open62541

Fraunhofer IOSB

IEEE ETFA 2024 – IEEE International Conference on Emerging Technologies and Factory Automation

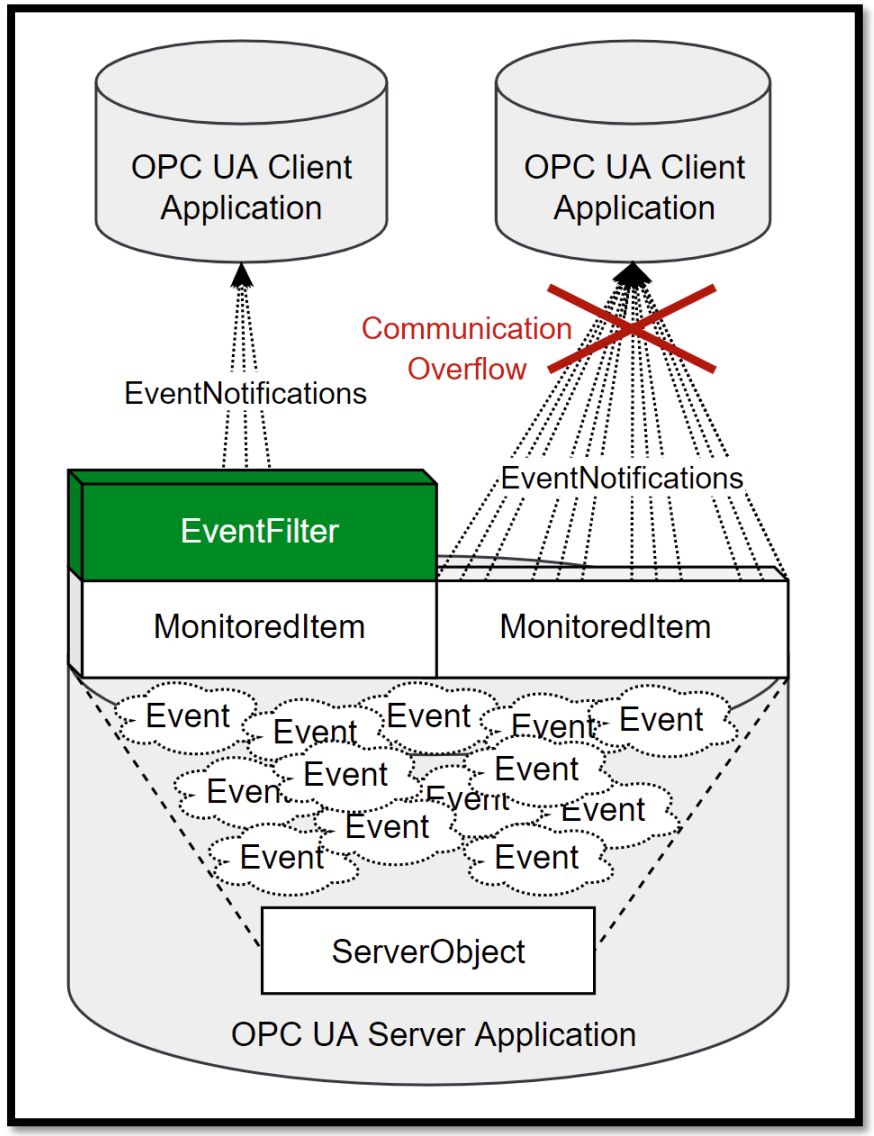
A Query Language for OPC UA Event Filters

Florian Düwel, Andreas Ebner, Julius Pfrommer

OPC UA EventFilter

Background

- OPC UA Events are "temporary OPC UA Objects" with the properties of the Event (fields)
  - Properties carry the information of an Event
  - *BaseEventType* defines standard properties, such as the Event's severity
  - Custom subtypes for additional properties
- Events are emitted by Objects in the information model
  - Events *bubble upwards* in the information model and are again emitted by each parent Object
- Clients access Events via the OPC UA Subscription mechanism
  - MonitoredItem attaches to an Object and listens for the Events
  - MonitoredItem defines an **EventFilter**
    - Which properties to transmit (**Select-Clause**, required)
    - Which event instances to transmit (**Where-Clause**, optional)



