Republishing Mechanisms for R-GMA

Benefits and Approaches.

Talk by: Alasdair Gray

Collaborators: Andy Cooke, Lisha Ma, and Werner Nutt

Heriot-Watt University

Summary

- Current situation in R-GMA.
- Example of a republisher hierarchy that users want.
- Problems of creating and maintaining a hierarchy of republishers.
- Other open issues.

Current Situation in R-GMA: Primary Producers

- Continuous Queries:
  - Stream Producer.
  - Resilient Stream Producer.
  - Circular Buffer Producer (Deprecated).
- Snapshot Query:
  - Latest Producer.
  - Canonical Producer.
- History Query:
  - Database (History) Producer.
  - Canonical Producer.

Current Situation in R-GMA: Republishers

- Currently called an Archiver.
- Constructed in two ways:

1. `Archiver(rdms, user, pass)`

   → Database Producer

1. `Archiver(`*`insertable`*`)`

   → `Stream Producer`

   → `Latest Producer`

   → `Database Producer`

Summary

- Current situation in R-GMA.
- Example of a republisher hierarchy that users want.
- Problems of creating and maintaining a hierarchy of republishers.
- Other open issues.

Example Scenario

- Producers for `cpuLoad` running on individual machines.
  - These are very small streams (burns/brooks) of data.
- **Aim:** combine these burns to form larger streams.
- Three levels of views: `SELECT * FROM cpuLoad`

Producers of burns:

```
V1: WHERE country='britain' AND loc='hw'
    AND machineID=42
```

Confluent of burns at site level:

```
V2: WHERE country='britain' AND loc='hw'
```

Confluent of streams at national level:

```
V3: WHERE country='britain'
```

Limits of Current R-GMA Republishers

In the current R-GMA system we could not have:

- A stream republisher for `V3` consuming from `V2`.
- Forced to choose type of republisher when created.

Why can't `V3` consume from `V2`?

- No mechanisms to make sure that:
  1. Republishers don't consume from themselves.
  2. Loops of republishers are not created.

The Scenario

- Hierarchy of republishers. i.e. a republisher can consume from another republisher.
  - A republisher cannot consume from itself.
  - Based on `cpuLoad` example.
  - Illustrates:
    1. Difficulties that arise.
    2. Possible approaches.
    3. Benefits of creating hierarchies.

Assumptions:

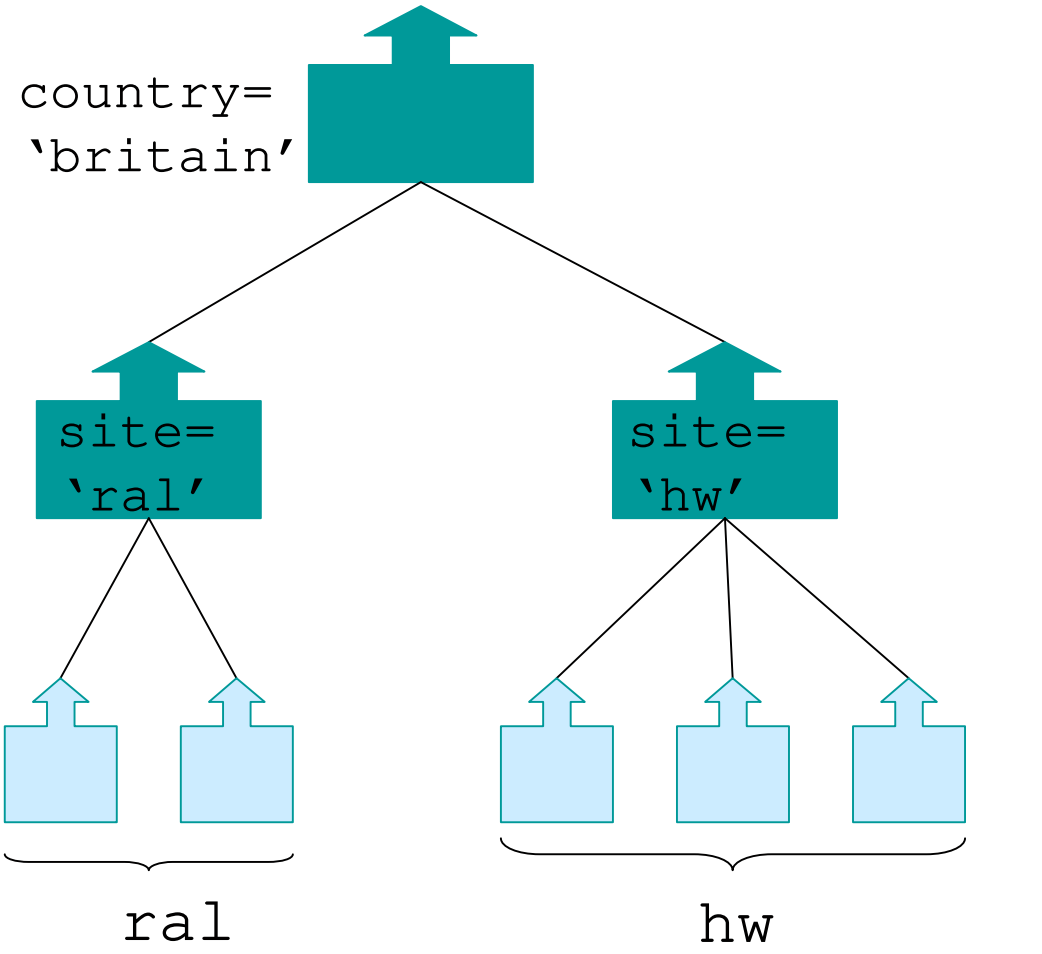
1. Republishers are complete with respect to their view definition.
2. All relevant producers are stream producers.

