Using Lua for BACnet OEM solutions in building automation

How to brew coffee with Lua and BACnet

Robert Schlephorst
schlephorst@se-elektronic.de

SE Elektronic GmbH

Lua Workshop 2013

about SE Elektronic GmbH

- company founded 1983 (30 years)
- develops building automation products: sensors, actors, building controller
- development & production in south Germany

What is BACnet?

- Building Automation and Control Networks
- ASHRAE/ANSI 135-2010, ISO 16484-5
- Communication between devices and building management system
- Provide BMS with information
- Sensors, Actors
- AirConditioning, Elevators, Escalators, ...

<table>
<tr><td colspan="4">BACnet Application Layer</td></tr>
<tr><td colspan="4">BACnet Network Layer</td></tr>
<tr><td>BVLC</td><td>MS/TP</td><td>LON</td><td>PTP</td></tr>
<tr><td>UDP/IP</td><td>RS-485</td><td></td><td>RS-232</td></tr>
</table>

Why BACnet?
BACnet compared to other protocols

e.g. *Modbus*

- designed for resource limited devices
- master/slave communication
- read/write registers/file records
- datatypes limited (simple like integer, float)
- no data descriptions readable
  $\longrightarrow$ need device documentation for integration

Why BACnet?
BACnet compared to other protocols

BACnet

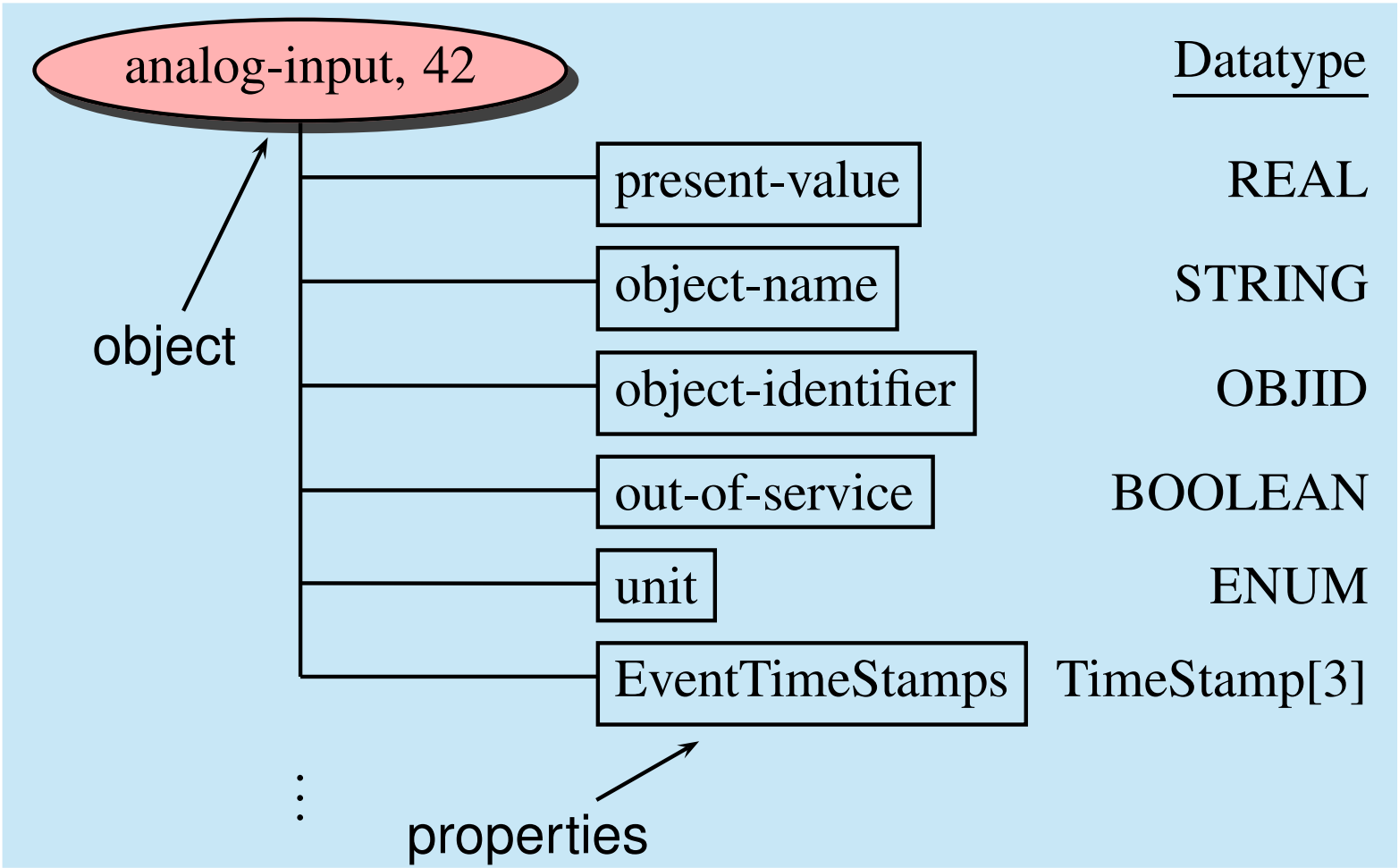
- data/features structured as objects
- developers like objects
- searches possible, objects browsable
- Event & ChangeOfValue Notifications
- BACnet network routable
- conformance tests $\longrightarrow$ cross-vendor interoperability

