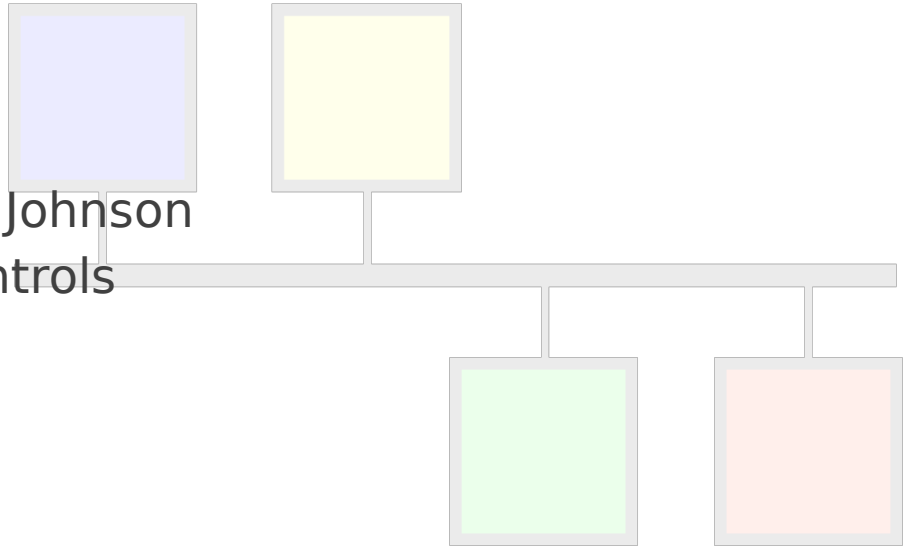
Argonne NATIONAL LABORATORY

Longer string communication with EPICS R3.14.11 IOCs

Andrew Johnson
AES/Controls



Outline

- The DBF_STRING problem
  - Solutions considered
  - Field Modifiers
  - Effects of Field Modifier
- Current support
  - Shell scripts (Base catools)
  - MEDM
  - Perl (cap5)
  - IOC commands
- How to add support
- Future ideas for EPICS Base
- Questions and Demo

The DBF_STRING problem

epicsTypes.h contains

```
#define MAX_STRING_SIZE 40
```

Controls the maximum length of an EPICS string

Prevents using longer strings in many places

The CA wire protocol is the main problem

Software outside of EPICS Base might not use that macro

3.14.x IOC record names can be up to 60 characters long

Can't change INP/OUT links to point to PV names longer than 40 characters over CA

Link flags like NPP and NMS shorten this limit even more

Record types can define string fields of any fixed length up to SHRT_MAX (32767) characters

Solutions considered

Make MAX_STRING_SIZE bigger

Breaks CA communication between versions

How big is “bigger”?

Have to consider the memory limitations of small embedded IOCs

Would have unknown effects on existing CA client programs

Add a new variable-length string type to CA

Not compatible with existing CA clients or IOCs

Would need significant changes to existing client programs

Major difficulties with older clients, e.g. MEDM

Can we use character arrays to pass long strings through CA?

MEDM and EDM can already use char arrays as strings

No significant length limitations

Client has to explicitly ask IOC for new behavior

Field Modifiers

Base R3.14.11 IOCs support a Field Modifier on string field names

Appending '**$**' to the name of a string or link field in a PV name changes the reported native type from DBR_STRING to array of DBR_CHAR

Only works when naming string or link fields

When omitting the field name .VAL, a dot is required

Use **myStringPV.$** or **myStringPV.VAL$**

myStringPV$ is a different record

The number of elements of the array gives the string's maximum length

If a CA clients fetches fewer elements, it must add a terminating zero byte to the array for safety