The Set Up

National Republisher

Local/site Republisher

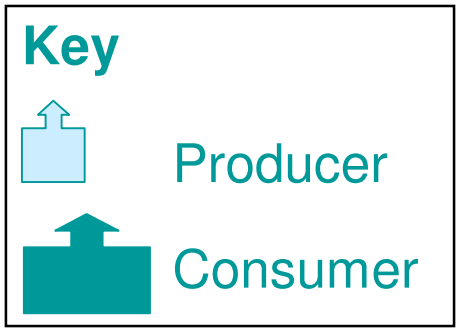
Primary Producers



Summary

- Current situation in R-GMA.
- Example of a republisher hierarchy that users want.
- Problems of creating and maintaining a hierarchy of republishers.
- Other open issues.

Question: How do we add a new producer?



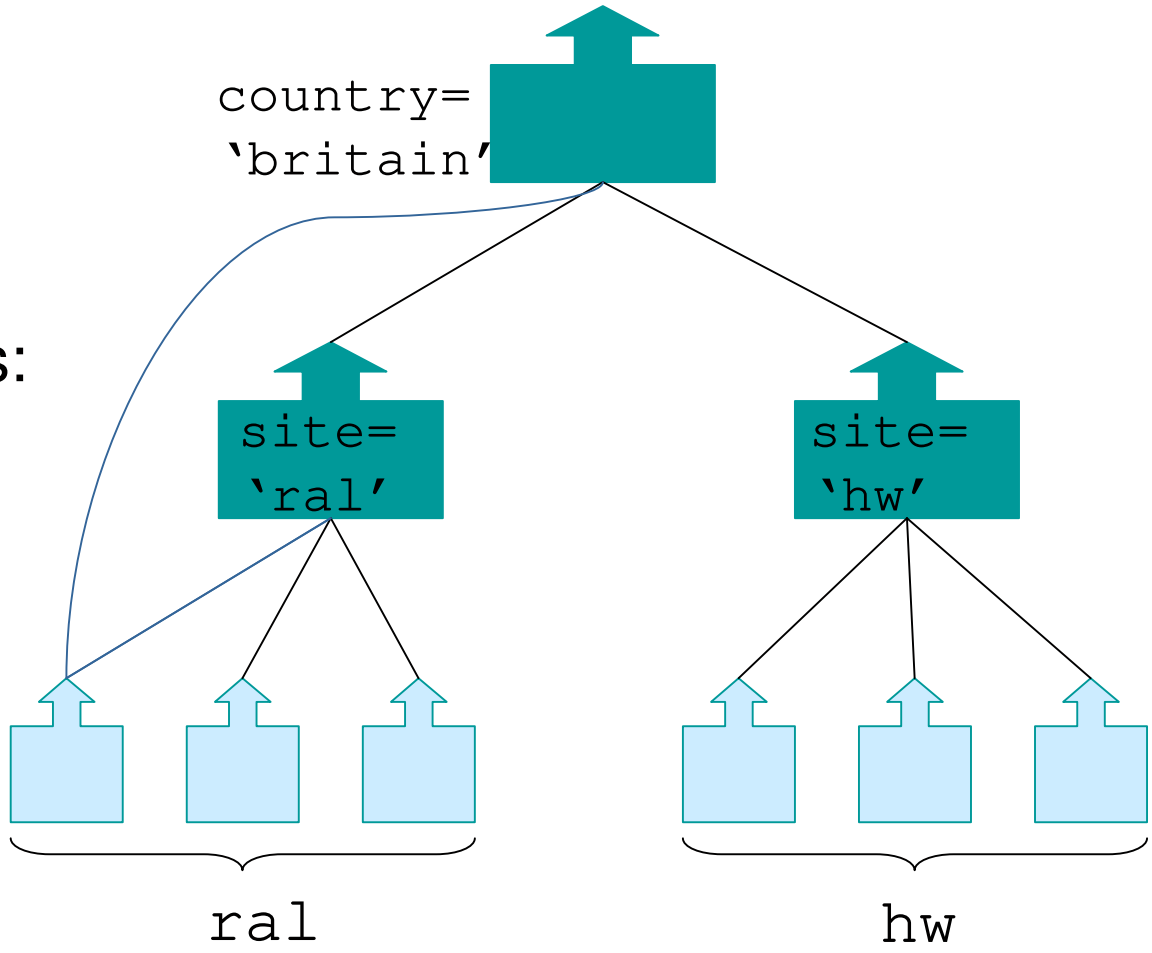
A new machine is added at `ral`.

