# Resource Usage Service

## OGF20 Working Group Session

G. Netzer, R. Piro, M. Riedel, X. Chen

# OGF IPR Policies Apply

- "I acknowledge that participation in this meeting is subject to the OGF Intellectual Property Policy."
- Intellectual Property Notices Note Well: All statements related to the activities of the OGF and addressed to the OGF are subject to all provisions of Appendix B of GFD-C.1, which grants to the OGF and its participants certain licenses and rights in such statements. Such statements include verbal statements in OGF meetings, as well as written and electronic communications made at any time or place, which are addressed to:
  - the OGF plenary session,
  - any OGF working group or portion thereof,
  - the OGF Board of Directors, the GFSG, or any member thereof on behalf of the OGF,
  - the ADCOM, or any member thereof on behalf of the ADCOM,
  - any OGF mailing list, including any group list, or any other list functioning under OGF auspices,
  - the OGF Editor or the document authoring and review process
- Statements made outside of a OGF meeting, mailing list or other function, that are clearly not intended to be input to an OGF activity, group or function, are not subject to these provisions.
- Excerpt from Appendix B of GFD-C.1: "Where the OGF knows of rights, or claimed rights, the OGF secretariat shall attempt to obtain from the claimant of such rights, a written assurance that upon approval by the GFSG of the relevant OGF document(s), any party will be able to obtain the right to implement, use and distribute the technology or works when implementing, using or distributing technology based upon the specific specification(s) under openly specified, reasonable, non-discriminatory terms. The working group or research group proposing the use of the technology with respect to which the proprietary rights are claimed may assist the OGF secretariat in this effort. The results of this procedure shall not affect advancement of document, except that the GFSG may defer approval where a delay may facilitate the obtaining of such assurances. The results will, however, be recorded by the OGF Secretariat, and made available. The GFSG may also direct that a summary of the results be included in any GFD published containing the specification."

- OGF Intellectual Property Policies are adapted from the IETF Intellectual Property Policies that support the Internet Standards Process.

# Agenda

- Brief introduction to RUS (5”)
- Core RUS Specification Comments (55”)
  - XPath expression restrictions
  - IDL & WS-I/WS-RF rendering status
  - Batch processing
  - Audit Information and <RecordHistory>
  - Handling of Mandatory Elements
- Requirements for Advanced RUS (20”)
- Project Presentations (10”)

# Brief introduction to RUS (1)

- The RUS-WG defines an Web Services interface to a Resource Usage Service, allowing for operations such as (current draft):
  - insertUsageRecords
    - in: list of URs; out: OperationResult
  - modify/deleteUsageRecords
    - in: XUpdate/XPath expression; out: OperationResult
  - replaceUsageRecords
    - in: RUSRecordIds, new URs; out: OperationResult
  - extractRUSUsageRecords
    - in: XPath/XQuery expression, out: list of RUS-URs

# Brief introduction to RUS (2)

- Some features (current draft):
  - allows for keeping audit trail for record insertion/modification/deletion ...
    - currently: RUS-UR wraps UR + audit trail
  - allows to specify additional mandatory elements of the UR format, that a RUS implementation needs to be present in the stored UR instances
    - list of mandatory elements upon request
  - can return faults on specific operations:
    - RUSProcessingFault, RUSInsertFault, RUSUserNotAuthorisedFault, ...

# Core RUS: XPath expressions

- XPath can select parts of a UR.
- RUS Core  returns only whole URs.
- Suggestion 1:
  - Allow RUS Core to return parts of a UR.
    - advantage: full compliance with XPath; user gets only what he wants
    - disadvantage: complex queries are less scalable

      (RUSQueryTooComplexFault?)
- Suggestion 2:
  - Restrict the allowed XPath expressions.

# Core RUS: IDL and renderings

- Currently: only WS-I
- Idea: have interface description independent of exact rendering (see Byte I/O WG)
  - advantage: specification more easily adaptable
  - disadvantage: lack of interoperability!?
- Interface Description Language (IDL) for describing methods, etc.

```
module RUS {
   interface RUS {  [...]
          rus:OperationResult insertUsageRecords(in ur:UsageRecords record);
   [...] }; };
```

- Renderings planned for WS-I & WS-RF.

# Core RUS: Batch Processing (1)

- Current status:
  - insert: input is a list of URs
  - modify: currently only single records can be modified (*requires change*)
  - replace: list of records to replace
  - delete: XPath or id-list to select records
  - query: Result is returned as one big message: *needs an extension*!

# Core RUS: Batch Processing (2)

- Additional *WS-Enumeration*-based method to get query results in small pieces?
  - Need also WS-Addressing!
  - Has the standard drawbacks?
- Client can request a limited number of records?
- Server returns a kind of RUSReplyTooBigFault?

# Core RUS: Audit Information (1)

- Currently recorded for:
  - Insertion
  - Modification (replace, increment, modify)
  - Candidate: deletion
- What is recorded:
  - Message, Signature, Timestamp, Operation, Requestor
  - Also record undo information, certificates?
  - Information on concurrency protection?

# Core RUS: Audit Information (2)

- Returned as RecordHistory in each RUSUsageRecord (upon query)
- Always returned, even if not wanted.
- Can only be returned as long as UR exists in RUS.
- Proposal: add a extractRecordHistory method.
- [Proposal: completely remove wrapping RUS-UR]

# Core RUS: Mandatory Elements

- Currently UR elements can be declared mandatory.
  - Can only declare elements from UR spec.
  - Cannot put restrictions on content.
- Proposal: Allow for other elements (UR Resource extensions) to be listed.

# MandatoryElementsType

- Currently a list of UR elements:

```
<MandatoryElementsType>
   <MachineName />
   ...
</MandatoryElementsType>
```

- Make it a list of XML names:

```
<MandatoryElementsType>
   <Element>MachineName<Element/>
   ...
</MandatoryElementsType>
```

# Advanced RUS

- Requirements gathering:
  - aggregation (discuss with UR-WG)
  - notification service for changes?
    - interesting for data replication via RUS interface
  - other ideas?

# RUS Advanced Features

- Context
  - Service Interface Definition
  - instead of implementation mechanisms
- Aggregation
  - Summarize OGF-UR Usage Records with certain aggregation grouping criteria;
- Data Replication
  - Synchronization of two or more usage record storage;
  - Out of the scope of SIDs;

# Aggregation (1)

- Aggregation as Extraction
  - Definition: aggregate and return OGF Usage Records relating to certain grouping criteria;
  - RUS:ExtractAggrUsageRecords
  - Input:
    - XPath statement
  - Output:
    - OperationResult
    - Aggregate Usage Records
  - Faults:
    - InvalidInput;
    - UserUnauthorisedFault;

# Aggregation (2)

- Aggregation as Insert
  - Definition: Aggregate and insert OGF usage records grouped with certain criteria into a separate aggregate repository;
  - Input:

# Auditing

- Who aggregate and insert usage records at when?
- Same mechanism as Job usage auditing?
- Useful for inter-grid RUS data synchronization.

# Modification

- Share same semantics with SIDs of job usage records
  - modifySpecAggrUsageRecords;
  - ReplaceAggrUsageRecords;
  - DeleteAggrUsageRecords;
- How to specify aggregate usage storage
  - As a parameter?
  - As a configuration element?
  - Others?

# Issues

- Not standard schema for aggregate usage records;
- Overlapping contexts with SIDs for job usage records;
- How to represents auditing information;
- If narrow SIDs into only “aggregation as extraction”, worry about performance (implementation-specific as well?)

# Full Copyright Notice