

REST Design Patterns for Robust Asynchronous Notification

Using simple observe/notify to build a robust and reusable design pattern for asynchronous notifications

Problems

- Observe is not a well managed relationship
 - The list of observers is hidden server state
 - Client can't be certain if it is still in the list
 - Conditional Observe is difficult to manage
- Events have life cycle beyond one notification
 - Alerts are generated, acknowledged, and eventually cleared
 - Use cases for asynchronous Event delivery, polling, and batch Event processing

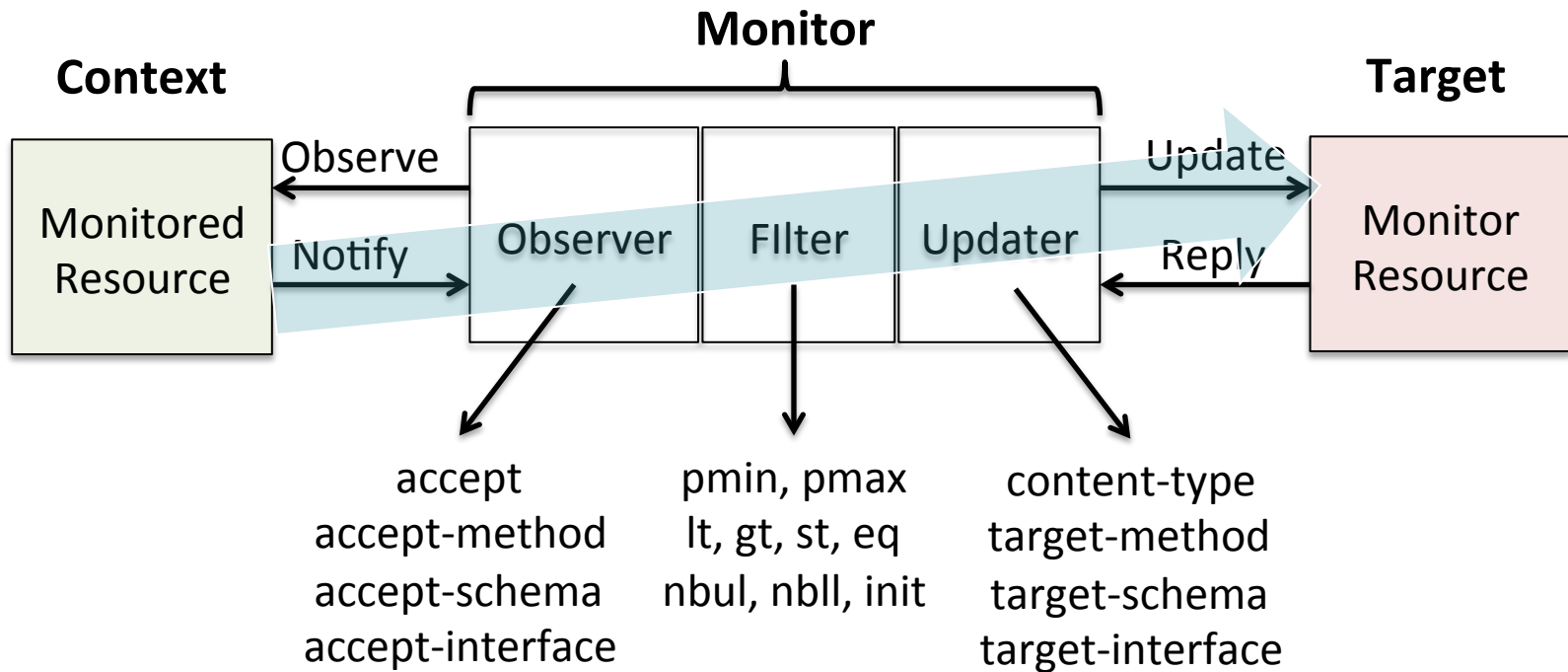
Design Patterns

- Monitor
 - Create a managed Observe relationship using a REST resource with a defined link relation and parameter set
- Events
 - REST resource to represent an Event instance
 - Maintain Event instances in an observable collection

Monitor

- Use the IANA registered "monitor" link relation
 - *Description: Refers to a resource that can be used to monitor changes in an HTTP resource (RFC5989)*
 - Similar to "boundto" (dynlink) but defines a unidirectional state update from context to target.
- A Monitor may use Observe on the server to obtain state changes of the context resource
- A Monitor may implement conditional notification using filter parameters (dynlink) as well as defining transfer methods and formats
- Monitor parameters may be encoded as link attributes or stored in a monitor configuration resource
- A Monitor may support multiple source and target protocols based on URI scheme (mqtt, coap, http)