Which consumers should be informed? There are two options:

1. Site and national republishers.

   or

2. Site republisher only.

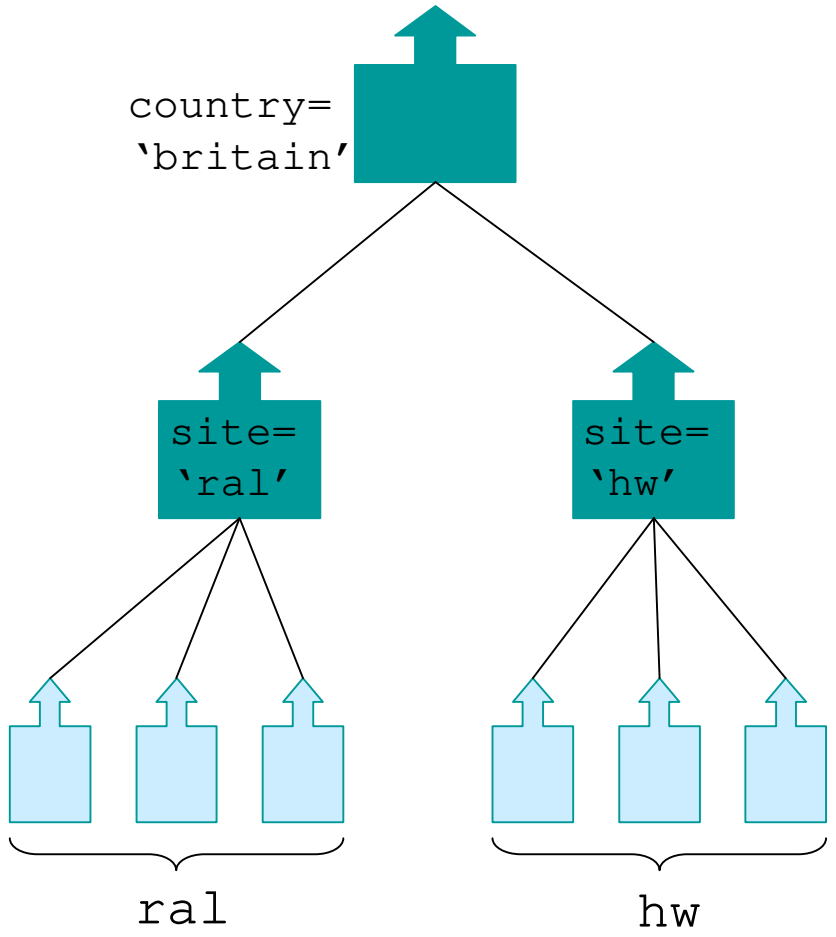


Efficiency

- Option 1: connect producer to all relevant republishers.
  - Easy to implement: simply find all relevant consumers and start streaming.
  - Duplication of tuples.
- Option 2: connect producer to most specific republisher.
  - Provides performance gains due to:
    - Lower load on new producer.
    - Lower network bandwidth.
    - No duplicate tuples (in general).
  - Requires more sophisticated logic.

Issues of Implementing Option 2

- How does the system know which republishers to inform and which to ignore?
- Which component makes this decision?
  - The republisher agents.
  - The consumer agents.
  - The registry.
- What information is needed to make this decision?
- Where is this information stored?