Why BACnet?
BACnet compared to other protocols

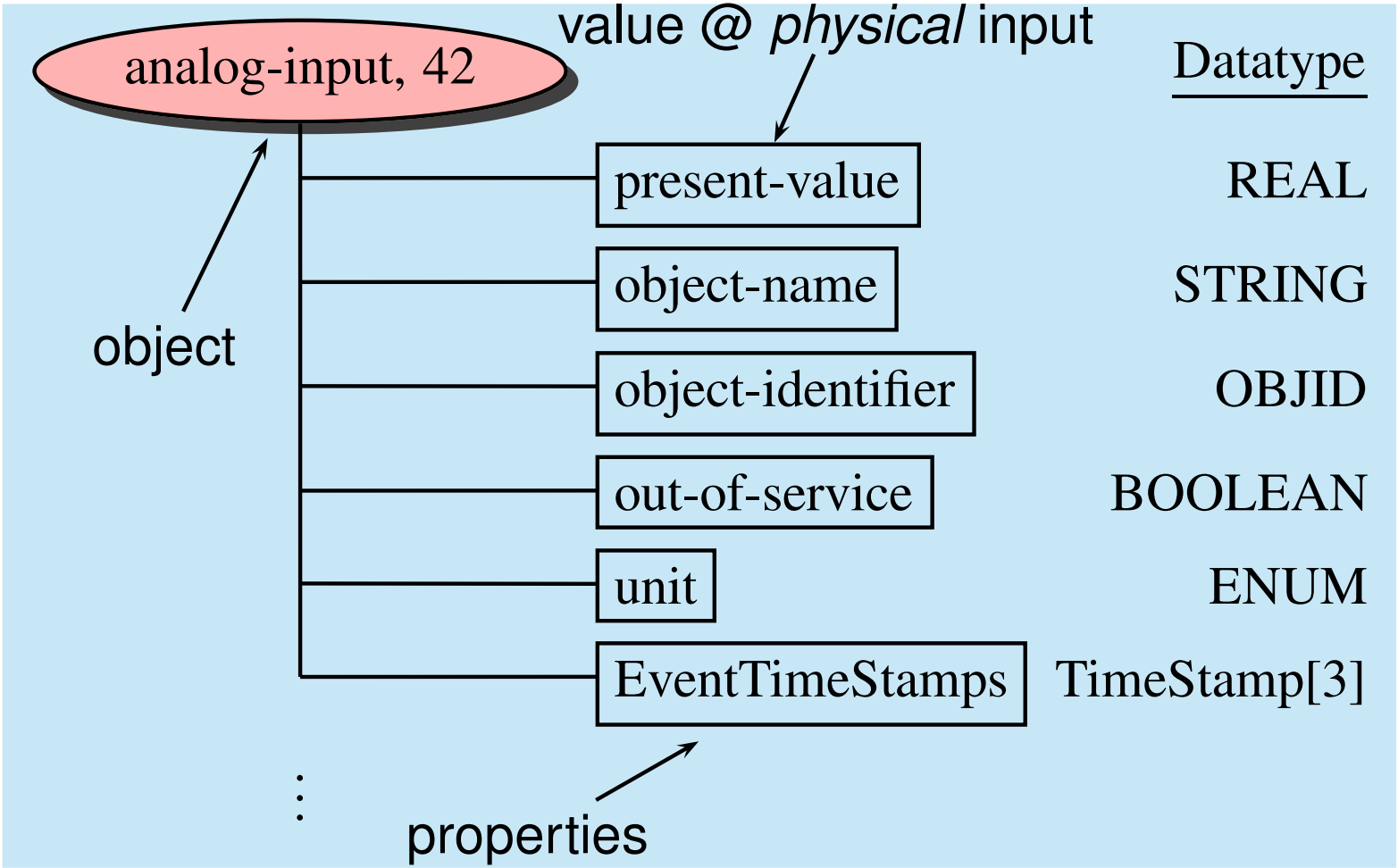
BACnet

- data/features structured as objects
- developers like objects
- searches possible, objects browsable
- Event & ChangeOfValue Notifications
- BACnet network routable
- conformance tests $\longrightarrow$ cross-vendor interoperability
- high complexity
- many optional features
- stack partitioned into functional blocks

