Reflexive Forwarding for CCNx and NDN Protocols …an introduction recapitulated

draft-irtf-icnrg-reflexive-forwarding-00

Hitoshi Asaeda
Dave Oran
Dirk Kutscher

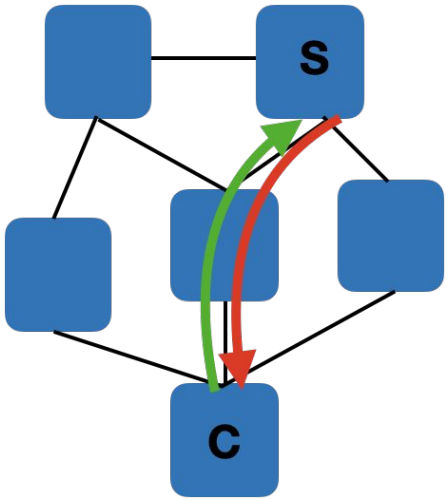
ICNRG @ IETF-121

Outline

- Motivations for multi-way interactions in ICN
- Problems with existing approaches.
- Overview of the Reflexive Forwarding design
- Use Cases for reflexive forwarding

Motivation

- Many scenarios benefit from ICN's robust and secure two-way exchange through INTEREST/DATA
- There are other scenarios though where that is not sufficient
  - RESTful communication, e.g., Web over ICN
  - Remote Method Invocation
  - Phone-home scenarios
  - Peer state synchronization
- Desirable features
  - Pushing Data
  - RESTful-like session continuation
- Our goal: enable these scenarios in an ICN-idiomatic way
  - As a foundation for the scenarios above and more
  - Most relevant (probably): RESTful ICN



Motivations for multi-way Handshakes

- Remote Method Invocation (RMI, aka RPC)
  - Fetch arguments
  - Perform authorization
  - Separate invocation from results return
- Phone-home for sensor/actuators
  - Fetch from gateway rather than push from device
  - Eliminate polling
- Peer State Synchronization
  - 3-way (or more) handshakes needed to avoid hazards
  - Complicated state machines for things needing negotiation (e.g. SIP/SDP)

Problems with Existing approaches: Pushing Data

- Interest messages get big
  - Might need fragmentation (ugh!)
  - Messes up assumption of small(ish)interests for congestion control
- Need to sign interests for pushed data to be believed
  - Bigger interest still
  - Computational cost on producer to check signature
- Wasted bandwidth if computation started by pushed data winds up abandoned

Problems with Existing approaches: Independent Exchanges

- Consumer needs a routable name prefix
  - Exposes consumer to unwanted traffic
  - Puts burden on routing to propagate far enough to reach producer
  - In mobile environments, consumer becomes producer as well, necessitating producer mobility machinery for pure client-initiated client/server exchanges
- Consumer gets to choose the name to use to reach it by
  - Opens up big hole to mount reflection attacks
- Correlating the two independent Interest/Data exchanges can be error-prone
  - Catastrophic if done wrong for key exchange
  - Complicated state machine management (c.f. SIP & SDP)