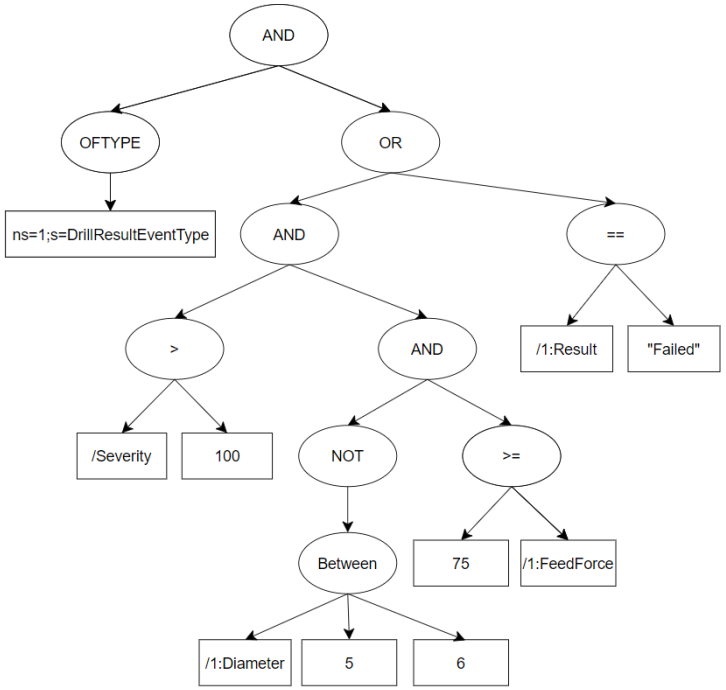
 © Fraunhofer IOSB

Fraunhofer IOSB

Example EventFilter Where-Clause

Much too hard for end-users to define

Logical Structure of the ContentFilter



Corresponding *ContentFilter* Data Type

Index	Operator	Operands
0	AND	ElementOperand 1, ElementOperand 2
1	OFTYPE	Literal ns=1;s=DrillResultEventType
2	OR	ElementOperand 3, ElementOperand 9
3	AND	ElementOperand 4, ElementOperand 5
4	>	SAO /Severity, Literal 100
5	AND	ElementOperand 6, ElementOperand 8
6	NOT	ElementOperand 7
7	BETWEEN	SAO /1:Diameter, Literal 5, 6
8	>=	Literal 75.5, SAO /1:FeedForce
9	==	SAO /1:Result, Literal ”Failed”

- Users have to supply this “bytecode”
- Difficult, verbose, error-prone, … → Not fun

➢ OPC UA EventFilters are very useful. But the tooling is not-so-great.

 © Fraunhofer IOSB

Fraunhofer IOSB

OPC UA EventFilter Query Language

Select-Clause

- Select-Clause is a comma separated list of SimpleAttributeOperands (SAO)

```
// select-clause
SELECT /1:OrderId,
       ns=1;s=DrillResultEventType/1:PartId#Value,
       /Severity
```

- Feature-complete human-readable syntax for expressing SimpleAttributeOperands

```
// SimpleAttributeOperand
ns=2;s=TruckEvent/3:Truck/5:Wheel#Value[1:4]
```

TypeDefinitionId | BrowsePath | AttributeId | IndexRange

- Default Values for a simplified definition of SAOs:
  - Default TypeDefinitionId is the BaseEventType
  - Default AttributeId is the Value attribute
  - Default IndexRange is undefined

SimpleAttributeOperand Definition

Name	Type
SimpleAttributeOperand	structure
typeDefinitionId	NodeId
browsePath []	QualifiedName
attributeId	IntegerId
indexRange	NumericRange

```
// select-clause
SELECT /Severity
```

```
// select-clause
SELECT i=2041/0:Severity#Value
```