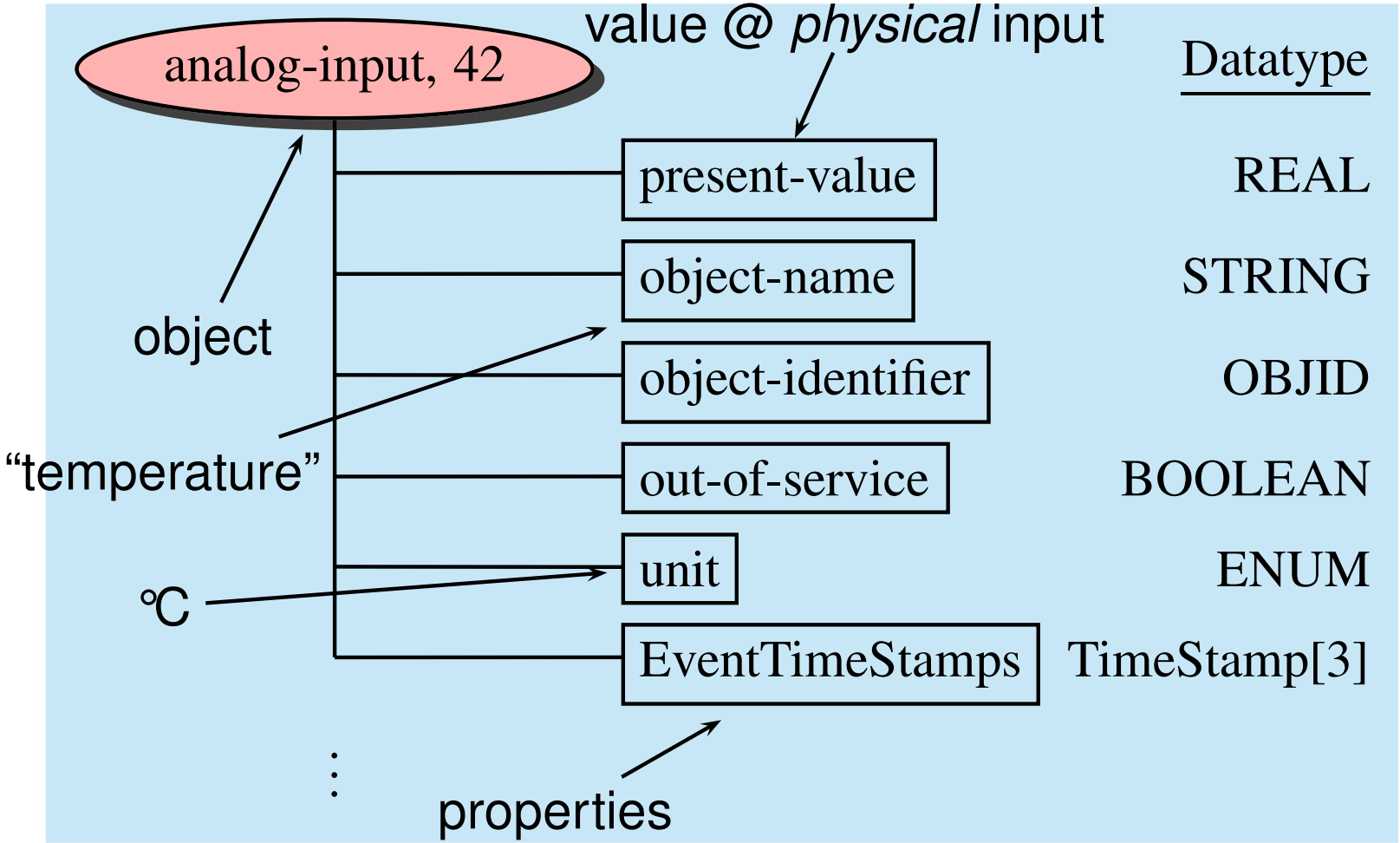
BACnet is all about objects



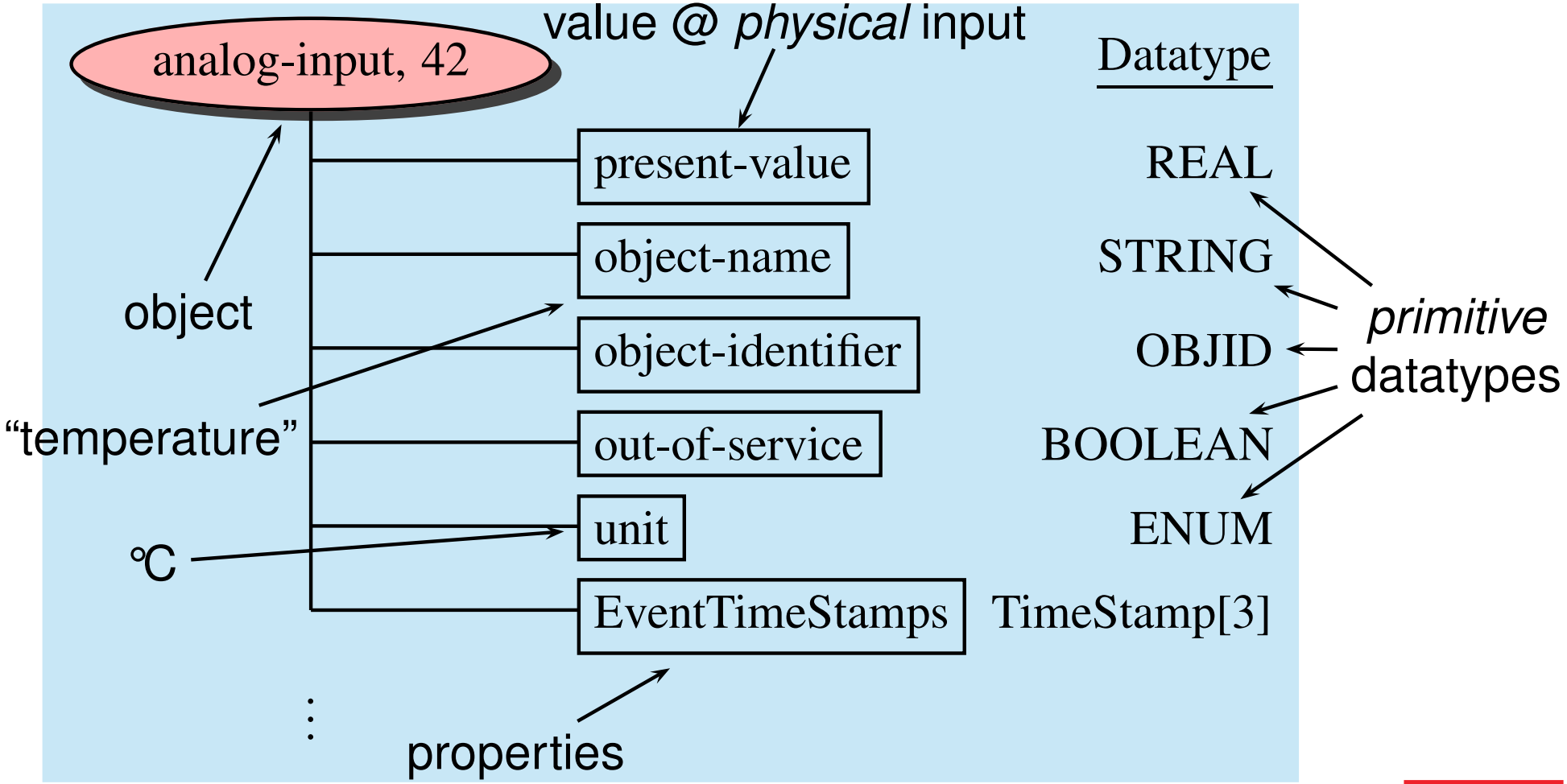
BACnet is all about objects



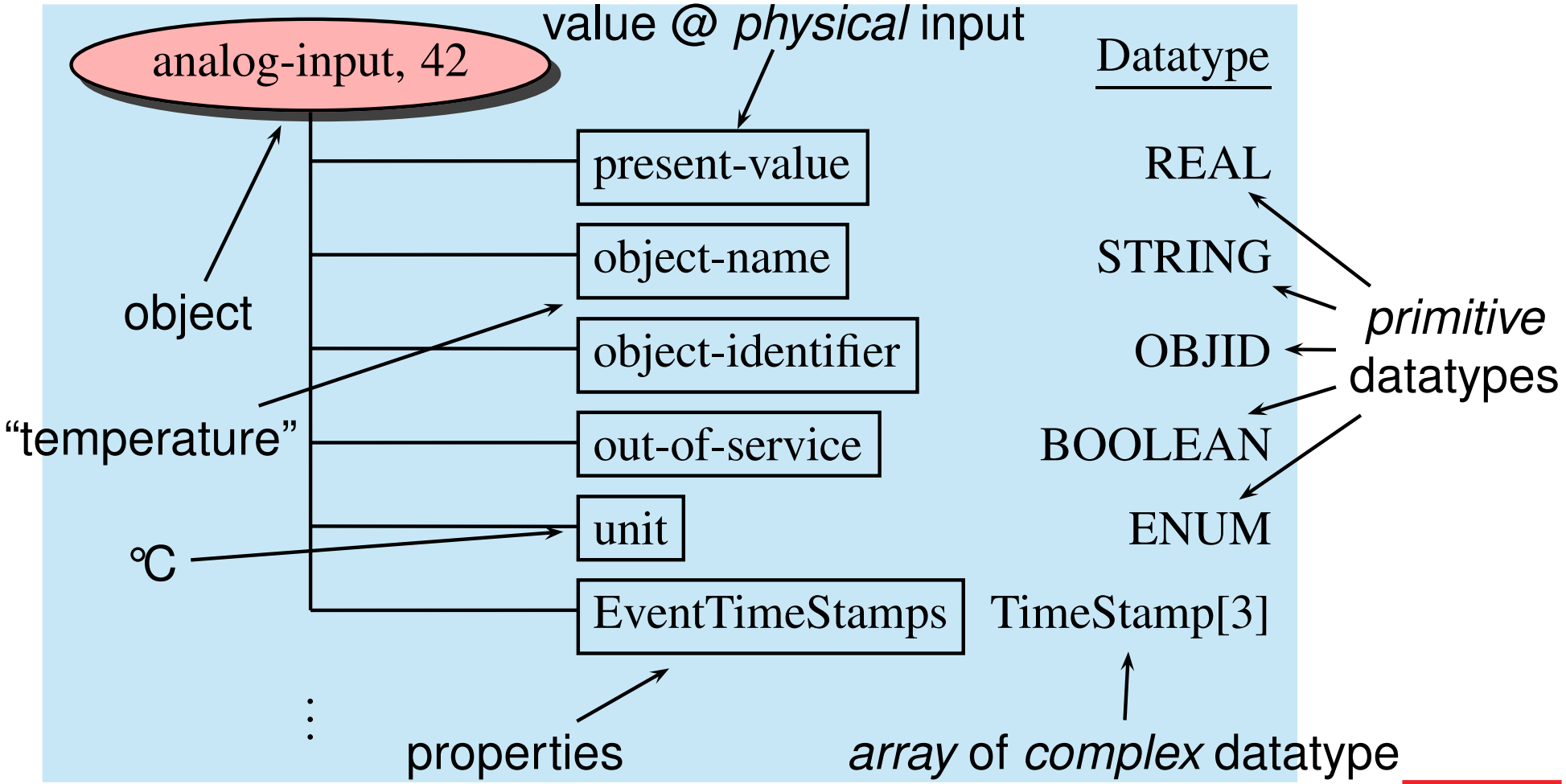
BACnet is all about objects



BACnet is all about objects



BACnet is all about objects



Application Layer

Communication Between Devices

- Object Access Services (Read, Write, Create, ...)
- Alarm & Event Services (EventNotification, ...)
- Device Management (Backup, Restore, ...)