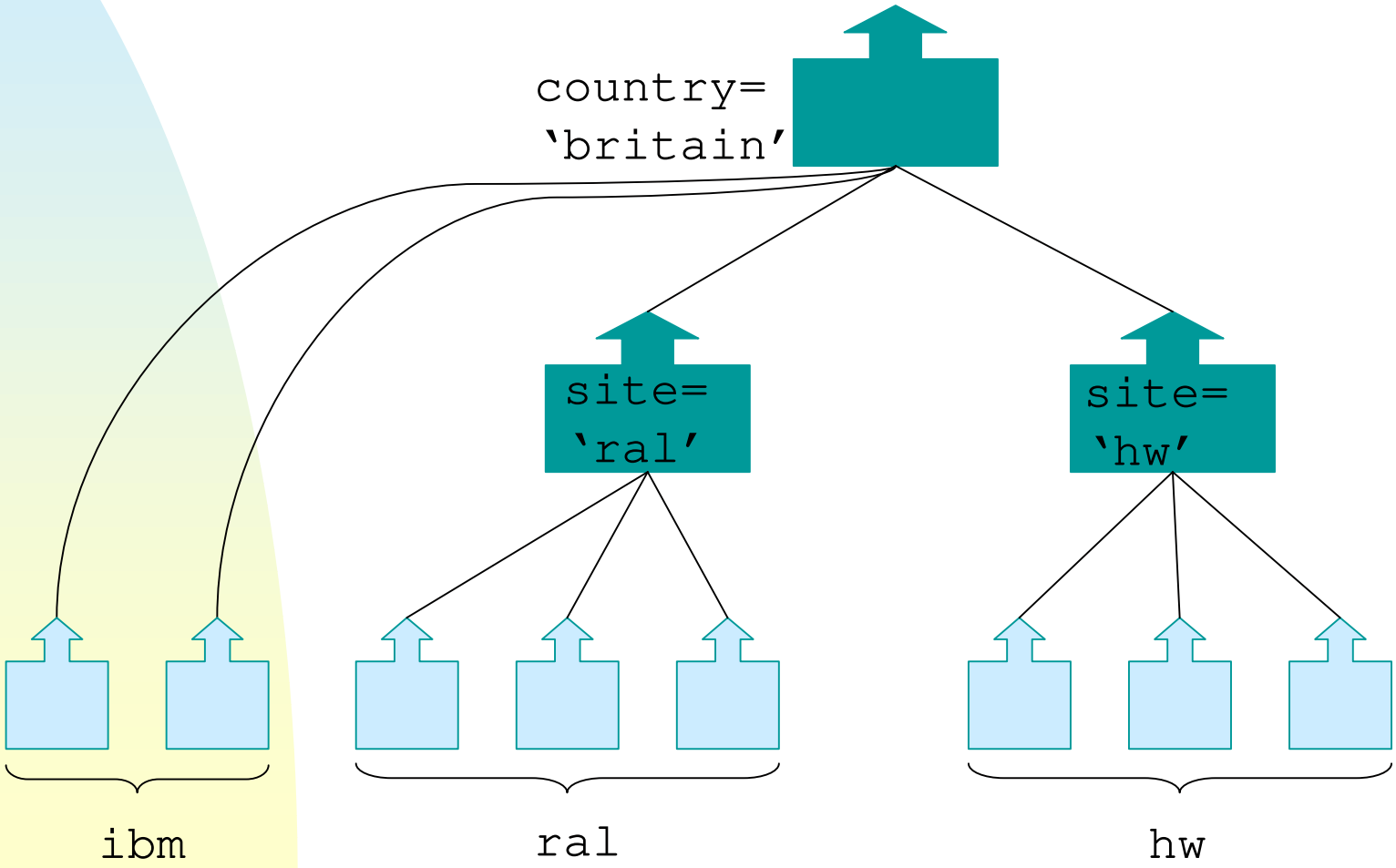
What else happens if we choose option 2?

- Need to consider the process of adding / removing an intermediary republisher.
- Effects on links between producers and other republishers.