© Fraunhofer IOSB

OPC UA EventFilter Query Language

Where-Clause (ContentFilter)

- A ContentFilter is a list of filter elements
- A filter element consists of an Operator and its Operands (arguments)
- Three kinds of Operands are allowed for EventFilter:
  - **SimpleAttributeOperand**: Points to a value inside the event instance
  - **ElementOperand**: Index of another filter element in the ContentFilter
  - **LiteralOperand**: A value in a Variant container

- Available **FilterOperators:**

EQUALS	INLIST
ISNULL	AND
GREATERTHAN	OR
LESSTHAN	CAST
GREATERTHANOREQUAL	LIKE
LESSTHANOREQUAL	OFTYPE
NOT	BITWISEAND
BETWEEN	BITWISEOR

- The filter elements are interpreted (back to front)
- The event is transmitted when the **first filter element** evaluates to **true**
  - Ternary Logic: true/false/null

Algorithm 1 OPC UA Event Filter Evaluation

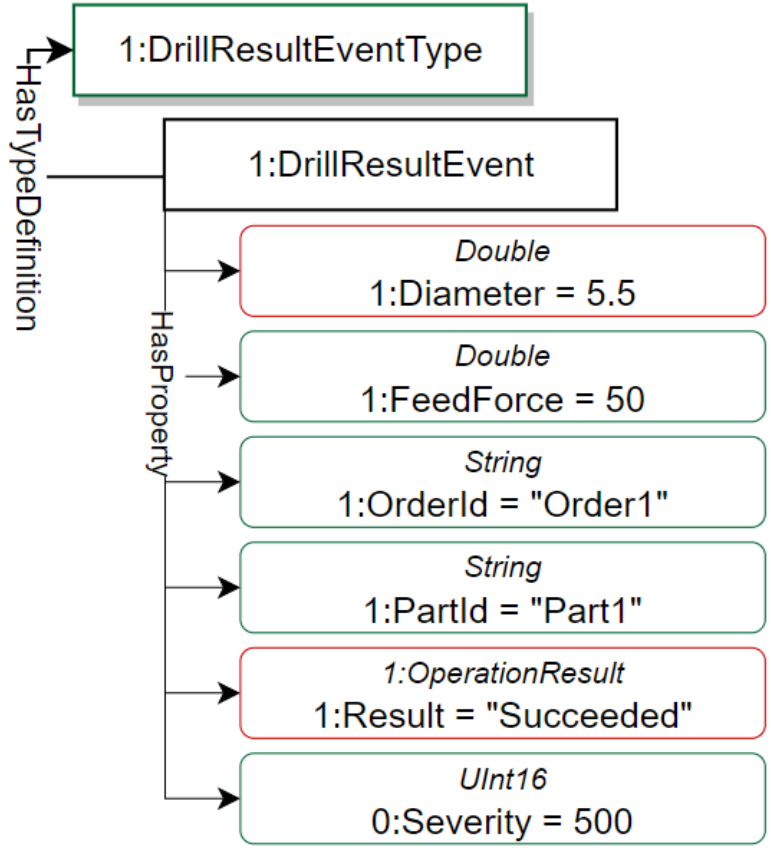
Input: List of ContentFilterElements E with length $|E| = N$

