CoRE Working Group
Internet-Draft
Intended status: Informational
Expires: January 13, 2022

M. Koster
SmartThings
B. Silverajan, Ed.
Tampere University

July 12, 2021

Conditional Attributes for Constrained RESTful Environments draft-ietf-core-conditional-attributes-00

Abstract

This specification defines Conditional Notification and Control Attributes that work with CoAP Observe (RFC7641).

Editor note

The git repository for the draft is found at TBD

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 13, 2022.

Copyright Notice

Table of Contents

1. Introduction

IETF Standards for machine to machine communication in constrained environments describe a REST protocol [RFC7252] and a set of related information standards that may be used to represent machine data and machine metadata in REST interfaces.

This specification defines Conditional Notification and Control Attributes for use with CoRE Observe [RFC7641].

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This specification requires readers to be familiar with all the terms and concepts that are discussed in [RFC7641]. This specification makes use of the following additional terminology:

Notification Band: A resource value range that may be bounded by a minimum and maximum value or may be unbounded having either a minimum or maximum value.

3. Conditional Attributes

This specification defines conditional attributes, which provide for fine-grained control of notification and state synchronization when using CoRE Observe [RFC7641]. When resource interfaces following this specification are made available over CoAP, the CoAP Observation mechanism [RFC7641] MAY also be used to observe any changes in a resource, and receive asynchronous notifications as a result. A resource marked as Observable in its link description SHOULD support these conditional attributes.

Note: In this draft, we assume that there are finite quantization effects in the internal or external updates to the value representing the state of a resource; specifically, that a resource state may be updated at any time with any valid value. We therefore avoid any continuous-time assumptions in the description of the conditional attributes and instead use the phrase "sampled value" to refer to a member of a sequence of values that may be internally observed from the resource state over time.

3.1. Conditional Notification Attributes

Conditional Notification Attributes define the conditions that trigger a notification. Conditional Notification Attributes SHOULD be evaluated on all potential notifications from a resource, whether resulting from an internal server-driven sampling process or from external update requests to the server.

The set of Conditional Notification Attributes defined here allow a client to control how often a client is interested in receiving notifications and how much a value should change for the new representation state to be interesting. One or more Conditional

Notification Attributes MAY be included as query parameters in an Observe request.

Conditional Notification Attributes are defined below:

Attribute	Parameter	Value
Greater Than	gt	xs:decimal
Less Than	lt	xs:decimal
Change Step	st	xs:decimal (>0)
Notification Band	band	xs:boolean
Edge	edge	xs:boolean

Table 1: Conditional Notification Attributes

3.1.1. Greater Than (gt)

When present, Greater Than indicates the upper limit value the sampled value SHOULD cross before triggering a notification. A notification is sent whenever the sampled value crosses the specified upper limit value, relative to the last reported value, and the time fpr pmin has elapsed since the last notification. The sampled value is sent in the notification. If the value continues to rise, no notifications are generated as a result of gt. If the value drops below the upper limit value then a notification is sent, subject again to the pmin time.

The Greater Than parameter can only be supported on resources with a scalar numeric value.

3.1.2. Less Than (lt)

When present, Less Than indicates the lower limit value the resource value SHOULD cross before triggering a notification. A notification is sent when the samples value crosses the specified lower limit value, relative to the last reported value, and the time fpr pmin has elapsed since the last notification. The sampled value is sent in the notification. If the value continues to fall no notifications are generated as a result of lt. If the value rises above the lower limit value then a new notification is sent, subject to the pmin time.

The Less Than parameter can only be supported on resources with a scalar numeric value.

3.1.3. Change Step (st)

When present, the change step indicates how much the value representing a resource state SHOULD change before triggering a notification, compared to the old state. Upon reception of a query including the st attribute, the current resource state representing the most recently sampled value is reported, and then set as the last reported value (last_rep_v). When a subsequent sampled value or update of the resource state differs from the last reported state by an amount, positive or negative, greater than or equal to st, and the time for pmin has elapsed since the last notification, a notification is sent and the last reported value is updated to the new resource state sent in the notification. The change step MUST be greater than zero otherwise the receiver MUST return a CoAP error code 4.00 "Bad Request" (or equivalent).

