Real-time CORBA for soft-radio

Part 2c

Optimization considerations for soft-radio designer
Anatomy of the GIOP protocol
High performance IDL
CDR streams and IDL

(Mark Glenn, S. Aslam-Mir)

RT-CORBA for soft-radio

Optimisation considerations for soft-radio designer

- **Use of CORBA Anys**
- **Use of Anys vs Unions**
- **Use of CORBA structs – typecode concerns**
- **Efficient layout of CORBA structs**
- **Naming methods in CORBA interfaces (GIOP)**
- **Naming objects in the server (OIDs)**
- **Use of Custom Transports – use ETF**

RT-CORBA for soft-radio

Anatomy of GIOP

- **CORBA IDL interface**

```
interface I
{
    void f( );
};
```

RTORB for data transport

- **CORBA Request - wire**

```
0x47, 0x49, 0x4F, 0x50,                 // 'G' 'I' 'O' 'P'
0x01, 0x00,                             // GIOP version '1' '0'
0x01,                                   // flgs
0x00,                                   // msg type  (one of 8 )
0x28, 0x00, 0x00, 0x00,                 // msg size

/* RequestHeader */
0x00, 0x00, 0x00, 0x00,                 // svc context
0x00, 0x00, 0x00, 0x00,                 // request id
0x01,                                   // response_expected
0x00, 0x00, 0x00,                       // padding
0x09, 0x00, 0x00, 0x00,                 // object key size
0x00, 0x00, 0x00, 0x00, 0x00,           // object key body
        0x00, 0x00, 0x00, 0x00,
0x04, 0x31, 0x00,                       // padding
0x02, 0x00, 0x00, 0x00,                 // operation name size
0x66, 0x00,                             // operation name ('f' '0')
0x00, 0x00,                             // padding
0x00, 0x00, 0x00, 0x00                  // principal
```

RT-CORBA for soft-radio

► **High Performance IDL - Super optimized marshalling – intelligently**

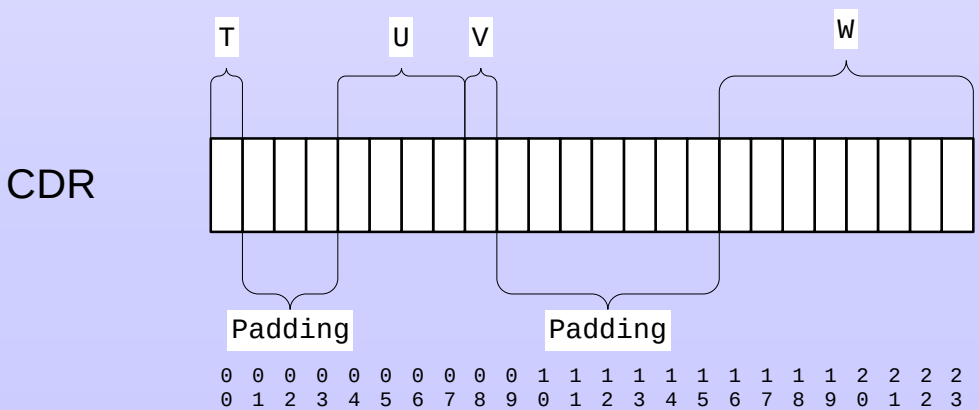
e.g. IDL structure that yields inefficient CDR layout.

- **Padding required to align data per CDR rules**

```
struct S {
  boolean  T;
  long     U;
  char     V;
  double   W;
};

-- same CDR layout as --

void Func(in boolean T, in long U, in char V, in double W);
```



RT-CORBA for soft-radio

- **CDR streams**

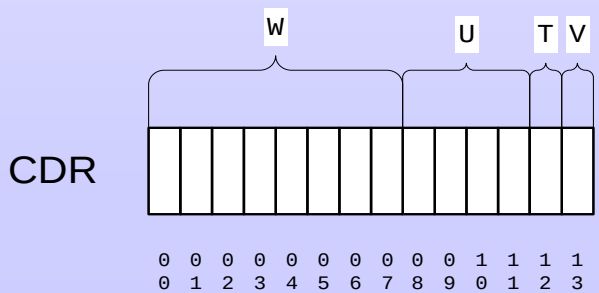
  IDL structure organization that yields optimal CDR layout.