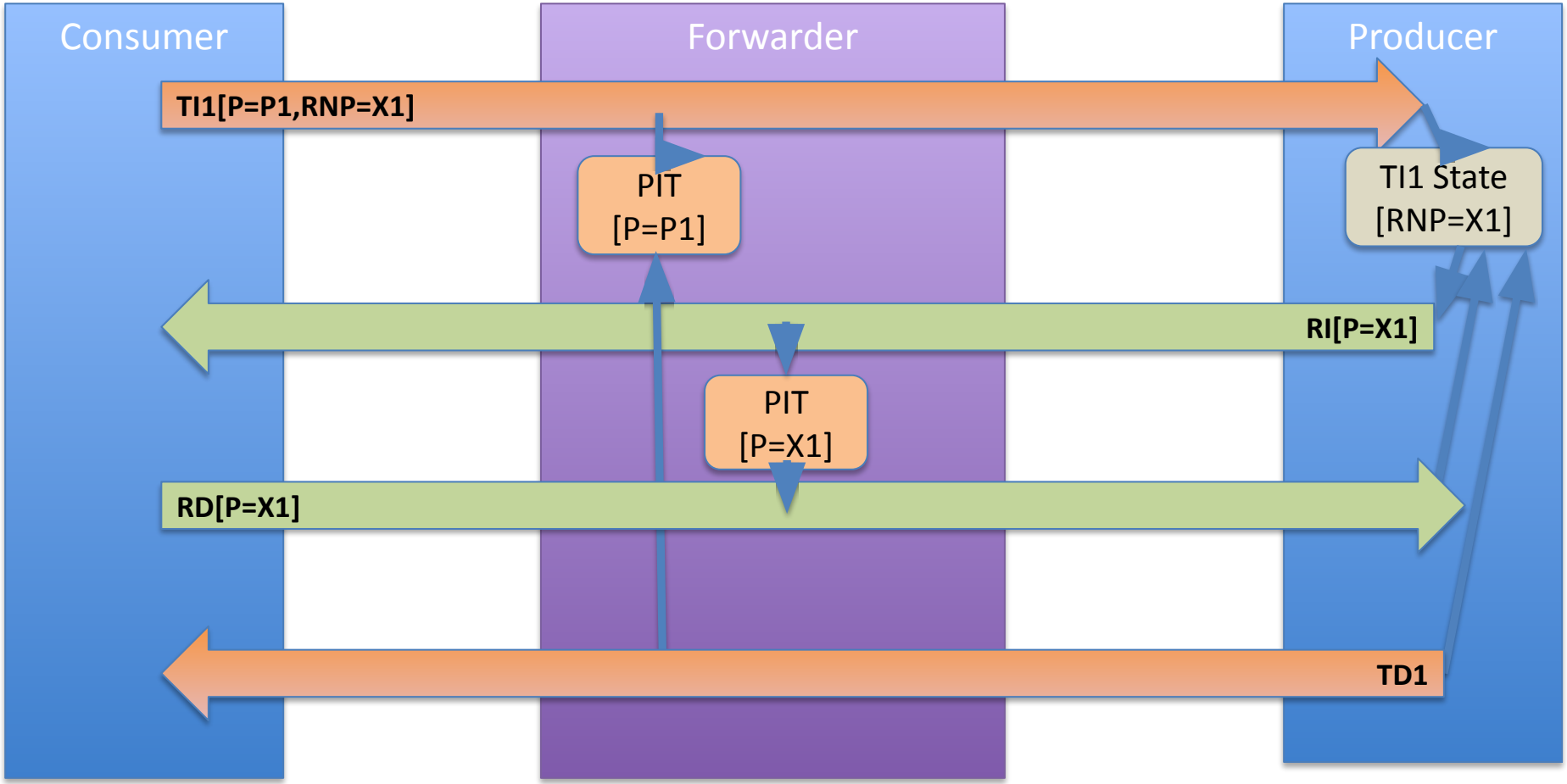
Design Overview

- Utilize forwarder state established by an Interest sent from consumer to producer
  - Allow for not just a returning Data message, but a *Reflexive* Interest to flow from producer to the unique consumer who sent the original Interest
- Define a scheme for *Reflexive Name Prefixes*
  - These can only be seen and understood by the already established consumer/producer pairing
  - They do not reveal consumer identity (temporary names within the RI context)
- Provide a forwarder mechanism to allow routing these back to the consumer from the producer
- Couple the state of the original Interest/Data exchange with the reflexive exchange(s)
  - ensure state gets mapped correctly by both consumer and producer
  - and unwound properly at the forwarders when the Data message responding to the original Interest is sent back