The Change Step parameter can only be supported on resource states represented with a scalar numeric value.

Note: Due to sampling and other constraints, e.g. pmin, the change in resource states received in two sequential notifications may differ by more than st.

3.1.4. Notification Band (band)

The notification band attribute allows a bounded or unbounded (based on a minimum or maximum) value range that may trigger multiple notifications. This enables use cases where different ranges results in differing behaviour. For example, in monitoring the temperature of machinery, whilst the temperature is in the normal operating range, only periodic updates are needed. However as the temperature moves to more abnormal ranges more frequent state updates may be sent to clients.

Without a notification band, a transition across a less than (lt), or greater than (gt) limit only generates one notification. This means that it is not possible to describe a case where multiple notifications are sent so long as the limit is exceeded.

The band attribute works as a modifier to the behaviour of gt and lt. Therefore, if band is present in a query, gt, lt or both, MUST be included.

When band is present with the lt attribute, it defines the lower bound for the notification band (notification band minimum).

Notifications occur when the resource value is equal to or above the notification band minimum. If lt is not present there is no minimum value for the band.

When band is present with the gt attribute, it defines the upper bound for the notification band (notification band maximum). Notifications occur when the resource value is equal to or below the notification band maximum. If gt is not present there is no maximum value for the band.

If band is present with both the gt and lt attributes, notification occurs when the resource value is greater than or equal to gt or when the resource value is less than or equal to lt.

If a band is specified in which the value of gt is less than that of lt, in-band notification occurs. That is, notification occurs whenever the resource value is between the gt and lt values, including equal to gt or lt.

If the band is specified in which the value of gt is greater than that of lt, out-of-band notification occurs. That is, notification occurs when the resource value not between the gt and lt values, excluding equal to gt and lt.

The Notification Band parameter can only be supported on resources with a scalar numeric value.

3.1.5. Edge (edge)

When present, the Edge attribute indicates interest for receiving notifications of either the falling edge or the rising edge transition of a boolean resource state. When the value of the Edge attribute is 0, the server notifies the client each time a resource state changes from True to False. When the value of the Edge attribute is 1, the server notifies the client each time a resource state changes from False to True.

The Edge attribute can only be supported on resources with a boolean value.

3.2. Conditional Control Attributes

Conditional Control Attributes define the time intervals between consecutive notifications as well as the cadence of the measurement of the conditions that trigger a notification. Conditional Control Attributes can be used to configure the internal server-driven sampling process for performing measurements of the conditions of a

resource. One or more Conditional Control Attributes MAY be included as query parameters in an Observe request.

Conditional Control Attributes are defined below:

Attribute	Parameter	Value
Minimum Period (s)	pmin	xs:decimal (>0)
Maximum Period (s)	pmax	xs:decimal (>0)
Minimum Evaluation Period (s)	epmin	xs:decimal (>0)
Maximum Evaluation Period (s)	epmax	xs:decimal (>0)
Confirmable Notification	con	xs:boolean

Table 2: Conditional Control Attributes

3.2.1. Minimum Period (pmin)

When present, the minimum period indicates the minimum time, in seconds, between two consecutive notifications (whether or not the resource state has changed). In the absence of this parameter, the minimum period is up to the server. The minimum period MUST be greater than zero otherwise the receiver MUST return a CoAP error code 4.00 "Bad Request" (or equivalent).

A server MAY update the resource state with the last sampled value that occured during the pmin interval, after the pmin interval expires.

Note: Due to finite quantization effects, the time between notifications may be greater than pmin even when the sampled value changes within the pmin interval. Pmin may or may not be used to drive the internal sampling process.

3.2.2. Maximum Period (pmax)

When present, the maximum period indicates the maximum time, in seconds, between two consecutive notifications (whether or not the resource state has changed). In the absence of this parameter, the maximum period is up to the server. The maximum period MUST be greater than zero and MUST be greater than, or equal to, the minimum period parameter (if present) otherwise the receiver MUST return a CoAP error code 4.00 "Bad Request" (or equivalent).