Examples:

myCalcPV.INPA$

myRecordAlias.NAME$

myStrCalcPV.SVAL$

Effects of Field Modifier

```
tux% cainfo record.NAME
record.NAME
    State:              connected
    Host:               tux.aps.anl.gov:5064
    Access:             read, no write
    Native data type:   DBF_STRING
    Request type:       DBR_STRING
    Element count:      1

tux% cainfo record.NAME$
record.NAME$
    State:              connected
    Host:               tux.aps.anl.gov:5064
    Access:             read, no write
    Native data type:   DBF_CHAR
    Request type:       DBR_CHAR
    Element count:      61
```

Current support

The only standard record types that provide long string fields are those that support arrays, mainly waveform, aSub and genSub

Most CA clients don’t yet support the use of char arrays as strings

Outside of MEDM and EDM there was little need for this before now

Clients provided with Base R3.14.11 support this

The following slides describe the clients that I know do

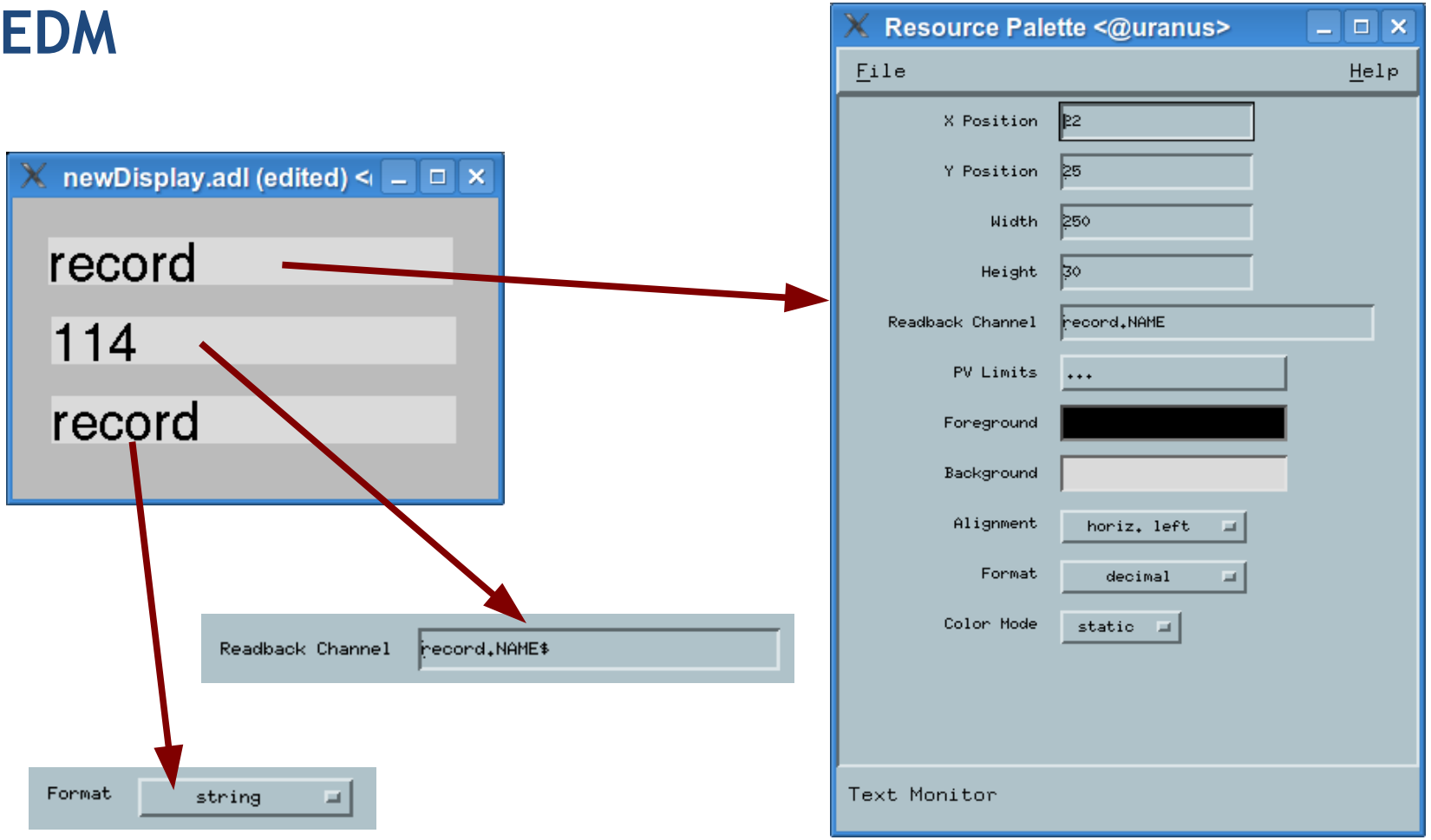
Shell scripts (Base catools)