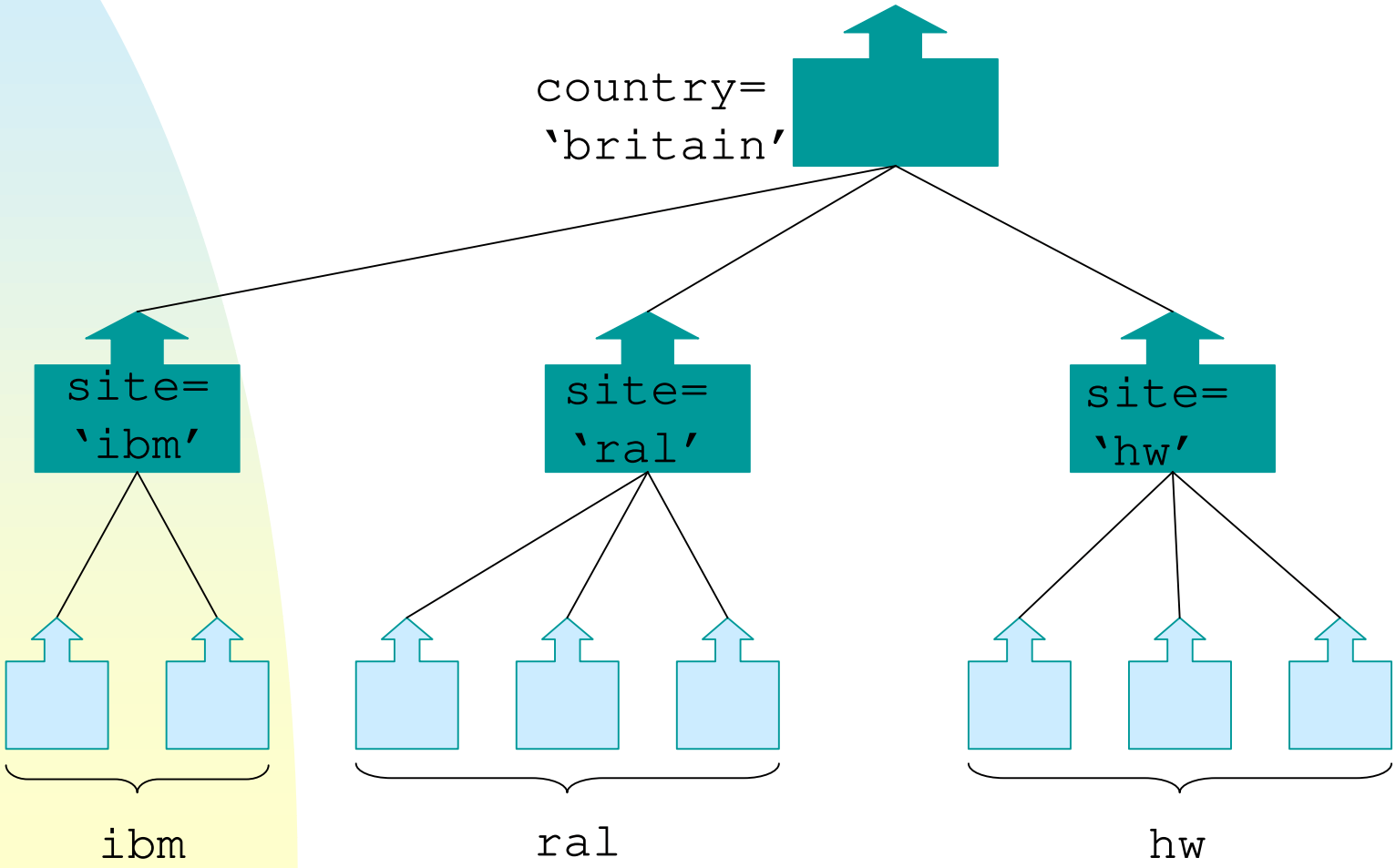
Requirements for Republishers

- No duplication of tuples.
  - Duplicates cause a problem for:
    - Aggregation queries.
    - Users performing statistical analysis.
- Completeness issues:
  1. No tuples lost.
  2. Republishes **all** tuples that conform to its view definition.
- Tuples in chronological order of timestamps.

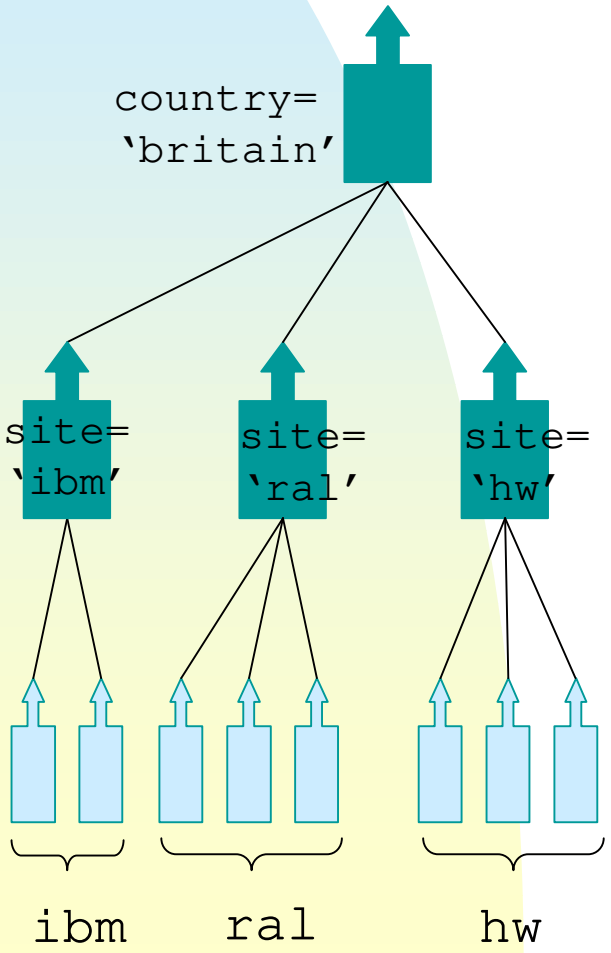
Question: How do we add an intermediary republisher?