3.2.3. Minimum Evaluation Period (epmin)

When present, the minimum evaluation period indicates the minimum time, in seconds, the client recommends to the server to wait between two consecutive measurements of the conditions of a resource since the client has no interest in the server doing more frequent measurements. When the minimum evaluation period expires after the previous measurement, the server MAY immediately perform a new measurement. In the absence of this parameter, the minimum evaluation period is not defined and thus not used by the server. The server MAY use pmin, if defined, as a guidance on the desired measurement cadence. The minimum evaluation period MUST be greater than zero otherwise the receiver MUST return a CoAP error code 4.00 "Bad Request" (or equivalent).

3.2.4. Maximum Evaluation Period (epmax)

When present, the maximum evaluation period indicates the maximum time, in seconds, the server MAY wait between two consecutive measurements of the conditions of a resource. When the maximum evaluation period expires after the previous measurement, the server MUST immediately perform a new measurement. In the absence of this parameter, the maximum evaluation period is not defined and thus not used by the server. The maximum evaluation period MUST be greater than zero and MUST be greater than the minimum evaluation period parameter (if present) otherwise the receiver MUST return a CoAP error code 4.00 "Bad Request" (or equivalent).

3.2.5. Confirmable Notification (con)

When present with a value of 1 in a query, the con attribute indicates a notification MUST be confirmable, i.e., the server MUST send the notification in a confirmable CoAP message, to request an acknowledgement from the client. When present with a value of 0 in a query, the con attribute indicates a notification can be confirmable or non-confirmable, i.e., it can be sent in a confirmable or a non-confirmable CoAP message.

3.3. Server processing of Conditional Attributes

Conditional Notification Attributes and Conditional Control Attributes may be present in the same query. However, they are not defined at multiple prioritization levels. The server sends a notification whenever any of the parameter conditions are met, upon which it updates its last notification value and time to prepare for the next notification. Only one notification occurs when there are multiple conditions being met at the same time. The reference code

below illustrates the logic to determine when a notification is to be sent.

```
bool notifiable( Resource * r ) {

#define BAND r->band
#define SCALAR_TYPE ( num_type == r->type )
#define STRING_TYPE ( str_type == r->type )
#define BOOLEAN_TYPE ( bool_type == r->type )
#define PMIN_EX ( r->last_sample_time - r->last_rep_time >= r->pmin )
#define PMAX_EX ( r->last_sample_time - r->last_rep_time > r->pmax )
#define LT_EX ( r->v < r->lt ^ r->last_rep_v < r->lt )
#define GT_EX ( r->v > r->gt ^ r->last_rep_v > r->gt )
#define ST_EX ( abs( r->v - r->last_rep_v ) >= r->st )
#define IN_BAND ( ( r->gt <= r->v && r->v <= r->lt ) || ( r->lt <= r->gt && r-
>gt <= r->v ) || ( r->v <= r->lt && r->lt <= r->gt ) )
#define VB_CHANGE ( r->vb != r->last_rep_vb )
#define VS_CHANGE ( r->vs != r->last_rep_vs )

  return (
    PMIN_EX &&
    ( SCALAR_TYPE ?
      ( ( !BAND && ( GT_EX || LT_EX || ST_EX || PMAX_EX ) ) ||
        ( BAND && IN_BAND && ( ST_EX || PMAX_EX) ) )
    : STRING_TYPE ?
      ( VS_CHANGE || PMAX_EX )
    : BOOLEAN_TYPE ?
      ( VB_CHANGE || PMAX_EX )
    : false )
  );
}
```

Figure 1: Code logic for conditional notification attribute interactions

4. Implementation Considerations

When pmax and pmin are equal, the expected behaviour is that notifications will be sent every (pmin == pmax) seconds. However, these notifications can only be fulfilled by the server on a best effort basis. Because pmin and pmax are designed as acceptable tolerance bounds for sending state updates, a query from an interested client containing equal pmin and pmax values must not be seen as a hard real-time scheduling contract between the client and the server.