The caget, caput and camonitor programs from Base R3.14.11 have a switch -S which means ‘print char arrays as a long string’

```
tux% caget record.NAME record.NAME$
record.NAME             record
record.NAME$ 61 114 101 99 111 114 100 0 0 0 0 0 0 0 0 0 0 …

tux% caget -S record.NAME record.NAME$
record.NAME             record
record.NAME$ record
```

MEDM



EDM has very similar support for long strings

Perl (cap5)

The Perl support for long strings in Base R3.14.11 works like this:

$chan->put(*VALUES*)

If **$chan** is an array of chars and ***VALUES*** is a scalar, the string representation of ***VALUES*** is passed to CA as a char array

If **$chan** is an array of chars and ***VALUES*** is a list, the integer representations of the list items are assembled into a char array and passed to CA

$chan->get

If **$chan** is an array of chars, it is fetched and presented as a string

if **$chan** is a single char, it is presented as an integer

For other field types, **get** only fetches the first element of an array anyway

$chan->get_callback(*SUBR*, *TYPE*, *COUNT*)

$chan->create_subscription(*MASK*, *SUB*, *TYPE*, *COUNT*)

TYPE and ***COUNT*** are optional, their default values depend on **$chan**

if **$chan** is a single char, ***TYPE*** defaults to **"DBR_LONG"**

If **$chan** is an array of chars, ***TYPE*** defaults to **"DBR_CHAR"**

When the data is returned, DBF_CHAR values are always presented as a string

To fetch a DBF_CHAR array as integers, pass **"DBF_LONG"** for ***TYPE***

IOC commands

Long string fields can be initialized normally with dbLoadRecords

Both dbpf and dbgf support accessing char arrays as strings

```
epics> dbpf asub.DESC "This string is longer than forty characters"
DBR_STRING:          "This string is longer than forty charac"
epics> dbpf asub.B "This string is longer than forty characters"
DBR_CHAR[44]:        "This string is longer than forty characters"
```

Unfortunately neither dbpf nor dbgf display the complete string when used with a '$' field modifier, due to a bug

```
epics> dbgf asub.DESC$
DBR_CHAR[19]:        "This string is long"
epics> dbgf asub.DESC
DBR_STRING:          "This string is longer than forty charac"
```

The R3.14.11 Known Problems page has a patch to fix this

How to add support

Guidelines and suggestions for developers to add long string support to their own code

Do not try to parse the PV name to detect when to use a char array

You won't catch waveform or array subroutine record fields

CA Clients do not have to be built with R3.14.11, any version will work

The IOC does have to use R3.14.11 to support field modifiers though

If a different API is used to access scalar and array fields, you may be able to handle scalar I/O to a char array as a long string access

If the API has a data type argument, you might be able to add a new pseudo-type for long string processing

Future ideas for EPICS Base

More field modifiers

Array offsets

```
waveform.VAL[100]
waveform.VAL[-20]
```

Lists of fields?

```
transform.A,B,C,D
```

Structured fields?

```
record.TIME.secPastEpoch,TIME.nsec
```

JSON Modifiers

```
SRCurrent.VAL{"rate":0.5}
```

Support structured data using JSON encoding

```
{"IDgap":{"us":25,"ds":27.5}}
```

Questions

Questions for Tim Mooney:

- Any comments for synApps users?

Questions for the Audience:

- What other CA clients or libraries need support for long strings?