Question: How do we add an intermediary republisher?



Steps Involved in Adding an Intermediary Republisher



Involves the following tasks:

- Creating the new republisher.
- Start the republisher consuming from relevant producers.
- Start the republisher producing tuples.
- Find relevant higher level republishers.
- Remove any existing channels between producers and higher level republishers.

Adding Intermediary Republishers is Difficult

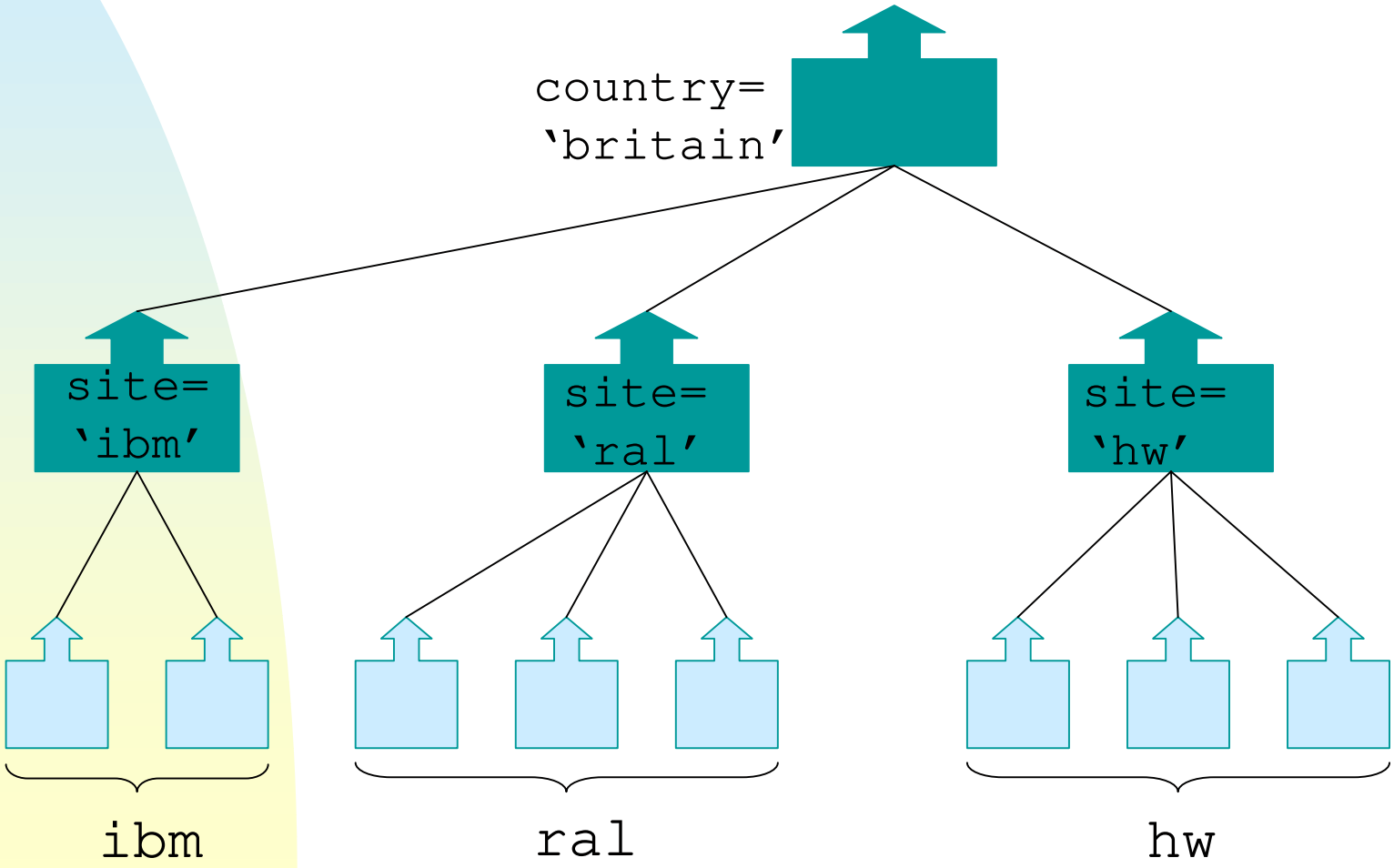
- Links between producers and higher level republisher may only be