When using multiple resource bindings (e.g. multiple Observations of resource) with different bands, consideration should be given to the

resolution of the resource value when setting sequential bands. For

example: Given BandA (Abmn=10, Bbmx=20) and BandB (Bbmn=21, Bbmx=30). If the resource value returns an integer then notifications for values between and inclusive of 10 and 30 will be triggered. Whereas if the resolution is to one decimal point (0.1) then notifications for values 20.1 to 20.9 will not be triggered.

The use of the notification band minimum and maximum allow for a synchronization whenever a change in the resource value occurs. Theoretically this could occur in-line with the server internal sample period or the configuration of epmin and epmax values for determining the resource value. Implementors SHOULD consider the resolution needed before updating the resource, e.g. updating the resource when a temperature sensor value changes by 0.001 degree versus 1 degree.

When a server has multiple observations with different measurement cadences as defined by the epmin and epmax values, the server MAY evaluate all observations when performing the measurement of any one observation.

5. Security Considerations

TBD IANA Considerations ==========

TBD

6. Acknowledgements

Hannes Tschofenig and Mert Ocak highlighted syntactical corrections in the usage of pmax and pmin in a query. Alan Soloway contributed text leading to the inclusion of epmin and epmax. David Navarro proposed allowing for pmax to be equal to pmin.

7. Contributors

Christian Groves
Australia
email: cngroves.std@gmail.com

Zach Shelby
ARM
Vuokatti
FINLAND
phone: +358 40 7796297
email: zach.shelby@arm.com

Matthieu Vial
Schneider-Electric
Grenoble
France
phone: +33 (0)47657 6522
eMail: matthieu.vial@schneider-electric.com

Jintao Zhu
Huawei
Xi'an, Shaanxi Province
China
email: jintao.zhu@huawei.com

8. Changelog

draft-ietf-core-conditional-attributes-00

- Conditional Atttributes section from draft-ietf-core-dynlink-13 separated into own WG draft

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC7641] Hartke, K., "Observing Resources in the Constrained Application Protocol (CoAP)", RFC 7641, DOI 10.17487/RFC7641, September 2015, <https://www.rfc-editor.org/info/rfc7641>.

Appendix A. Examples

This appendix provides some examples of the use of binding attribute / observe attributes.

Note: For brevity the only the method or response code is shown in the header field.

