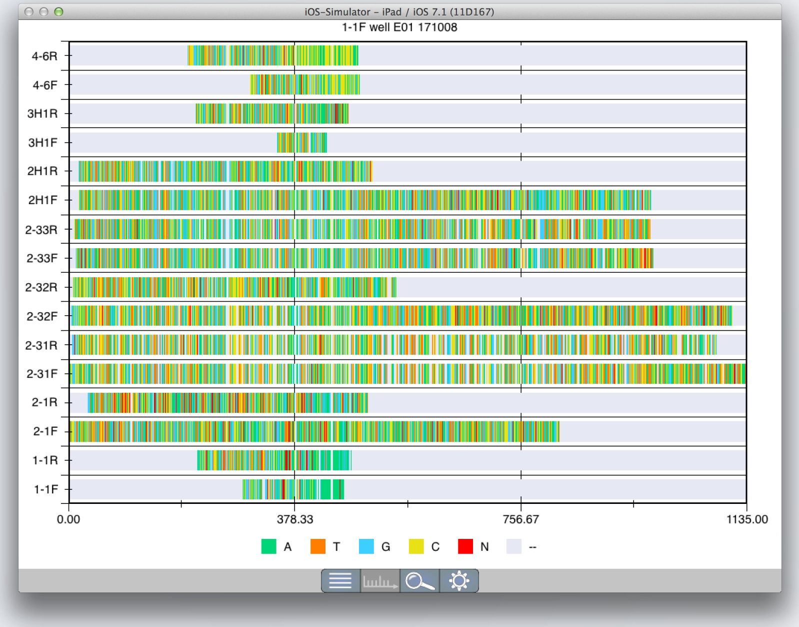


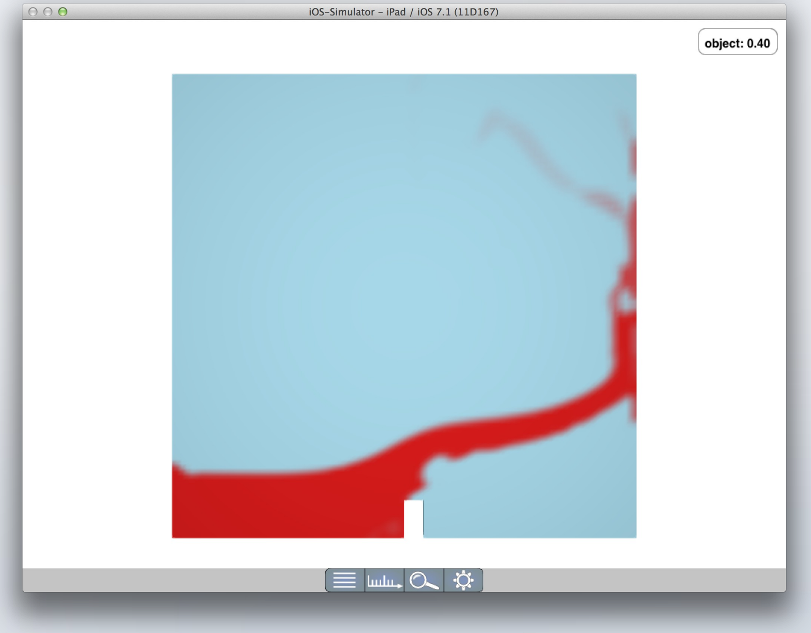
SciVi: Testing (1)

