What is TeRI?

- A framework for telephone-related information
  - Addresses the requirements in modern-problems
- Discussed this at Prague at IETF 94
- Successor to the TeRQ proposal
  - Generalized to acquisition, retrieval, management
- Like TeRQ, this is an information model
  - Trying to find the right semantics for records and operations
  - We’ll worry later about the proper encoding and transports
- We decided in Prague to do this in one spec
Telephone-Related Information

TeRI

Records

Admin Info

Service Info

Acquisition

Management

Queries

Just a logical picture
Moving Parts

- Acquisition protocol
  - How do I request and receive numbers?
- Management protocol
  - How do I provision services for number?
- Query protocol
  - How do I get information about a number?

- These protocols access overlapping data
  - If you can provision it, you should be able to query for it
- Surely this is a common information model
Mapping the Model to an Instance

• TeRI Records would live in servers
  – Could be public, centralized and monolithic
  – Could be distributed, or private
  – The logical architecture will be the same
  – Each TN might have multiple Records

• All sorts of entities might manage or query
  – Could be carriers, enterprises, or end users
  – Query access will vary depending on who is asking
  – Provisioning will reflect who provisioned
The TeRI Interfaces

- Client
- Inter-Mediary
- Server
- TeRI Records
- Authorities
- Queries
- Management
Operations and Records

• TeRI defines all three protocols in terms of this model
• Each protocol has its own Operations, but will operate on a common class of TeRI Records
• Operations will have their own Source, Subject, and Attributes
  – Source indicates the originator of the Operation
  – Subject would typically be a TN itself (or a range)
• TeRI Records contain information about TNs
  – Some Records might cover a range of TNs
Think SCRUD

• Search, Create, Read, Update, Delete
• Creation begins the lifecycle
  – A Registry always creates the first Record
  – Bootstrap administration record designating the Registry itself
• Should Records be partially updated, or wholly replaced?
  – Currently, the Authority who creates a record is the only one who can modify or delete it
    • i.e., a Registry creates a Record for a number, but each CSP would create a separate Record for services associated with it
The Acquisition Operation

• Query:
  – Source (Query Source, Query Intermediary)
  – Subject (Telephone Number/Range)
    • Used to have SPID, currently removed per MODERN scope
  – Attributes (constrains query, say, to finding a particular number in a range)

• Response:
  – Response Code
  – TeRI Record (newly generated assignment indicating who can control Records for this TN/Range)
The Management Operation

• Query:
  – Source (Query Source, Query Intermediary)
  – Subject (Telephone Number/Range)
    • Used to have SPID, currently removed per MODERN scope
  – TeRI Records (including Record ID)

• Response:
  – Response Code
The Retrieval Operation

• Query:
  – Source (Query Source, Query Intermediary)
  – Subject (Telephone Number/Range)
    • Used to have SPID, currently removed per MODERN scope
  – Attributes (constrains query: e.g., “voip” if only looking for VoIP, or Route Source, or Record ID)

• Response:
  – Response Code
  – TeRI Record
TeRI Record Contents

• TeRI Records would contain
  – **Subject** (the TN or TN range of the record)
  – **Authority** (Source of the data, usually the provisioner)
  – Contact (administrative contact, WHOIS/WEIRDS)
  – Service (a service associated with the TN)
  – **Identifier** (unique ID for the Record)
  – Signature (typically a crypto assurance of the Authority)

• Divided into Service and Administrative Information
  – Services records always have a Service
  – Administrative records always have a Contact

• Obviously different actors would set/get different Record elements
TeRI Record Element Types

- Telephone Number (RFC3966 – but should we revisit?)
  - Ranges – need some work here
- Domain Name
- URI
- IP Address
  - IPv4/IPv6
- Contact
  - Per jCard
- SPID
  - Currently specified as four-digits, other SPID types possible
    - GSPID, ITAD, etc.
- Trunk Group
  - Currently points to the Gurbani/Jennings RFC
- Display Name
  - Support for CNAM as well as a SIP “From” header field
- Extension
  - Reserved for further use
Transport and Encoding?

• Agree on semantics first, then define bindings and profiles
  – A binding is defined as an encoding and a transport
    • We want at least one binding per protocol, maybe allow more
  – Could build on JSON/HTTP, could build on ASN.1/UDP
  – Bindings need to detail how the elements of the data model are mapped to the encoding
    • Other low-level details like chunking, representation of cryptographic security, etc.
  – Requirement: to transcode between bindings without losing data (at an intermediary)

• Aim for maximum applicability
  – While not overcomplicating the model
This is a -00

• We need to figure out if we have the right Record elements and types
  – And an appropriate extensibility model
• Do we have the right semantics for operations?
• We need better understanding of element types