# OGF Components for Grid Visualization Systems – Status and Outlook

Andre Merzky – SAGA RG/WG

# Overview

- OGF visualization activities
- Related OGF activities
- Next steps
- Missing pieces

# OGF Visualization activities

- **Advanced Collaborative Environments-RG**
  - GFD.43: *"Security Requirements of Advanced Collaborative Environments"*
    - describes the security requirements of a broad range of advanced collaborative and/or visualization environments
    - The goal of this paper is to highlight security issues that are important to the community
  - No follow-up activities of ACE
  - Most group members left OGF
  - Uptake of listed requirements limited (AFAIK)

- **No other groups focused on viz or collaboration**

# Related OGF Activities

- Centered on data and resource management
- GridFTP/DAIS/(ByteIO)/REP/OREP/DFDL
  - Management of and access to data
- GRAAP/JSDL/BES
  - Resource and job management
- GHPN/FI/AUTHN/AUTHZ
  - Networking and security
- SAGA
  - API / application level activities

# Related OGF Activities

- DAIS and DFDL operate on Data Structures
  - DFDL:
    - Data Format and Description Language
    - Description of data structures in files
  - DAIS:
    - Data Access and Integration Service
    - Access to structured data (SQL, XML, …)

# Related OGF Activities

- GridFTP and ByteIO perform data transfer
  - GridFTP:
    - File oriented, but also used for streams
    - Extensible, striped, parallel transfer, …
  - ByteIO:
    - Focused on small data
    - Actually out of scope for visualization

# Related OGF Activities

- OGSA / BES / JSDL / HPCP / …
  - provide service management

    (service == component)
  - Allow to manage lifetime, deployment, AuthN/AuthZ, discovery etc.
  - No QoS for interconnection (?)
  - Implementations available and interoperable

# Related OGF Activities

- SAGA
  - Received a number of viz use cases (5)
  - Notifcation and async operations are part of the API look & feel
  - Stream API package includes
  - Message API package planned

    (focuses large binary messages)

# SAGA Message API

```
saga::endpoint ep  (url);
saga::message  msg (buf, len);

ep.connect (url);
ep.serve   ();

ep.send (msg);

saga::message msg2 = ep.receive ();
```

# Missing Pieces (SAGA)

- Component discovery
- Endpoint discovery
- Message syntax (data format/structure)
- Message semantics (data model)

- Plan for a layered model

# My opinion :-)

- Vizpipe is a distributed component system

  Grids are distributed environments
- OGF can help to provide infrastructure
  - Provisioning / deployment
  - Security
  - Communication channels
- OGF should **not** approach
  - Data formats
  - Data models
  - (OGF should allow for domain specific ones)

# What are the next steps

- Need to understand requirements of viz use cases better (scope, performance, communication pattern, reliability, …)
- `profile` the OGSA based solutions to viz use cases?
- Expand SAGA scope?

- What is needed on Service/MW level??