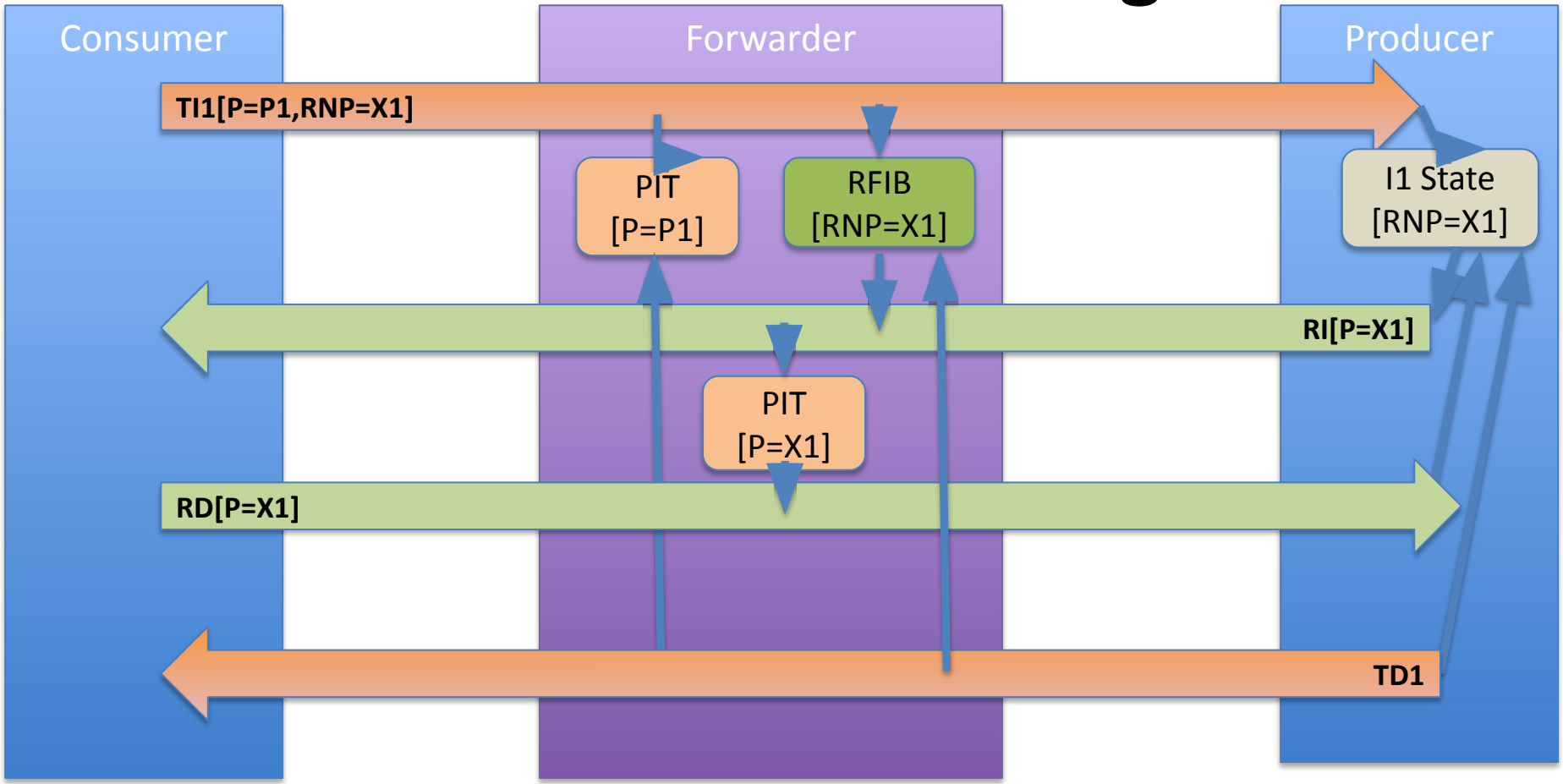
Design Overview

- Utilize forwarder state established by Interest sent from consumer to producer
  - Allow for not just a returning Data message, but a *Reflexive* Interest to flow from producer to the unique consumer who sent the original Interest
- Define a scheme for *Reflexive Name Prefixes*
  - Can only be seen and understood by already established consumer/producer pairing
  - Do not reveal consumer identity (temporary names within the RI context)
- Provide forwarder mechanism for routing these back to consumer from producer
- Couple state of the original Interest/Data exchange with the reflexive exchange(s)
  - Ensure state gets mapped correctly by both consumer and producer
  - And unwound properly at forwarders when Data message responding to original Interest is sent back