A.1. Minimum Period (pmin) example

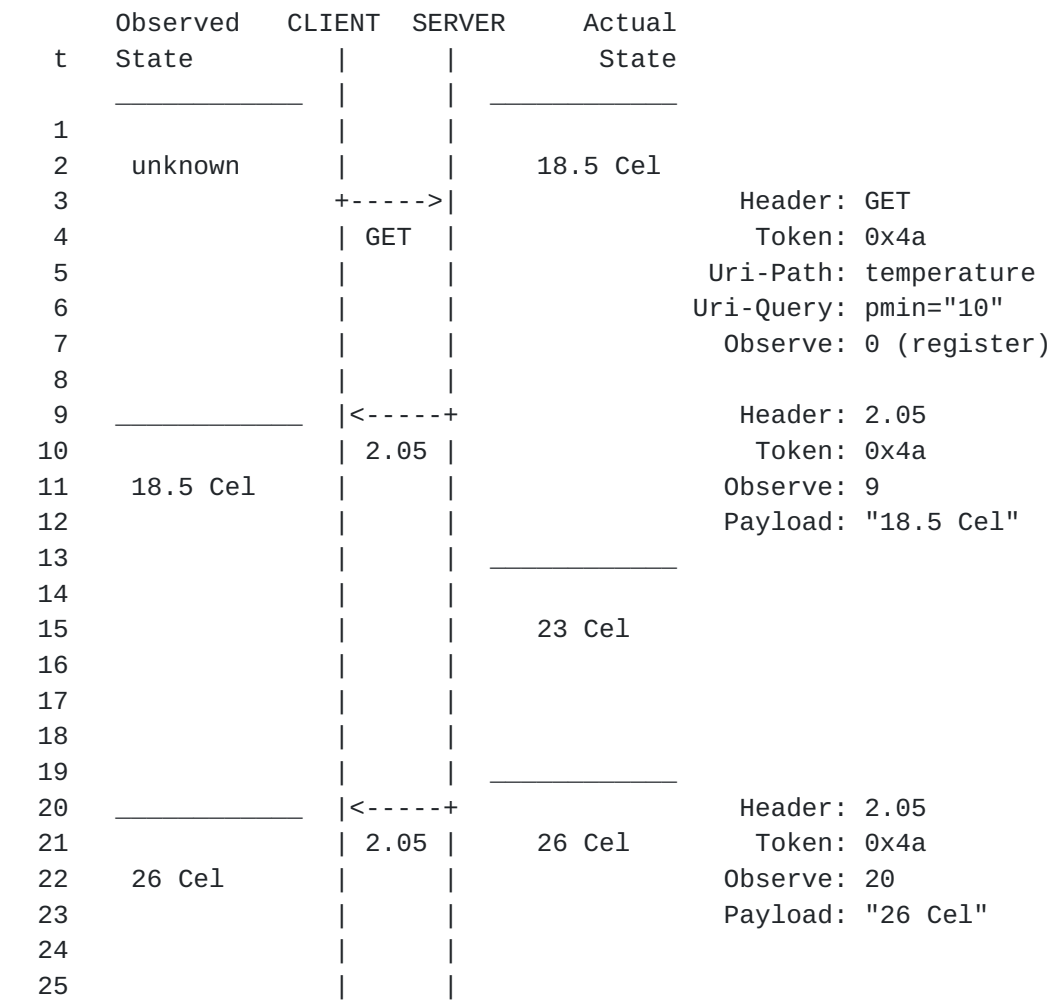
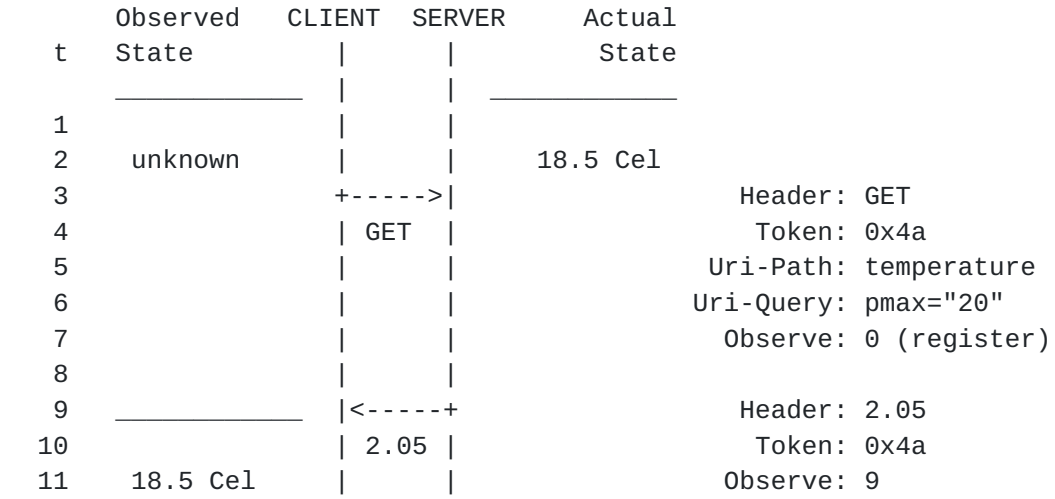


Figure 2: Client registers and receives one notification of the current state and one of a new state state when pmin time expires.