  removed after

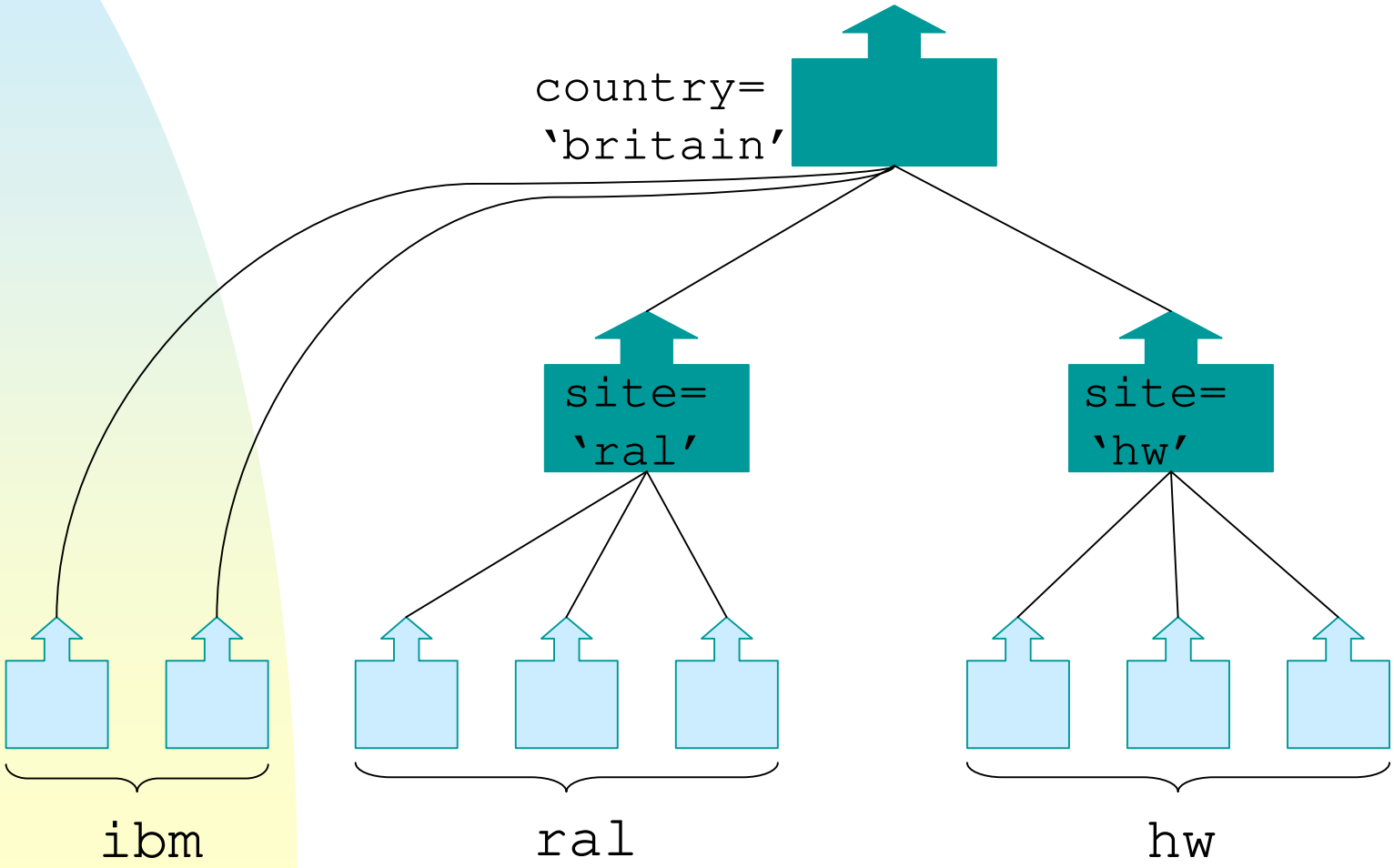
  the intermediary republisher is in place

  … otherwise we may **lose tuples**.
- However, this may lead to **duplicates**.