The BACnet Coffee Machine

UberCoffee 2013™

BACnet Requirements

- full BACnet integration (Coffee Management System)
- control water temperature → Loop (PID)
- log coffee consumption → Trendlog

The BACnet Coffee Machine

UberCoffee 2013™

BACnet Requirements

- full BACnet integration (Coffee Management System)
- control water temperature → Loop (PID)
- log coffee consumption → Trendlog

OEM Application

- send SMS on low coffee level

The BACnet Coffee Machine

UberCoffee 2013™

BACnet Requirements

- full BACnet integration (Coffee Management System)
- control water temperature → Loop (PID)
- log coffee consumption → Trendlog

OEM Application

- send SMS on low coffee level
- no more beans: send Email to Java™support

The BACnet Coffee Machine

UberCoffee 2013™

BACnet Requirements

- full BACnet integration (Coffee Management System)
- control water temperature → Loop (PID)
- log coffee consumption → Trendlog

OEM Application

- send SMS on low coffee level
- no more beans: send Email to procurement

The BACnet Coffee Machine

UberCoffee 2013™

BACnet Requirements

- full BACnet integration (Coffee Management System)
- control water temperature → Loop (PID)
- log coffee consumption → Trendlog

OEM Application

- send SMS on low coffee level
- no more beans: send Email to procurement
- web frontend

The BACnet Coffee Machine

UberCoffee 2013™

BACnet Requirements

- full BACnet integration (Coffee Management System)
- control water temperature → Loop (PID)
- log coffee consumption → Trendlog

OEM Application

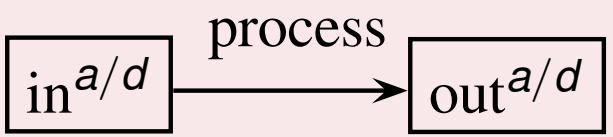
- send SMS on low coffee level
- no more beans: send Email to procurement
- web frontend
- itegrated display with customizable content

SE-ELEKTRONIC GMBH

BACnet demands new processing strategies

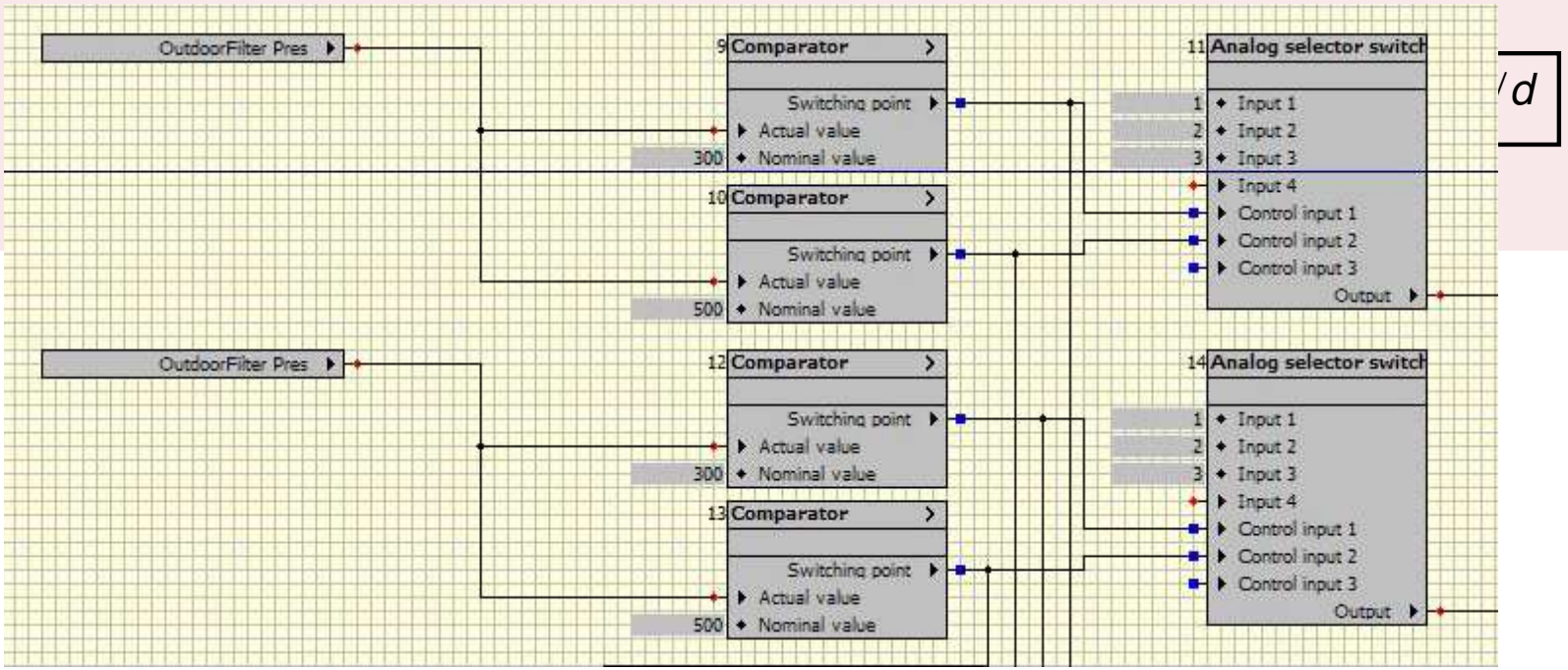
Legacy Programmable Logic

- application w/ cyclic I/O:
  read - process - write
- data points (digital & analog)



BACnet demands new processing strategies

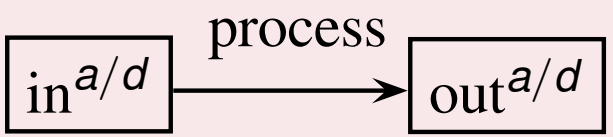
Legacy Programmable Logic



BACnet demands new processing strategies

Legacy Programmable Logic

- application w/ cyclic I/O:
  read - process - write
- data points (digital & analog)



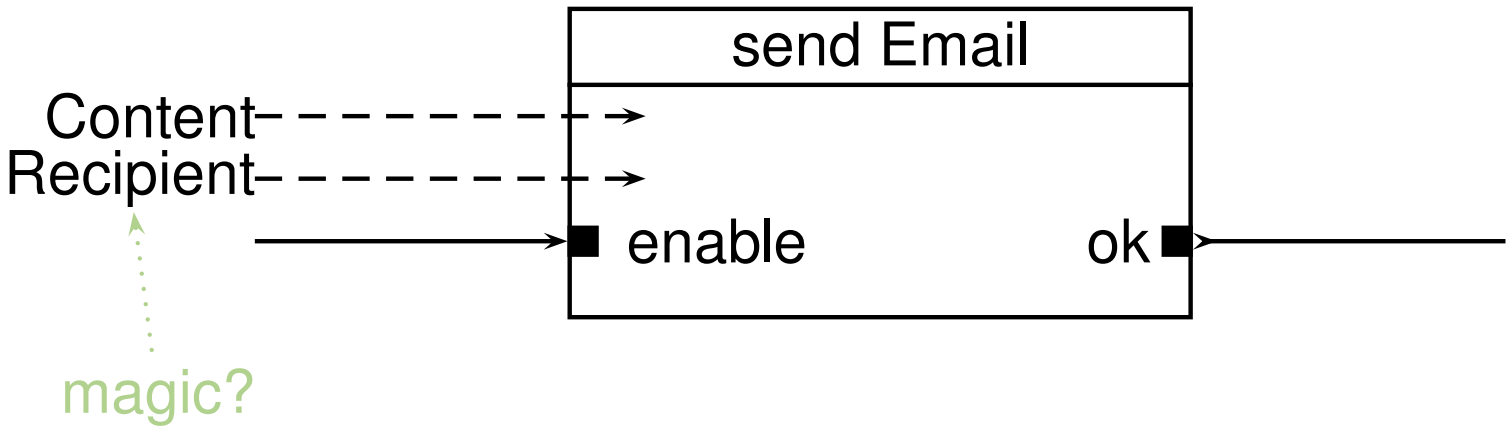
Problems

- translate between BACnet datatypes and data points
- impossible to read and/or modify complex properties
- can't use BACnet Notifications: ChangeOfValue & Events