A.2. Maximum Period (pmax) example



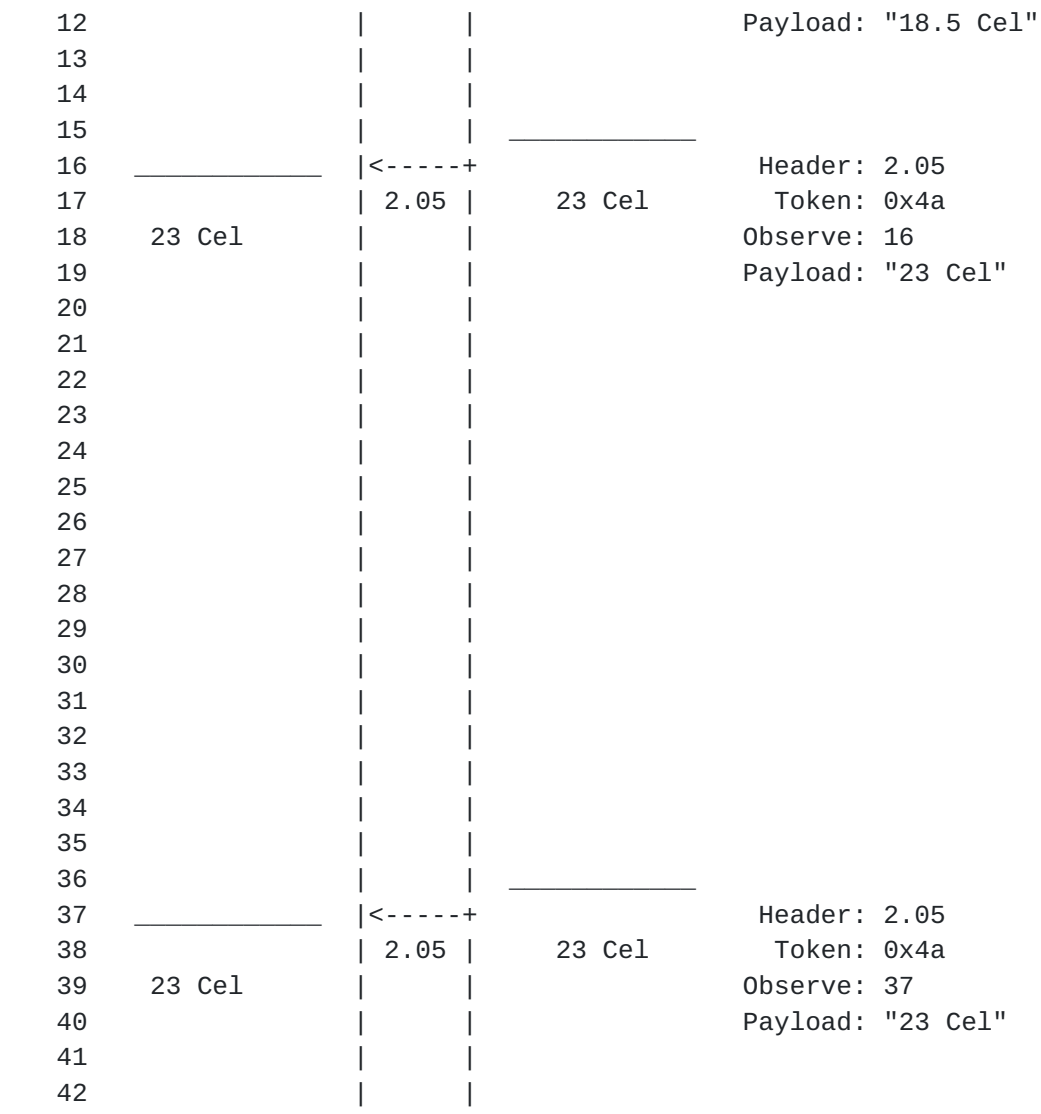


Figure 3: Client registers and receives one notification of the current state, one of a new state and one of an unchanged state when pmax time expires.