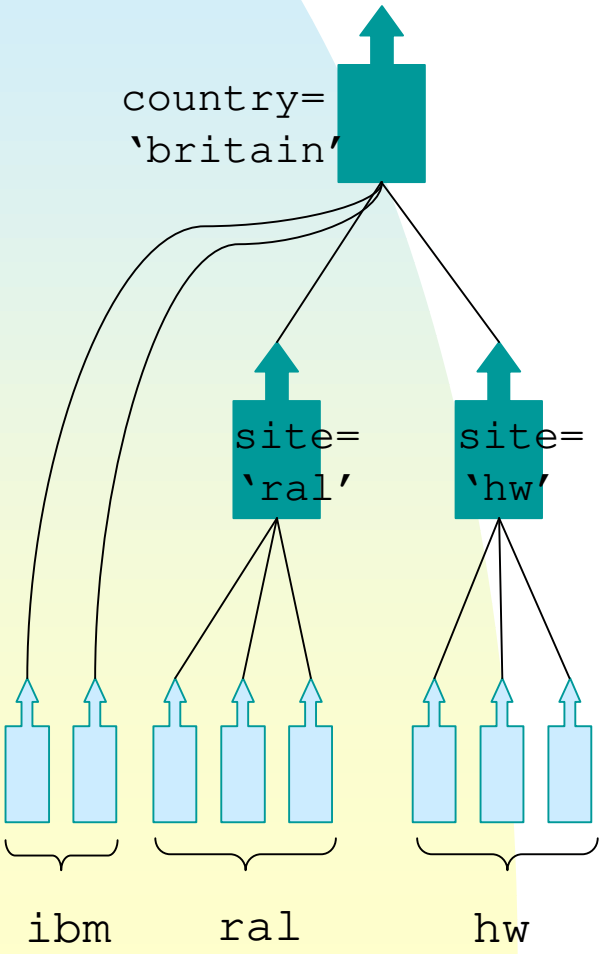
Question: How do we remove an intermediary republisher?



Question: How do we remove an intermediary republisher?



Steps Involved in Removing an Intermediary Republisher



Involves the following tasks:

- Creating links between producers and relevant higher level republishers.
- Stopping the intermediary republisher from consuming and publishing.
- Removing the intermediary republisher.

Removing an Intermediary Republisher is Difficult

- Intermediary republisher can only be **removed after** links between **producers** and **higher level republishers** are in place
  … otherwise we may **lose tuples**.
- However, this may lead to **duplicates**.

Requirements for the Protocol to Change the System

- Has to deal with:
  - Addition of new producers.
  - Addition of new intermediary republishers.
  - Removal of intermediary republishers.
- Has to achieve:
  - No loss of tuples
  - No generation of duplicate tuples.

Summary

- Current situation in R-GMA.
- Example of a republisher hierarchy that users want.
- Problems of creating and maintaining a hierarchy of republishers.
- Other open issues.

Other Issues: Completeness

- When is a republisher complete?
  - Simple if all its sources are registered as complete.
  - What if a source is a latest producer over a private stream, then can a republisher be complete that uses this source?
    - What if it ignores this source?

Other Issues: Duplicates

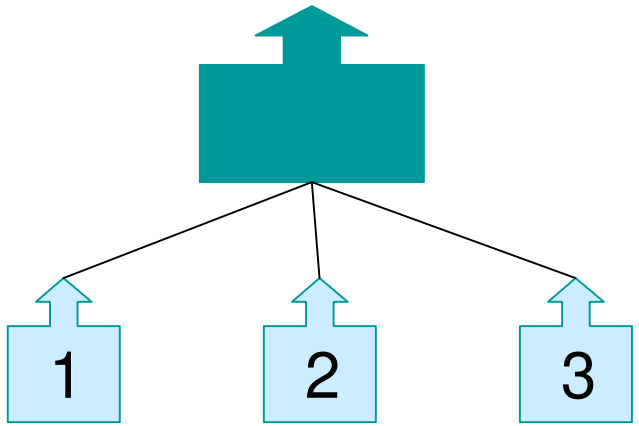
Will users be bothered?

Possibly if conducting statistical analysis of tuples.

Should we:

1. Filter duplicate tuples out. Requires duplicate tuple detection.
2. Ignore them and leave in the stream.

Other Issues: Tuple Arrival Order



Should the republisher receive:

1. All tuples from producer 1 in a burst, then producer 2, and then producer 3.
2. Apply some interleaving of tuple arrival.

Other Issues: Queries

- Which producer / republisher does the query ask for the answer?
  - The one that is the closest match.
  - All relevant producers and republishers.

    Defeats point of hierarchy.
- Should the user be able to restrict the types of query that a republisher can answer?

Discussion Points

- Ideally, how should the system behave?
- What system behaviour can the users live with?
- What are the user requirements from WP2?