SE-ELEKTRONIC GMBH

Legacy Solution

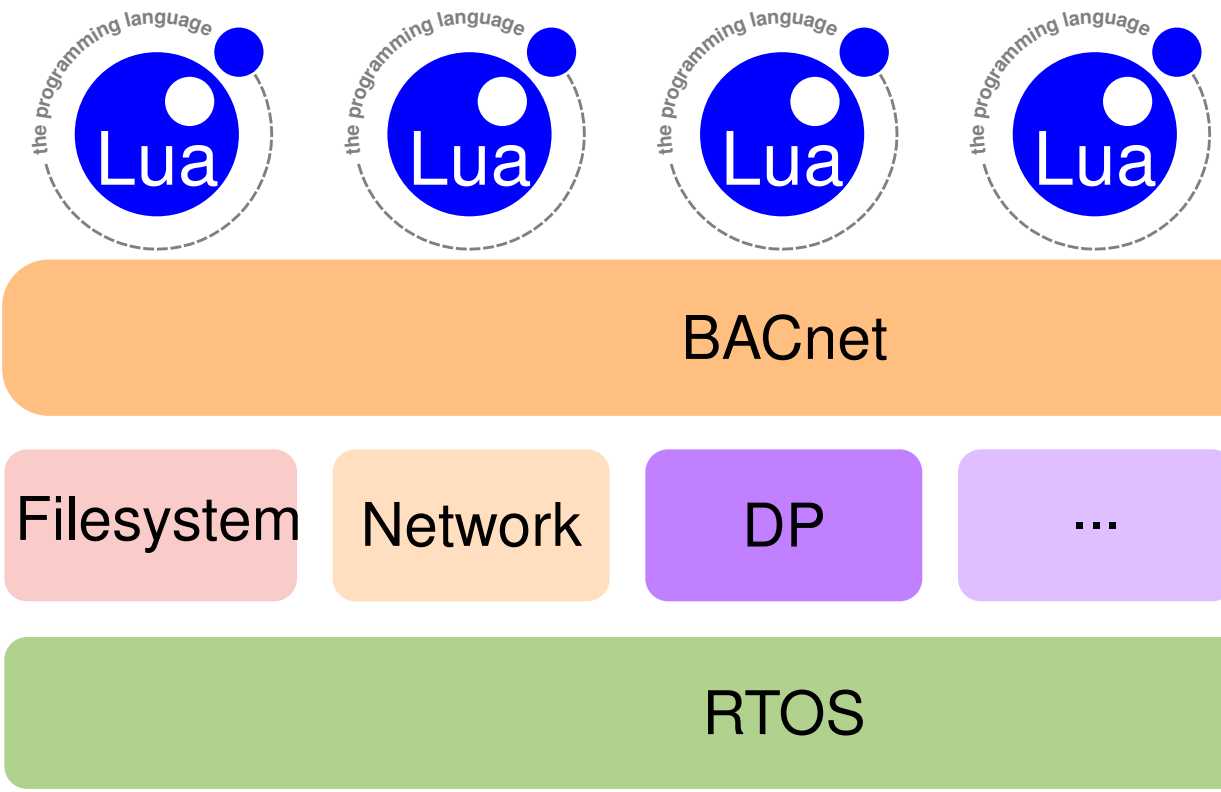
- create special data points/modules which trigger actions
- special (OEM) logic hardcoded in firmware



Solution

- Lua application for OEM tasks
- BACnet API for Lua
- BACnet datatypes for Lua
- asynchronous processing

Lua Program Object



Lua Program Object
Runtime

- Lua 5.2 VM
- Memory Quota $\longrightarrow$ restrict memory consumption
- Watchdog $\longrightarrow$ detect e.g. infinite loops
- Shell access and execution control via Telnet
- Loader for own package format
- is BACnet object $\longrightarrow$ configuration and control with BACnet