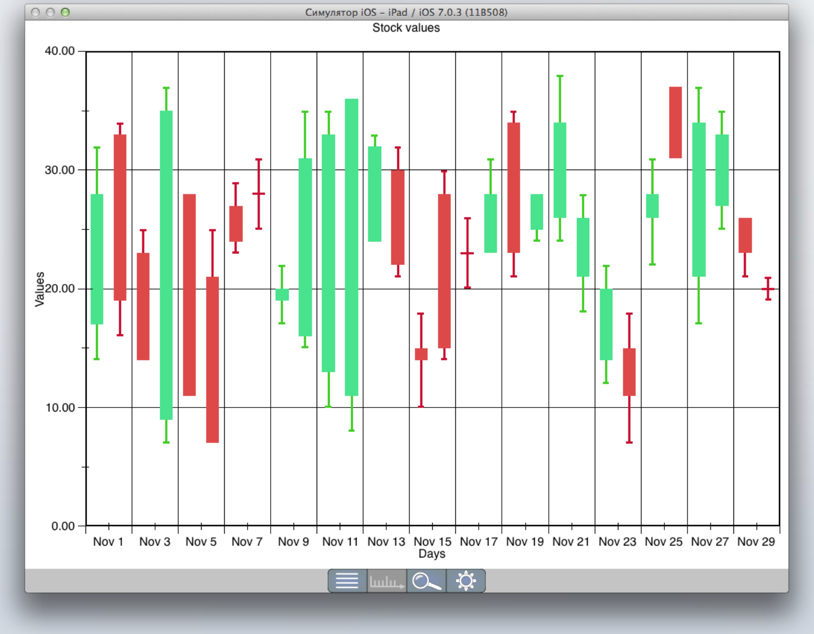
- We tested SciVi on the following real scientific tasks:
  - Modelling of nano-particles magnet momenta orientation in the magnetic field
    Solver: MagnetoDynamics-F
    Language: Fortran
  - Multiple alignment of DNA sequences
    Solver: ClustalW
    Language: C++
  - Fluid simulation
    Solver: OpenFOAM
    Language: C++
  - Changes of prices on BTC-e stock market
    Solver: Application that monitors BTC-e
    Language: Java

SciVi: Testing (2)









Example: MagnetoDynamic-F (1)

- **Solver has been developed in Perm State University**
- **Solver runs on supercomputer and produces output looking like**

```
0
0.440798910779106 0.193456073216762 -0.876510734669864
0.0480228711779573 -0.284014985805231 -0.957616463769227
0.36786492397633 0.792218280706476 -0.486893821507692
-0.114873074685615 0.0755407821401694 -0.990503794513643
. . .
0.35
0.501727936749538 -0.00133378223902795 -0.865024449660234
-0.00346408984151143 -0.272264254539676 -0.962216283265631
0.628307647042774 0.620768861805486 -0.468908862019364
-0.0396858270658191 0.11017684776047 -0.993119377188698
. . .
```