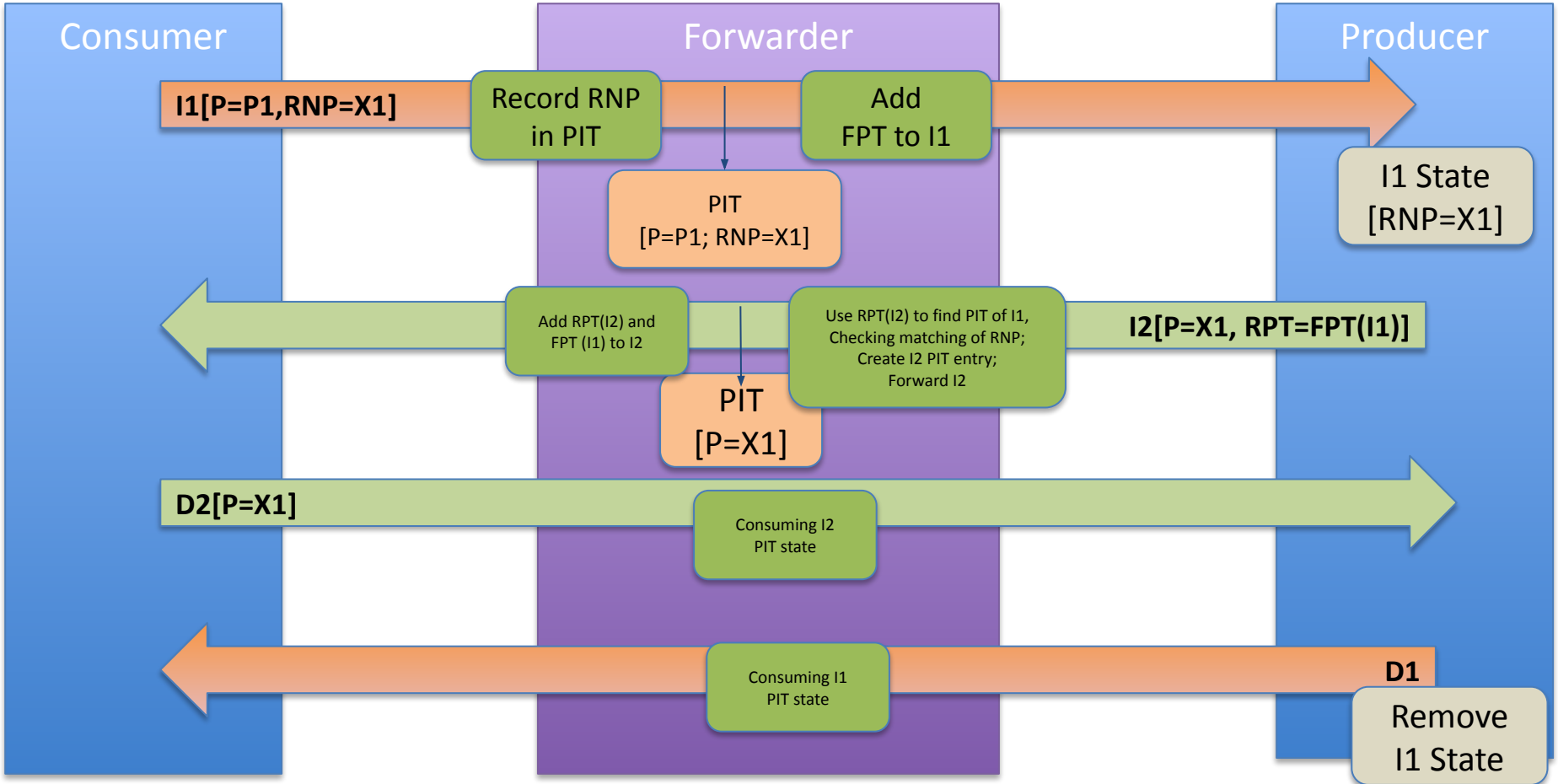
High-Level Protocol Overview



Protocol Walk-through



New Approach (version 02)