A.3. Greater Than (gt) example

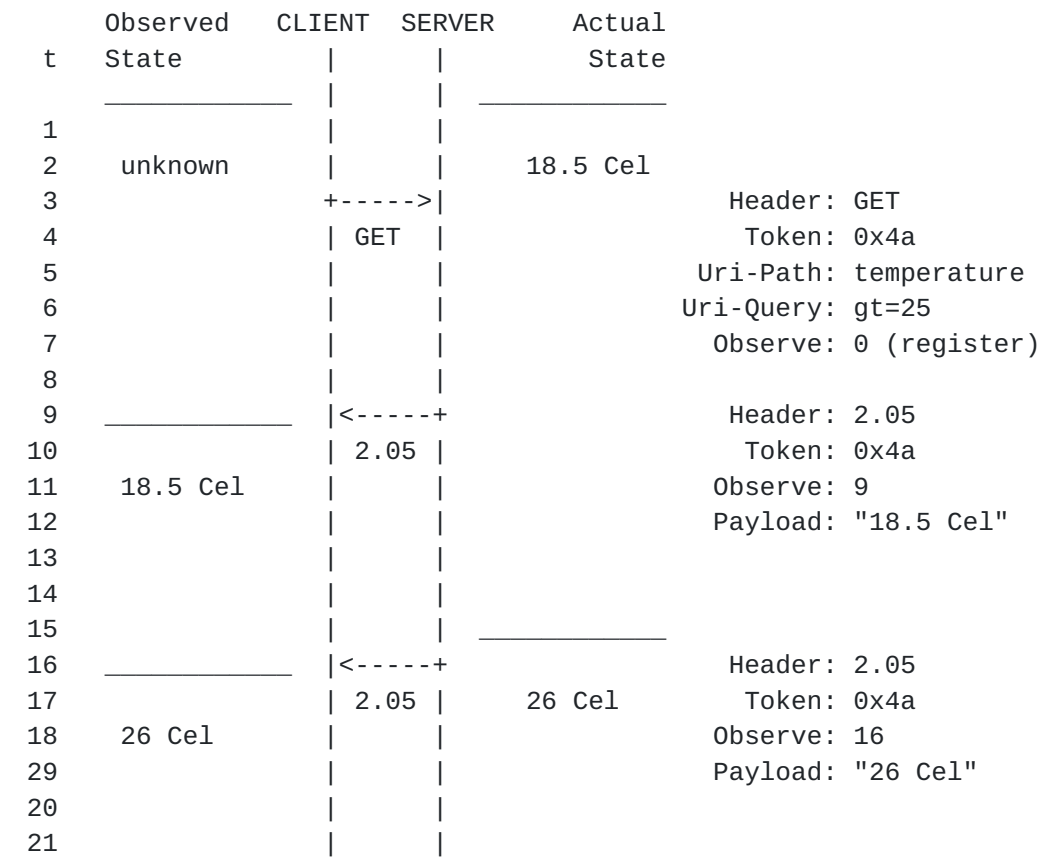
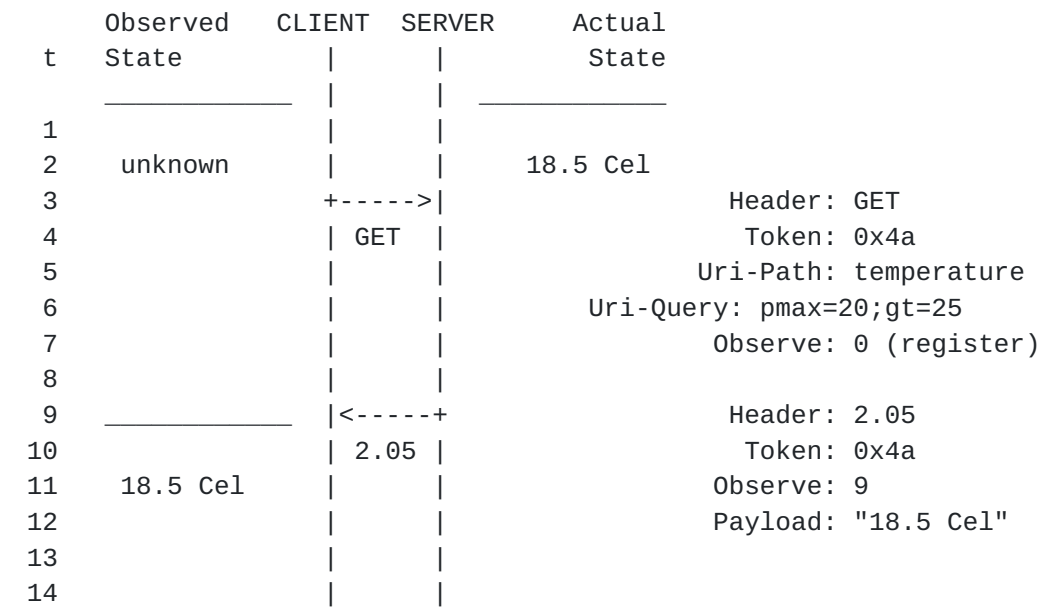


Figure 4: Client registers and receives one notification of the current state and one of a new state when it passes through the greater than threshold of 25.