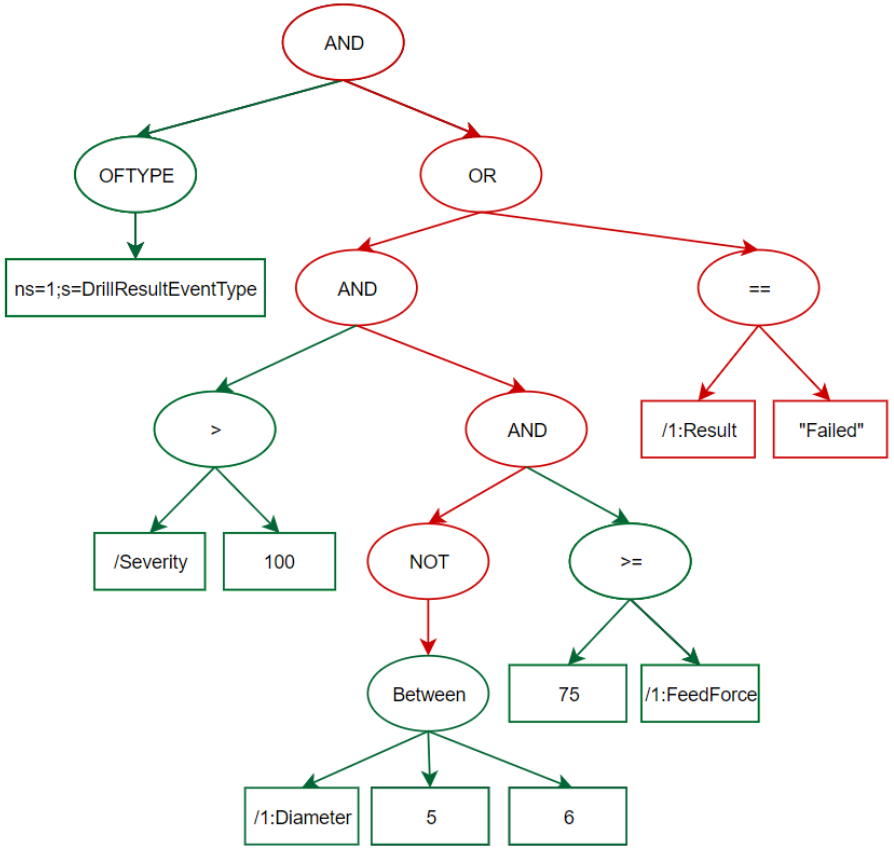
```
// Initialize results array of length N
R ← [n, ..., n]

// Resolve operand to a literal value
procedure resolve(o)
    if type(o) = ElementIndex then
        return R[o]
    else if type(o) = SimpleAttributeOperand then
        return read(o)
    return o // Already a literal

// Cast literal to ternary truth value
procedure ternary(o)
    if o ∈ {t, "true", "1"} then
        return t
    else if o ∈ {f, "false", "0"} then
        return f
    return n

// Evaluate operators from the back
for i = N − 1 ... 0 do
    (f, O) ← E[i]
    O ← [resolve(o) : o ∈ O]
    if f ∈ {and, or, not} then
        O ← [ternary(o) : o ∈ O]
    R[i] ← f(O)
return ternary(R[0])
```


© Fraunhofer IOSB

OPC UA Event Filters

Example ContentFilter Evaluation





- Evaluation of the ContentFilter fails since the criteria of the **NOT** Operator and the **EQUALS** Operator are not matched
  - 1:Diameter = 5.5 vs. 1:Diameter **NOT** BETWEEN 5 and 6
  - 1:Result = “Failed” vs. 1:Result = “Succeeded”