Lua Program Object
C-Libs/APIs

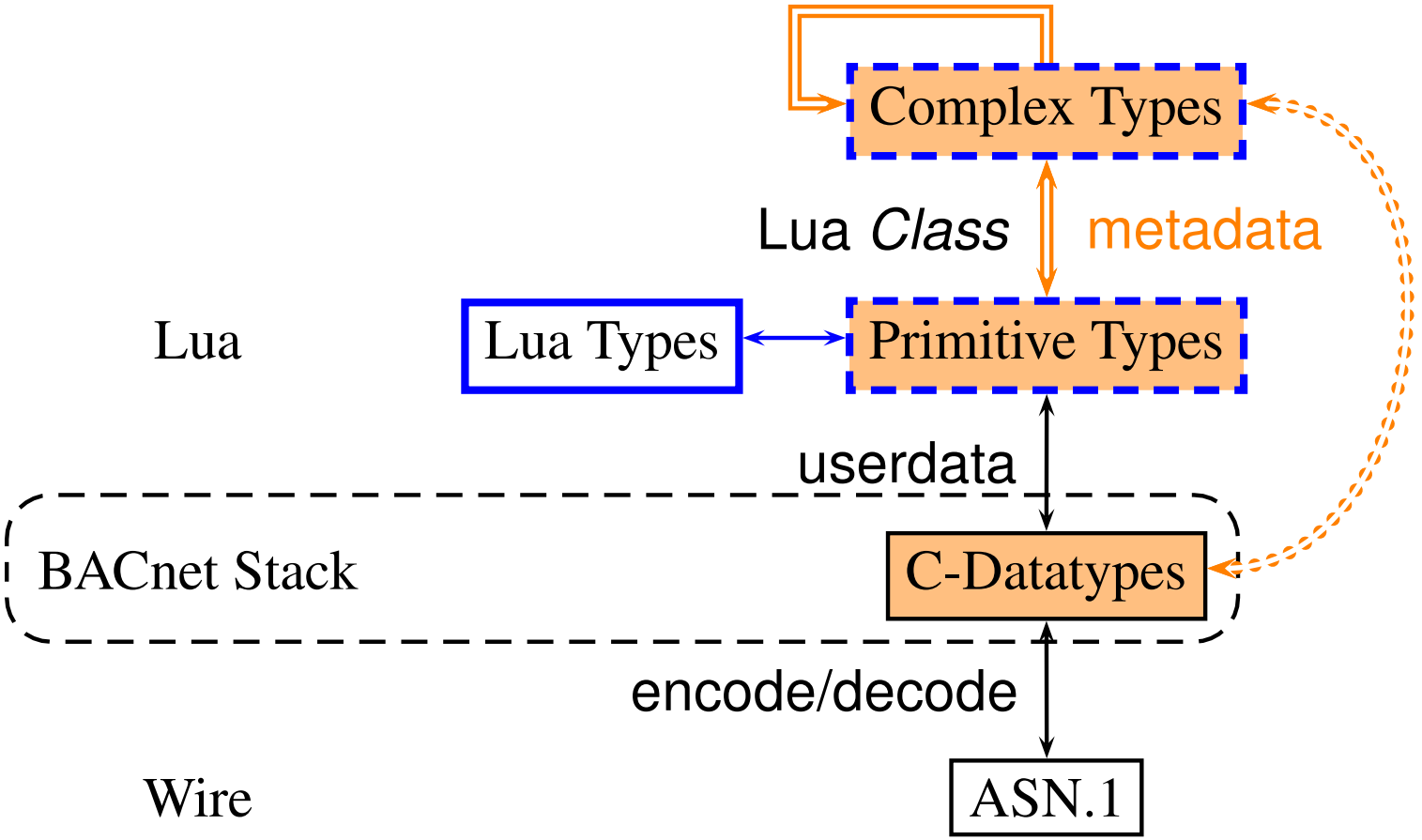
Libs

- Baselibs
- LuaSocket
- LuaFileSystem

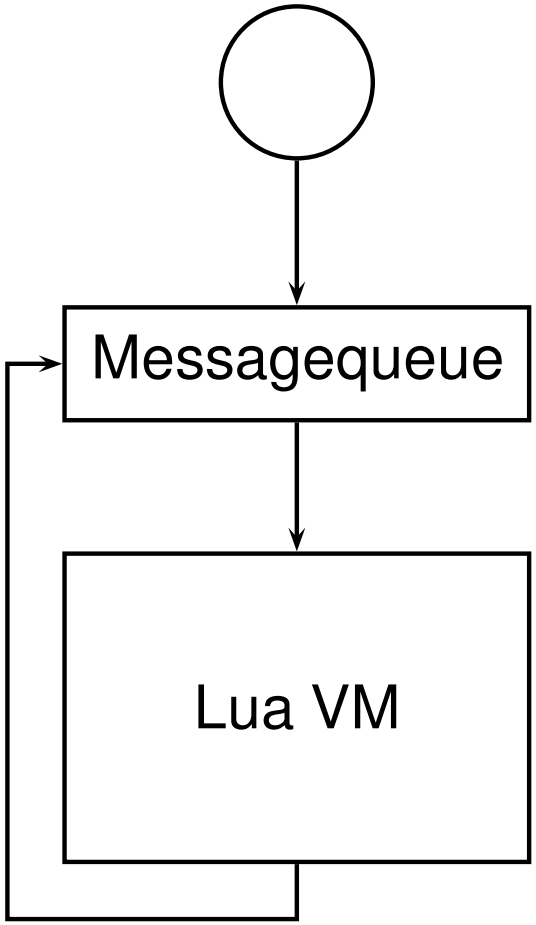
APIs

- Data Points
- Timer
- System Configuration
- User Accounts
- GUI
- BACnet

Lua Program Object
Type System



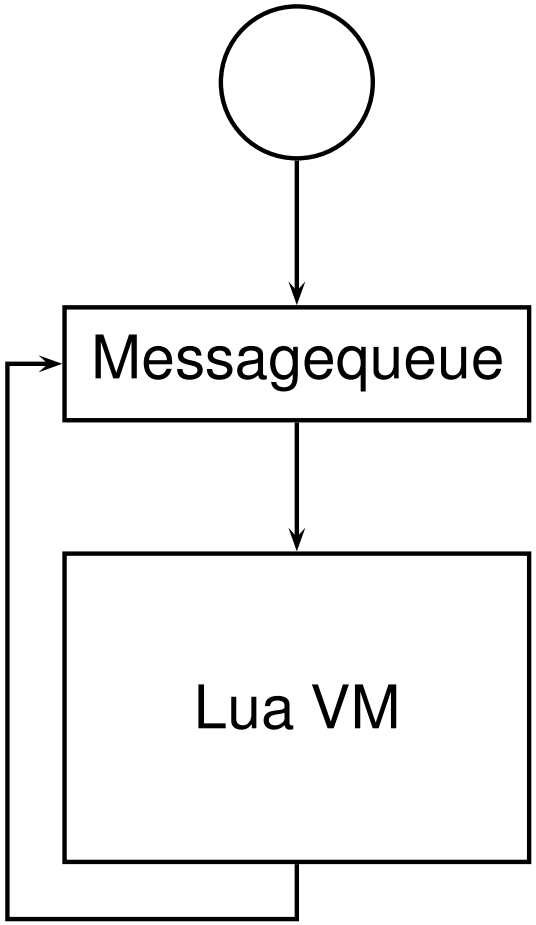
Asynchronous Processing



Callback Functions

- Timers
- System Events
- BACnet COV & BACnet Events
- BACnet Service Responses

Asynchronous Processing

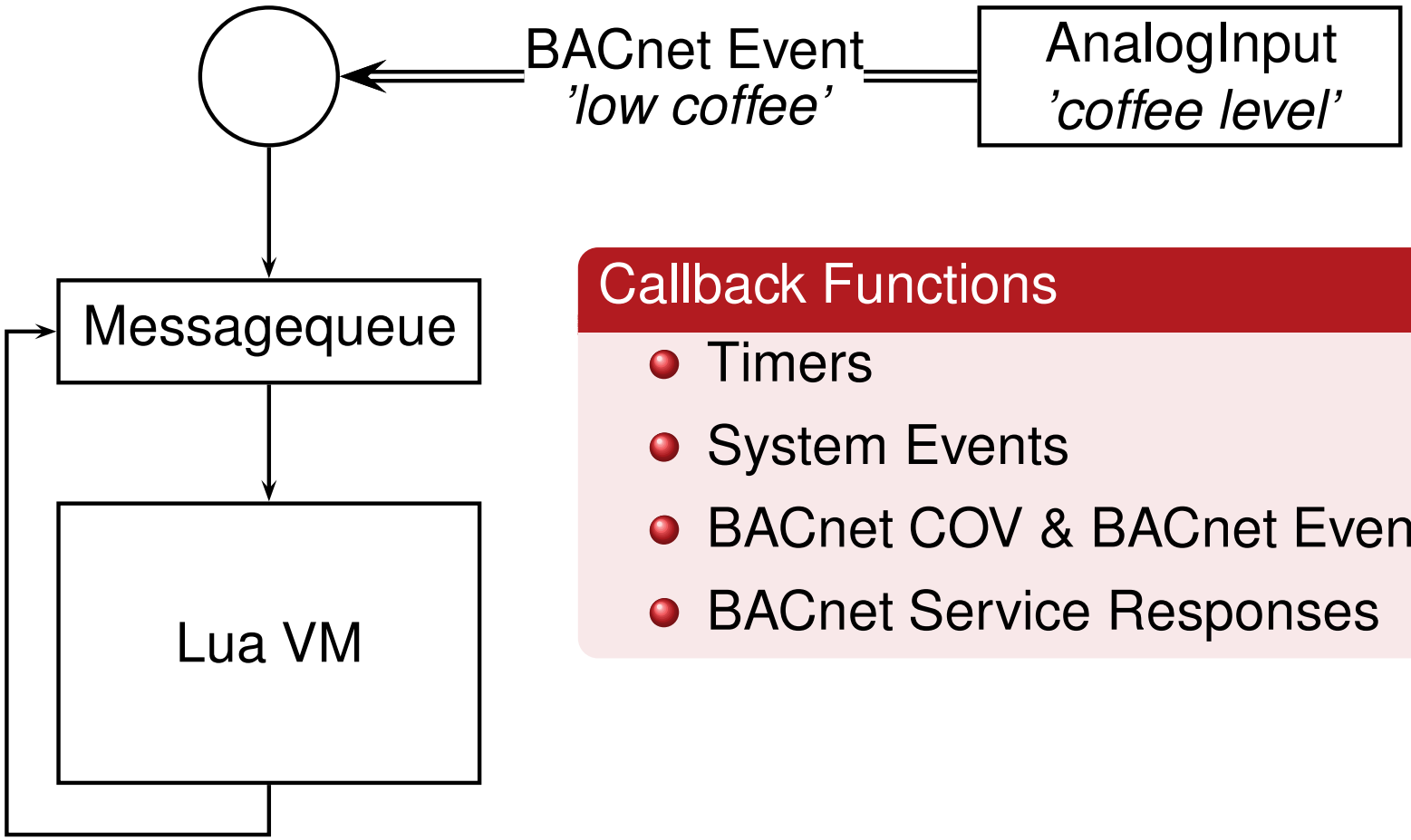


AnalogInput
'coffee level'