A.4. Greater Than (gt) and Period Max (pmax) example



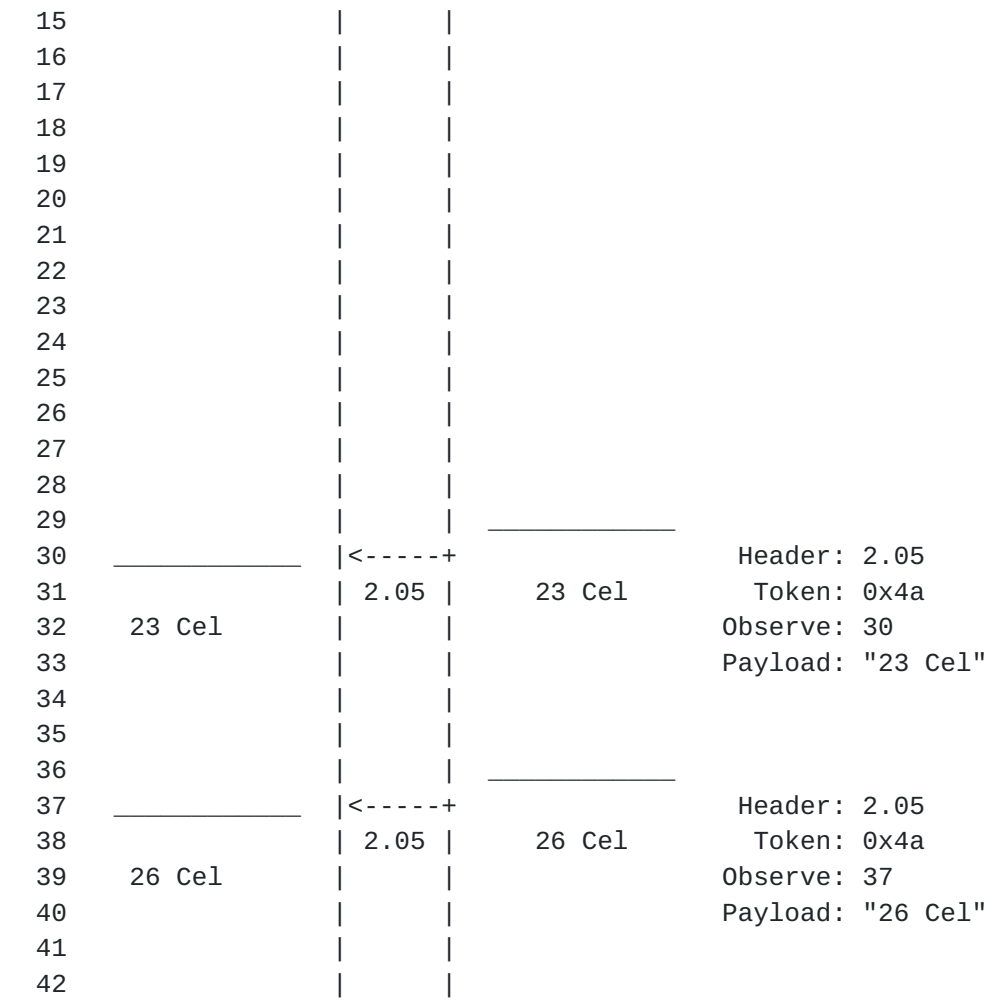


Figure 5: Client registers and receives one notification of the current state, one when pmax time expires and one of a new state when it passes through the greater than threshold of 25.

Authors' Addresses

Michael Koster
SmartThings
665 Clyde Avenue
Mountain View 94043
USA

Email: michael.koster@smartthings.com

Bilhanan Silverajan (editor)
Tampere University
Kalevantie 4
Tampere FI-33100
Finland

Email: bilhanan.silverajan@tuni.fi