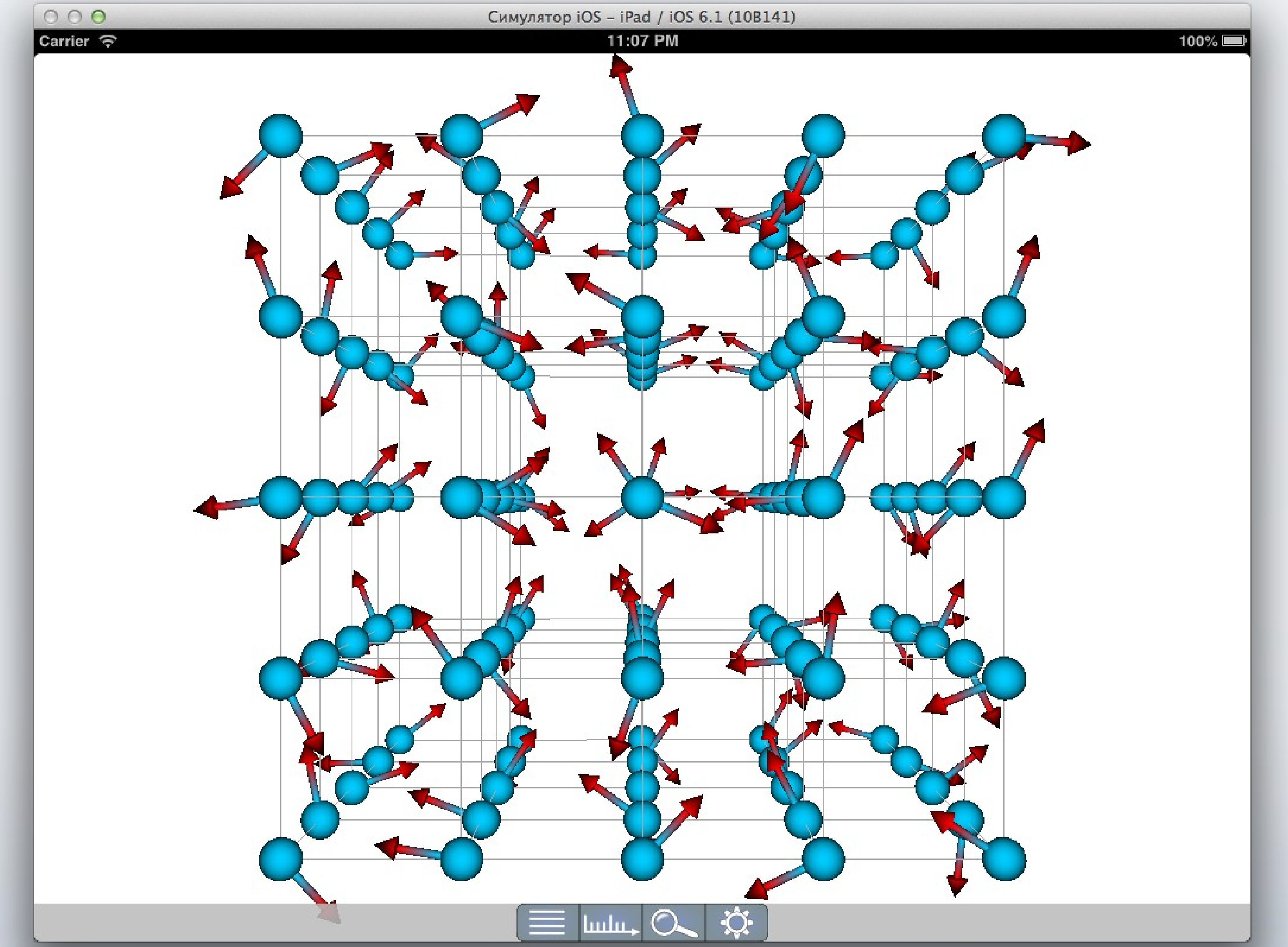
Example: MagnetoDynamic-F (2)

- **Server transforms the output data to the XML-description of scene, acceptable for SciVi**

```
<model id="spin_0">
  <data model="http://dl.dropbox.com/u/71028668/scivi/spins/spin.n3d"
        shader="blinn_vc"/>
  <position x="-0.750000" y="-0.750000" z="-0.750000"
            scaleX="0.045" scaleY="0.045" scaleZ="0.045"
            dirX="0.440799" dirY="0.193456" dirZ="-0.876511"/>
  <animation>
    <timestamp id="1">
      <position dirX="0.501728" dirY="-0.001334" dirZ="-0.865024"/>
    </timestamp>
    <timestamp id="2">
      <position dirX="0.492641" dirY="-0.206796" dirZ="-0.845305"/>
    </timestamp>
    <timestamp id="3">
      <position dirX="0.428445" dirY="-0.370627" dirZ="-0.824057"/>
    </timestamp>
    . . .
  </animation>
</model>
. . .
```

Example: MagnetoDynamic-F (3)

- **The result of client-side visualization**



SciVi and Parallel Computing

- Scientific data are usually Big Data
- SciVi is ready to meet Big Data because of scalability and flexibility
- The solver can be a parallel program
- The image is partially rendered on the server-side using pVTK which ensures efficient parallel rendering

Conclusion

- We developed scientific visualization system SciVi which provides:
  - Ontology-based automated integration with third-party solvers
  - Multi-platform portability
  - Adaptive distribution of rendering
  - Adaptive anti-aliasing
- SciVi was successfully tested on the different real scientific tasks
- SciVi allows the researchers in different branches of science to collaborate and to share their results

Future Plans

- To extend SciVi ontologies
- To enhance heuristics used for rendering distribution
- To implement additional data preprocessing algorithms on the server-side
- To implement additional rendering algorithms on the client-side
- To research the server-scaling in the context of Big Data

Thank you for your attention!

Konstantin Ryabinin,
e-mail: kostya.ryabinin@gmail.com

Svetlana Chuprina,
e-mail: chuprinas@inbox.ru

Perm State University,
Bukireva Str. 15, 614990, Perm, Russia

Cairns, 2014