Callback Functions

- Timers
- System Events
- BACnet COV & BACnet Events
- BACnet Service Responses

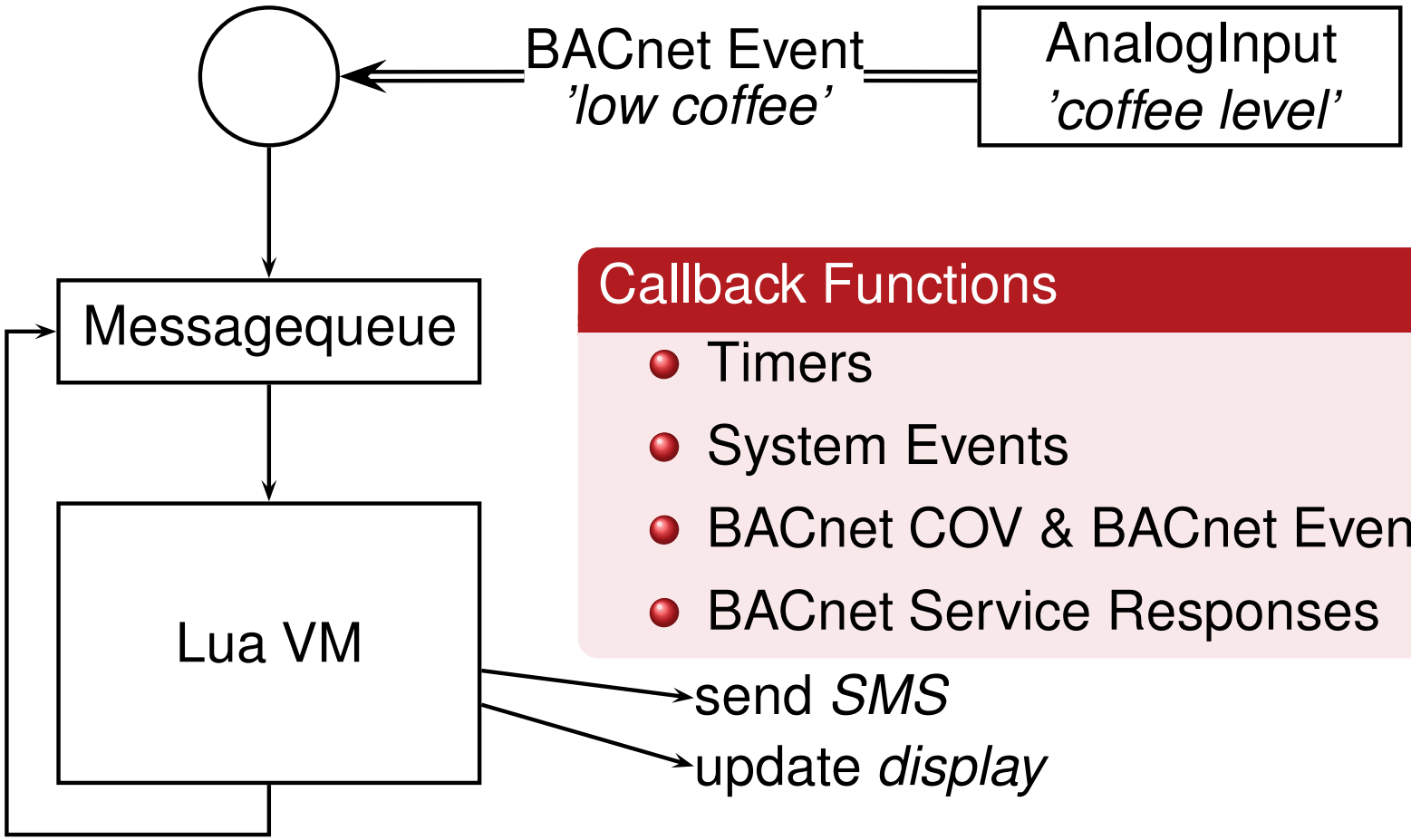
Asynchronous Processing



Callback Functions

- Timers
- System Events
- BACnet COV & BACnet Events
- BACnet Service Responses

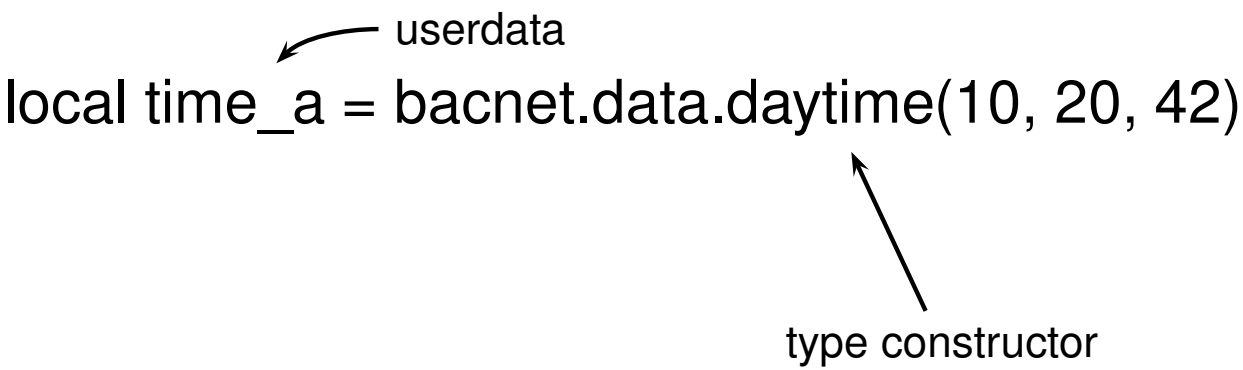
Asynchronous Processing



Callback Functions

- Timers
- System Events
- BACnet COV & BACnet Events
- BACnet Service Responses

Example
BACnet Datatype Construction in Lua



Example
BACnet Datatype Construction in Lua

```
local time_a = bacnet.data.daytime(10, 20, 42)
local time_b = bacnet.data.daytime("10:19:42")
```

overloaded for convenience

Example
BACnet Datatype Construction in Lua

```
local time_a = bacnet.data.daytime(10, 20, 42)
local time_b = bacnet.data.daytime("10:19:42")
local h,m,s = time_a()
```

convert to Lua type(s)

Example
BACnet Datatype Construction in Lua

```
local time_a = bacnet.data.daytime(10, 20, 42)
local time_b = bacnet.data.daytime("10:19:42")
local h,m,s = time_a()
local early = time_b < time_a
```