 © Fraunhofer IOSB

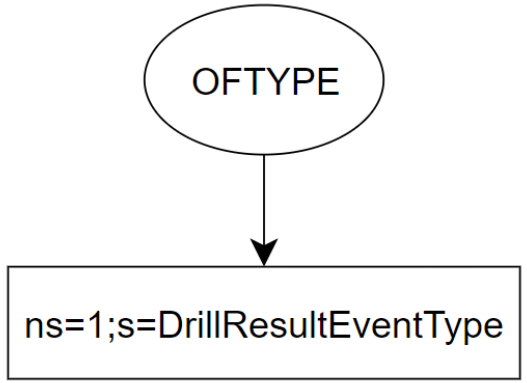
Fraunhofer IOSB

OPC UA Query Language

Where-Clause

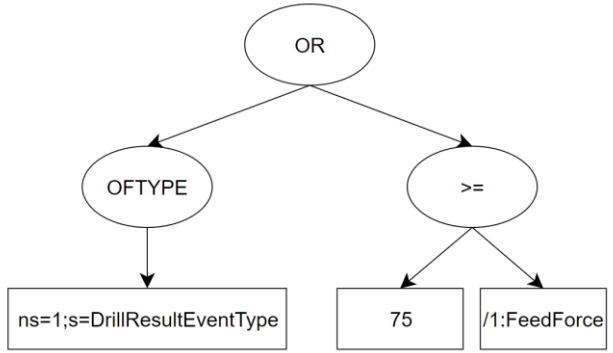
- Creates the ContentFilter and thus, the filter criteria to be evaluated at an Event's occurence
- Where-Clause consist of at least one FilterOperator

```
// where-clause
WHERE
OFTYPE ns=1;s=DrillResultEventType
```



- Operators can be chained together to span the tree-like structure of the ContentFilter
- Index of each Operator is extracted from the query

```
// where-clause
WHERE
OFTYPE ns=1;s=DrillResultEventType OR
(75 >= /1:FeedForce)
```