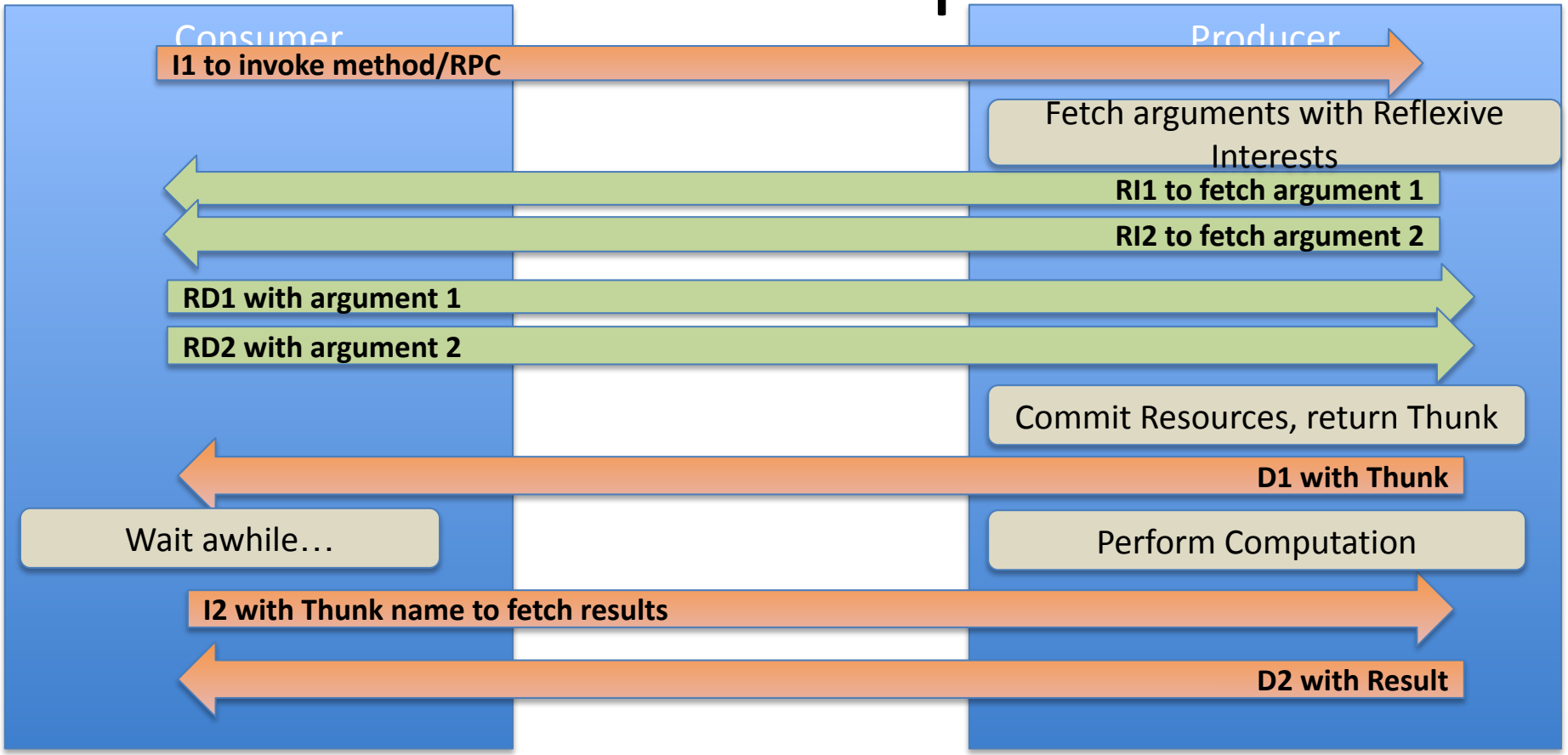
Typical Use Cases

- Remote Method Invocation
- RESTful Web Interactions
- Data Pull from sensors

Remote Method Invocation
(Pioneered by RICE)

- RICE uses (an earlier version of) Reflexive Interests for the following:
  - Retrieve authentication/authorization information from consumer
  - Fetch arguments to method calls
- Completion can be either:
  - Immediate through the returning Data message, or
  - Deferred to a separate exchange to retrieve results buy utilizing *Thunks.*
- Illustrated on following slide

RMI Example


Consumer
Producer
I1 to invoke method/RPC
Fetch arguments with Reflexive Interests
RI1 to fetch argument 1
RI2 to fetch argument 2
RD1 with argument 1
RD2 with argument 2
Commit Resources, return Thunk
D1 with Thunk
Wait awhile…
Perform Computation
I2 with Thunk name to fetch results
D2 with Result

RESTful Web Interactions

- Only place RESTful request via the URI in the initial Interest
- Get all the parameters, including AuthZ with Reflexive Interests
  - Cookies, Accept-foo headers, other HTTP goop
- Return results via regular Data messages

Data Pull from sensors

- Sensor only needs to act as consumer
- Wake up (on timer or event)
- ”Phone Home” to an application gateway or REPO
- This provokes a Reflexive Interest/Data exchange initiated from the gateway
- Data can either be:
  - Packaged/stored by gateway as the authoritative source
  - Named, encapsulated and signed by sensor itself

Phone Home Data Pull Example