Example
BACnet Datatype Construction in Lua

```
local time_a = bacnet.data.daytime(10, 20, 42)
local time_b = bacnet.data.daytime("10:19:42")
local h,m,s = time_a()
local early = time_b < time_a
local datetime = bacnet.data.date_time(
    bacnet.data.daytime("10:41:00"), bacnet.data.date(2013,11,24))
```

complex type is composition of data

time member

date member

Example
BACnet Datatype Construction in Lua

```
local time_a = bacnet.data.daytime(10, 20, 42)
local time_b = bacnet.data.daytime("10:19:42")
local h,m,s = time_a()
local early = time_b < time_a
local datetime = bacnet.data.date_time(
    bacnet.data.daytime("10:41:00"), bacnet.data.date(2013,11,24))
for member, value in pairs(datetime) do
    print(member, value) end
```

again metatable magic

Example
BACnet Datatype Construction in Lua

```
local time_a = bacnet.data.daytime(10, 20, 42)
local time_b = bacnet.data.daytime("10:19:42")
local h,m,s = time_a()
local early = time_b < time_a
local datetime = bacnet.data.date_time(
    bacnet.data.daytime("10:41:00"), bacnet.data.date(2013,11,24))
for member, value in pairs(datetime) do
    print(member, value) end
local constr_datetime = datetime:typefunc()
```

identical

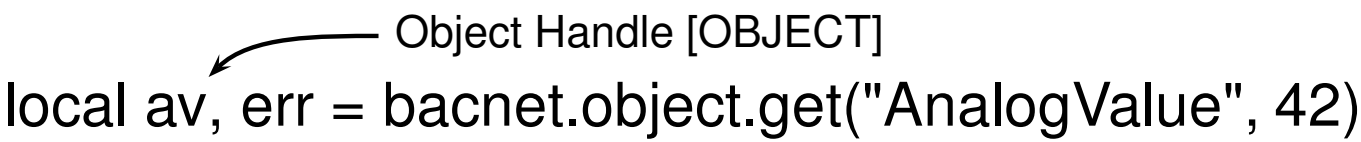
retrieve constructor from type

Example
BACnet Datatype Construction in Lua

```
local time_a = bacnet.data.daytime(10, 20, 42)
local time_b = bacnet.data.daytime("10:19:42")
local h,m,s = time_a()
local early = time_b < time_a
local datetime = bacnet.data.date_time(
    bacnet.data.daytime("10:41:00"), bacnet.data.date(2013,11,24))
for member, value in pairs(datetime) do
    print(member, value) end
local constr_datetime = datetime:typefunc()
local member_info = constr_datetime()
```

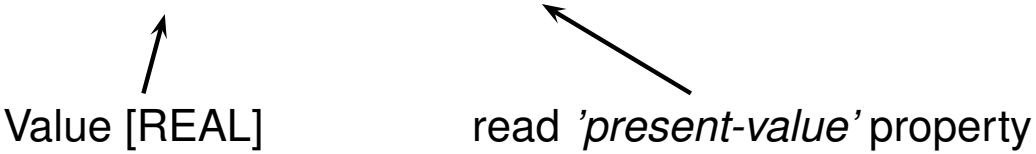
empty call delivers metainfo on members

Example
BACnet Object Access



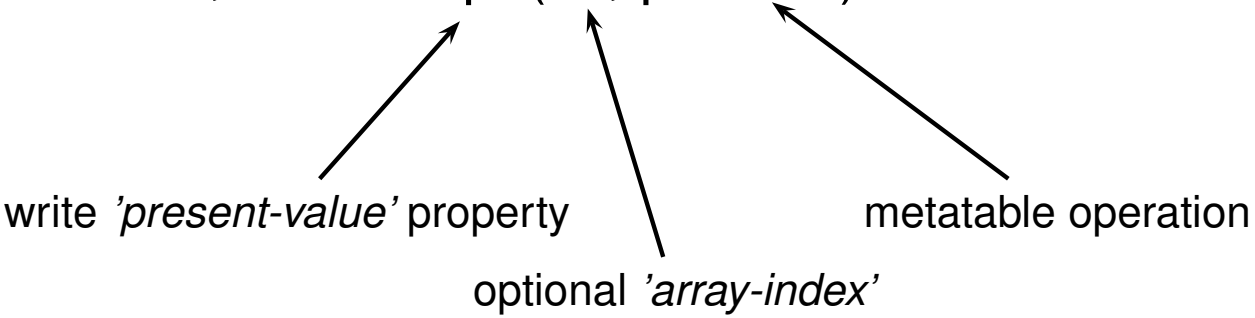
Example
BACnet Object Access

```
local av, err = bacnet.object.get("AnalogValue", 42)
local pv, err = av:pv()
```



Example
BACnet Object Access

```
local av, err = bacnet.object.get("AnalogValue", 42)
local pv, err = av:pv()
local ok, err = av:pv(nil, pv + 12)
```



Example
BACnet Object Access

```
local av, err = bacnet.object.get("AnalogValue", 42)
local pv, err = av:pv()
local ok, err = av:pv(nil, pv + 12)
local ok, err = av:property(   “low-limit”, nil,
                               bacnet.data.real(15.0))
```

property identifier

read/write arbitrary property

datatype constructor

Example
Timers

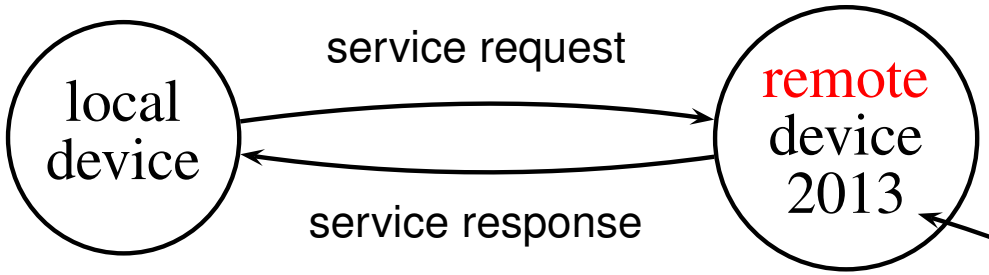
```
Timer = timer.new(function () [...] end)
Timer:start(50)
```

start timer with timeout 50*ms*

timer constructor

callback

Example
BACnet Remote Object Access




```
local ok, err = bacnet.service.write(
    2013,                                    remote device ID
    function (error) [...] end,              response callback (async)
    bacnet.data.objid("AnalogValue", 42),    object ID
    "present-value",                         property ID
    nil,                                     array index
    bacnet.data.real(42))                    write data
```

Application Examples

Display Unit (HMI)

- browse objects
- modify objects
- alarm notifications

Application Examples

Display Unit (HMI)

- browse objects
- modify objects
- alarm notifications

Webserver

- LuaSocket
- e.g. Xavante
- browse objects
- modify objects
- show alarms

Application Examples

Display Unit (HMI)

- browse objects
- modify objects
- alarm notifications

Control Application

- control algorithm in Lua
- based on BACnet objects
- optionally event based

Webserver

- LuaSocket
- e.g. Xavante
- browse objects
- modify objects
- show alarms

SE-ELEKTRONIC GMBH

Application Examples

Display Unit (HMI)

- browse objects
- modify objects
- alarm notifications

Control Application

- control algorithm in Lua
- based on BACnet objects
- optionally event based

Webserver

- LuaSocket
- e.g. Xavante
- browse objects
- modify objects
- show alarms

One-Time-Tasks

- initial startup operations
- diagnosis
- maintenance tasks

SE-ELEKTRONIC GMBH

Summary
Benefits

- Easy access to BACnet objects and properties.
- Versatile high-level API for customized BACnet applications.
- Possibility for rapid development, rapid deployment and easy in-target debugging of applications.
- No Firmware extension in C needed for OEM applications.

Summary
Benefits

- Easy access to BACnet objects and properties.
- Versatile high-level API for customized BACnet applications.
- Possibility for rapid development, rapid deployment and easy in-target debugging of applications.
- No Firmware extension in C needed for OEM applications.

QUESTIONS?