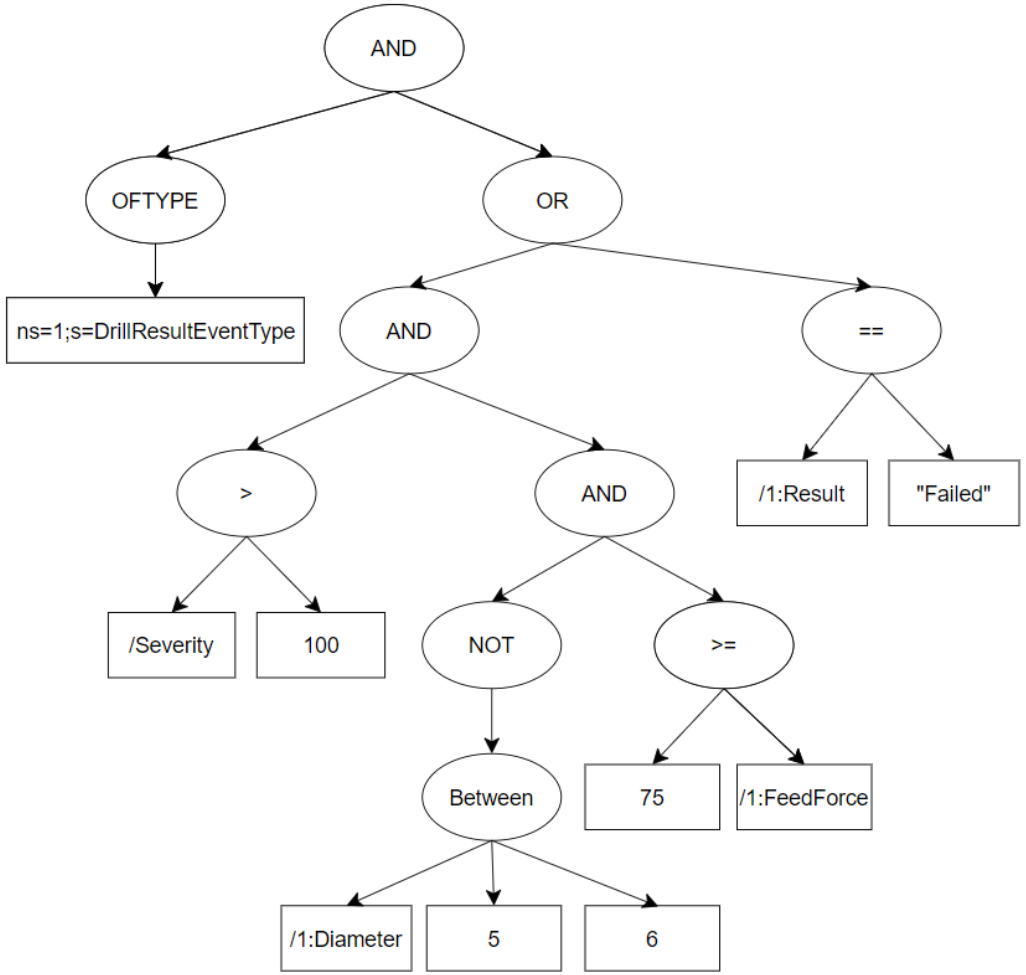


 © Fraunhofer IOSB

Fraunhofer IOSB

OPC UA Query Language

Query Example




```
// select-clause
SELECT /1:OrderId,
       ns=1;s=DrillResultEventType/1:PartId#Value,
       /Severity

// where-clause
WHERE OFTYPE ns=1;s=DrillResultEventType AND
      (( /Severity > 100 AND
      ((NOT (/1:Diameter BETWEEN [5,6]))
      AND (75 >= /1:FeedForce)))
      OR (/1:Result == "Failed"))
```

- Query to construct the EventFilter on the left side

 © Fraunhofer IOSB

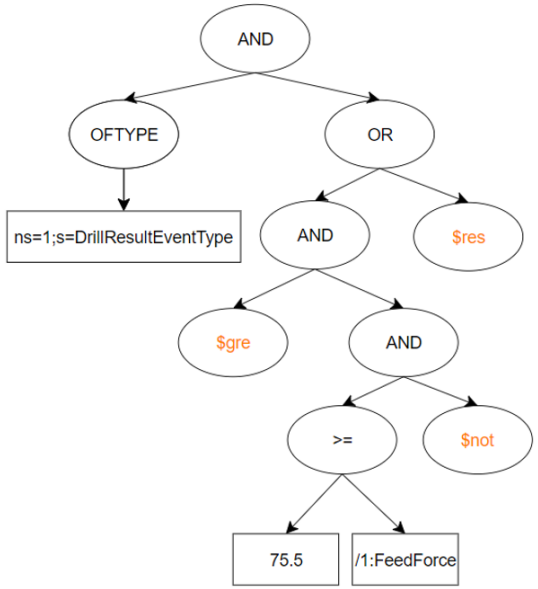
Fraunhofer IOSB

OPC UA Query Language

For-Clause

- The For-Clause only exists in the EventFilter query language → gets compiled away in the background
  - Allow reuse of definitions and cross-references between filter elements (filter elements must form an acyclic graph)
- Variable-like assignments of filter expressions
  - Can resolve to single Operators, Operands or entire "sub-graphs" of the ContentFilter

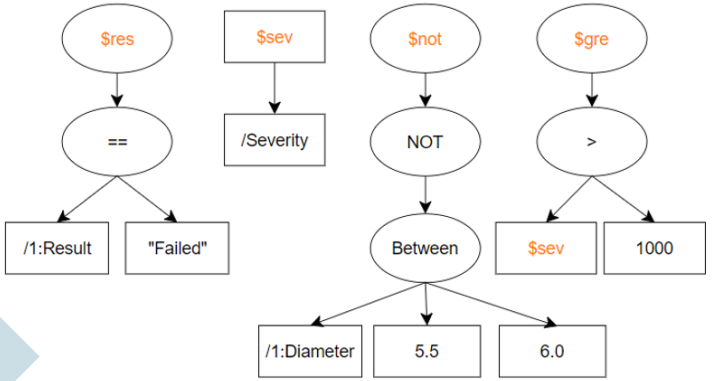
Where-Clause



Query-Statement

```
// select-clause
SELECT /1:OrderId,
       ns=1;s=DrillResultEventType/1:PartId#Value,
       $sev

// where-clause
WHERE OFTYPE ns=1;s=DrillResultEventType AND
      ((( $gre AND $not)
      AND (75 >= /1:FeedForce))
      OR $res)

// for-clause
FOR $sev := /Severity,
    $gre := $sev > 100,
    $res := /1:Result == "Failed",
    $not := NOT (/1:Diameter BETWEEN [5,6])
```