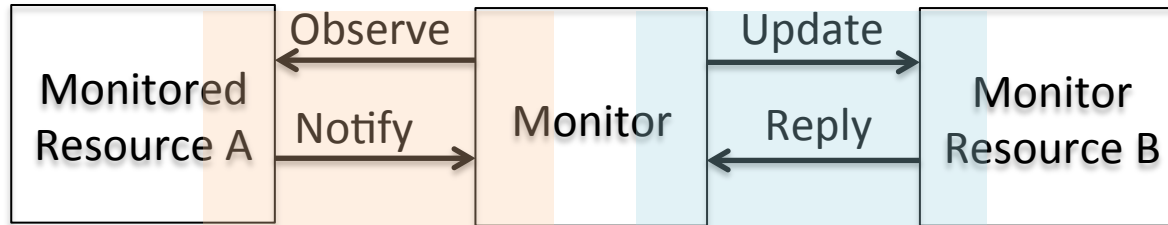
Monitor



Monitor Patterns

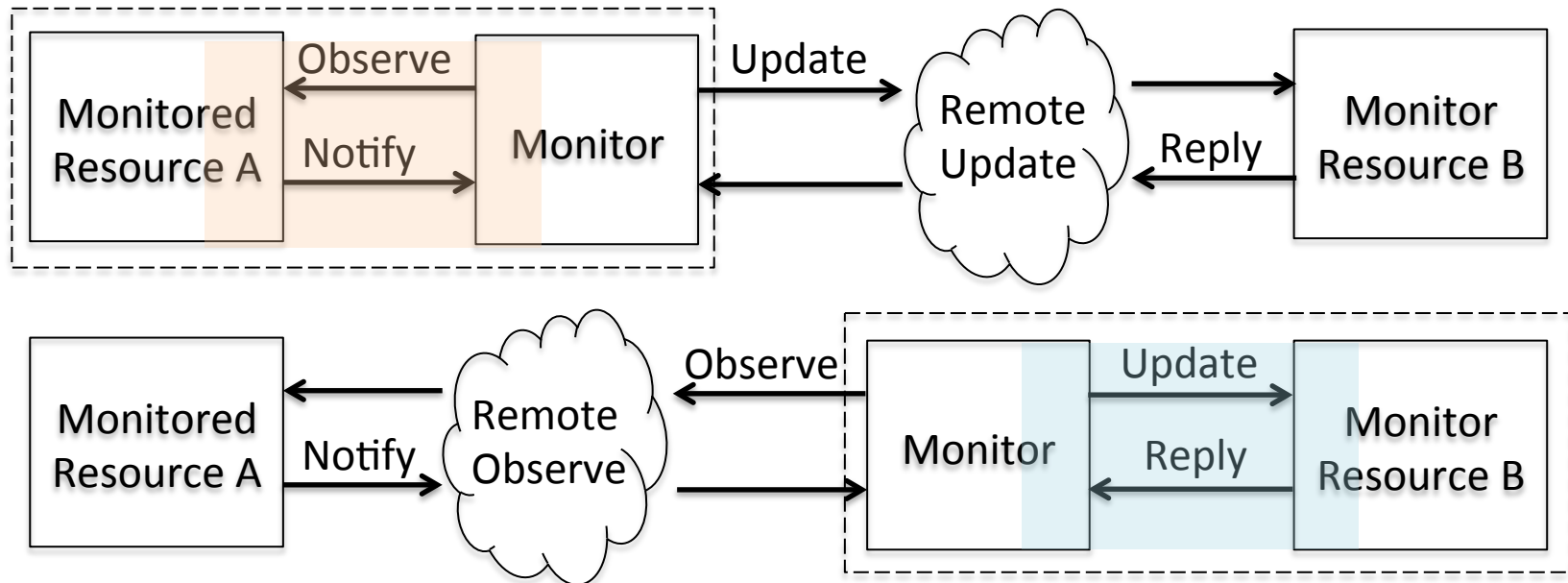
A Pushes State To B

```
<B>;rel=monitor
```



B Observes State From A

```
<>;anchor=A;rel=monitor
```



Monitor Link Examples

Monitor updates to the context using a resource

```
{  
  "rel": "monitor",  
  "href": "monitor"  
}
```

Observe a remote resource and update the context

```
{  
  "anchor": "coap://0m2m.net:5683/example/test",  
  "rel": "monitor",  
  "href": ""  
}
```

Monitor Link Examples

Subscribe a resource to an MQTT topic

```
{  
  "anchor": "mqtt://0m2m.net/example/topic",  
  "rel": "monitor",  
  "href": "updated-on-mqtt-notify"  
}
```

Publish updates on a resource to an MQTT topic

```
{  
  "anchor": "publish-updates-to-mqtt",  
  "rel": "monitor",  
  "href": "mqtt://0m2m.net/example/topic"  
}
```


Events

- State changes that require more than simple notification may be handled as Events
- Events have a life cycle, like log records, alerts, etc.
- A monitor may add state change notifications to a collection of Event instances using CREATE
- The Event collection is Observable and transmits newly created Event instances as notifications