For-Clause

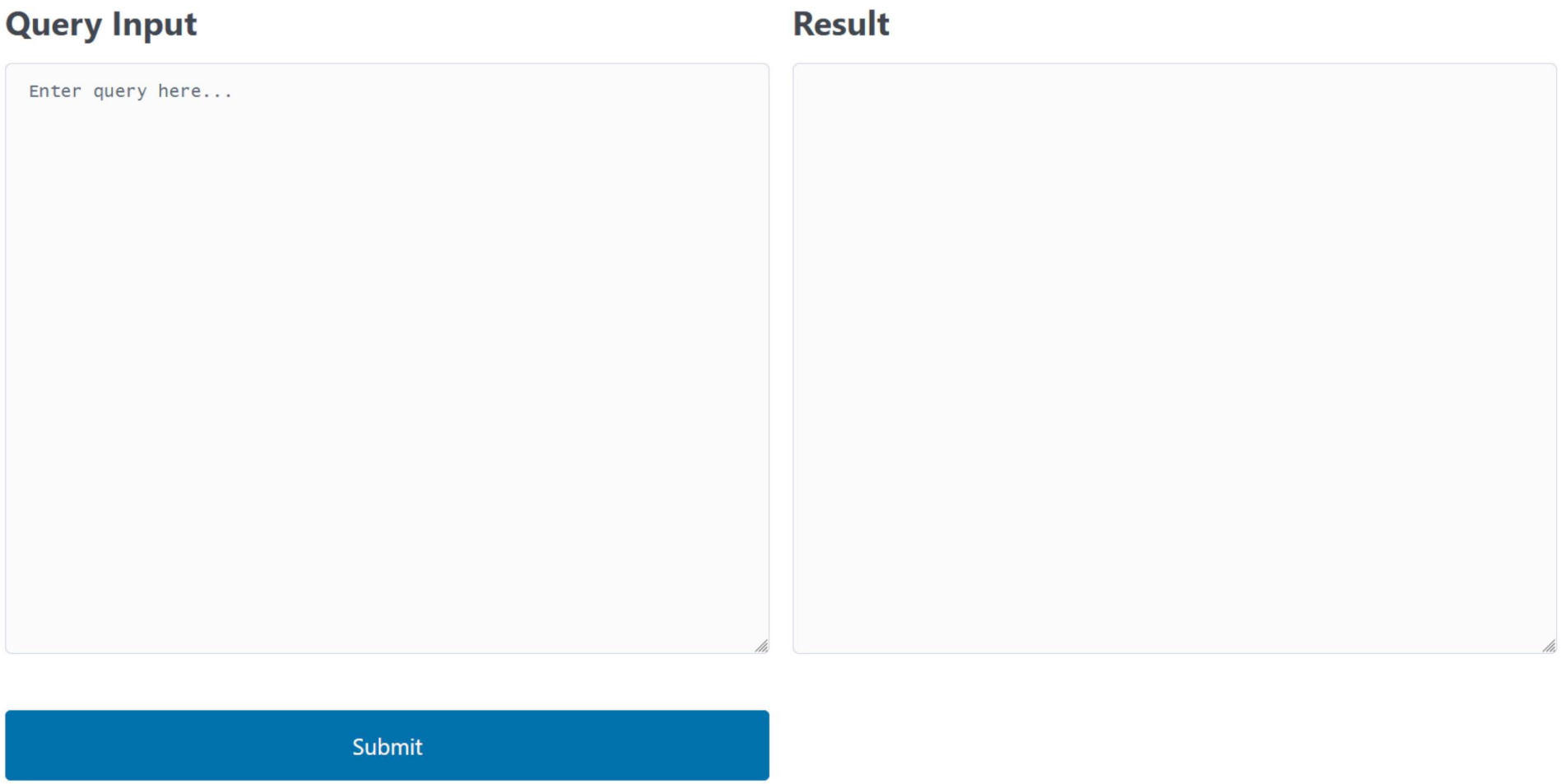


 © Fraunhofer IOSB

Fraunhofer IOSB

Demonstration

https://www.open62541.org/query-http/index.html



 © Fraunhofer IOSB

Fraunhofer IOSB

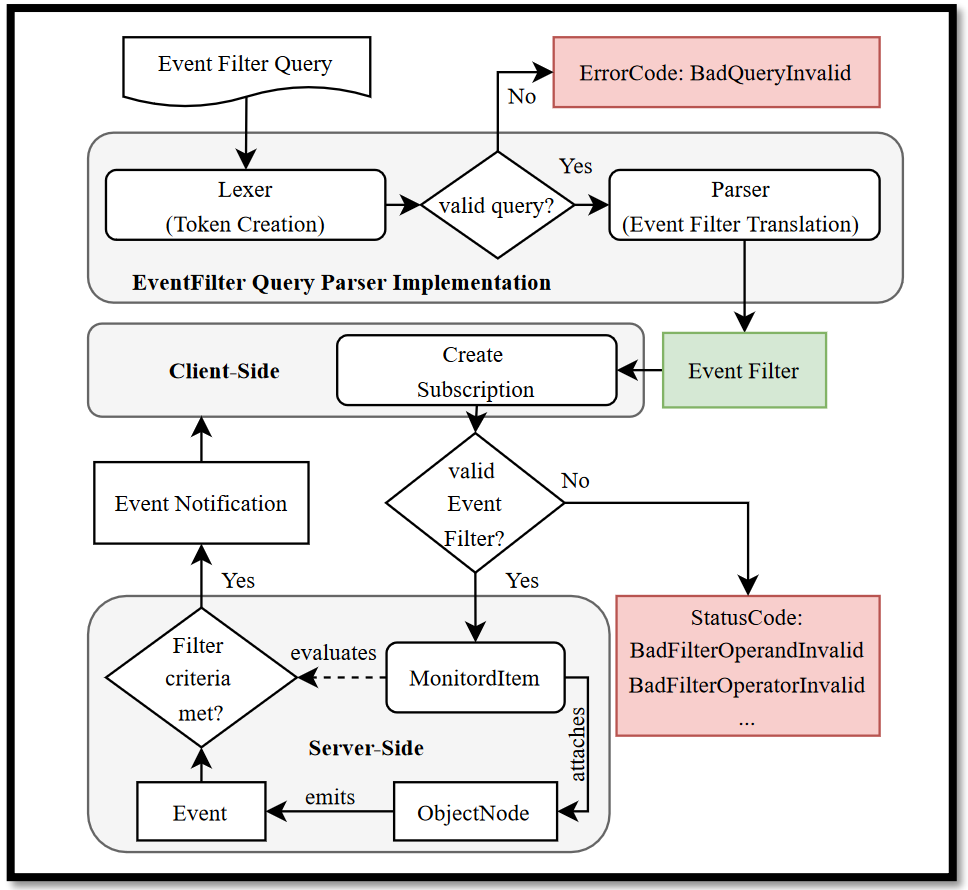
Summary

Main Message

- OPC UA Events are great, OPC UA EventFilters are great
- EventFilters are difficult to write in their "native form"
- The presented EventFilter Query Language makes OPC UA Events fun!

How can you use the EventFilter Query Language?

- The HTML5 implementation generates EventFilter in the JSON format
  - Output can be used with every major OPC UA SDK
  - https://www.open62541.org/query-http/index.html
- Implementation as part of open62541
  - https://github.com/open62541/open62541/blob/master/src/util/ua_eventfilter_grammar.y



- Concept of how to apply the EventFilter Query Language to existing OPC UA SDKs

 © Fraunhofer IOSB

Fraunhofer IOSB

Contact

M.Sc. Florian Düwel
Department: Cognitive Industrial Systems
Group: Adaptive Production Systems
Florian.duewel@iosb.fraunhofer.de