  - **Structure elements rearranged to eliminate need for padding.**

```
struct S {
  double   W;
  long     U;
  boolean  T;
  char     V;
};

-- same CDR layout as --

void Func(in double W, in long U, in boolean T, in char V);
```

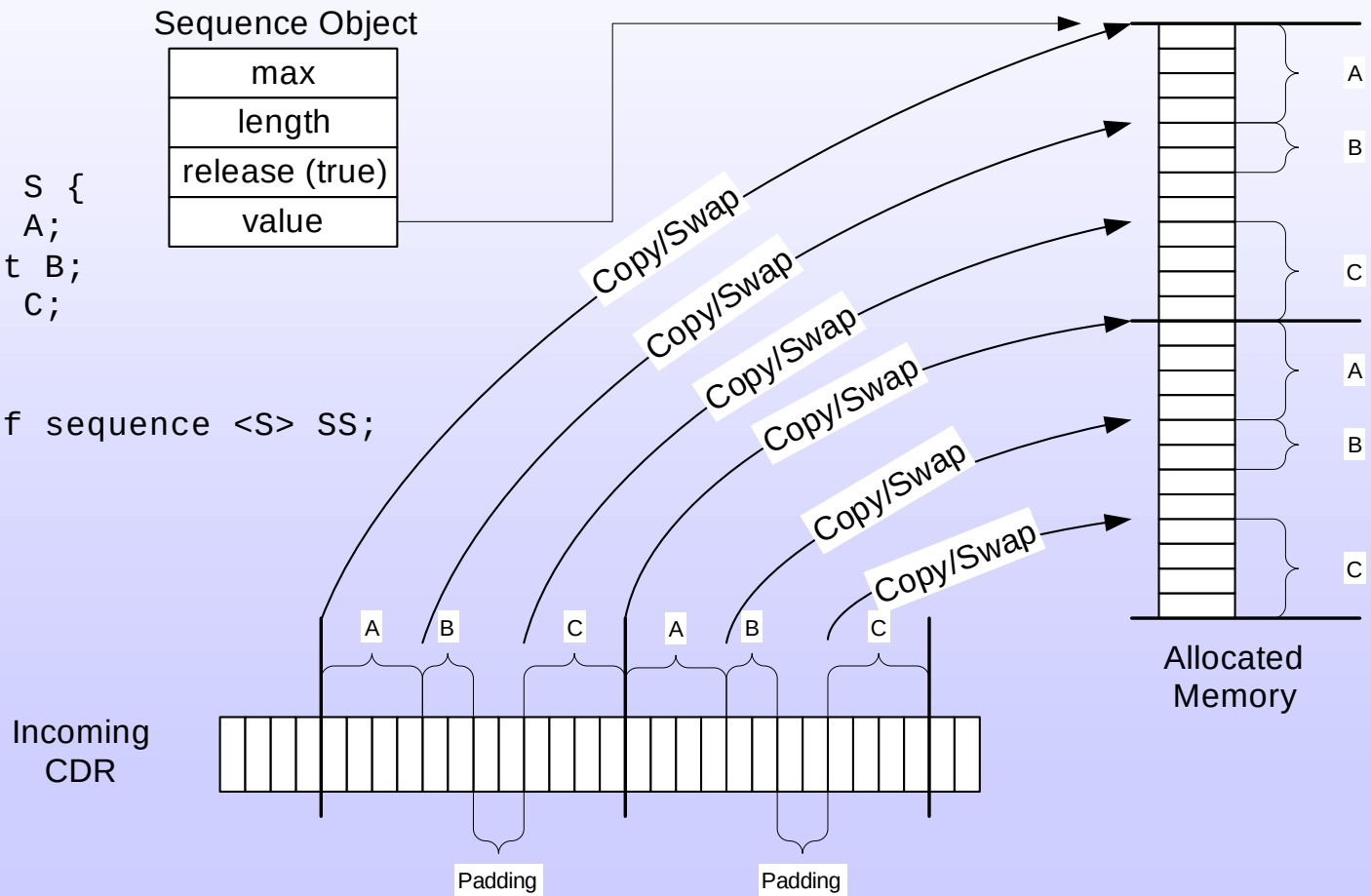


RT-CORBA for soft-radio

- Non optimized allocation and copying of a sequence of structs
  - Memory allocation and copying required normally – inefficient use of ORB

```
struct S {
  long A;
  short B;
  long C;
};

typedef sequence <S> SS;
```

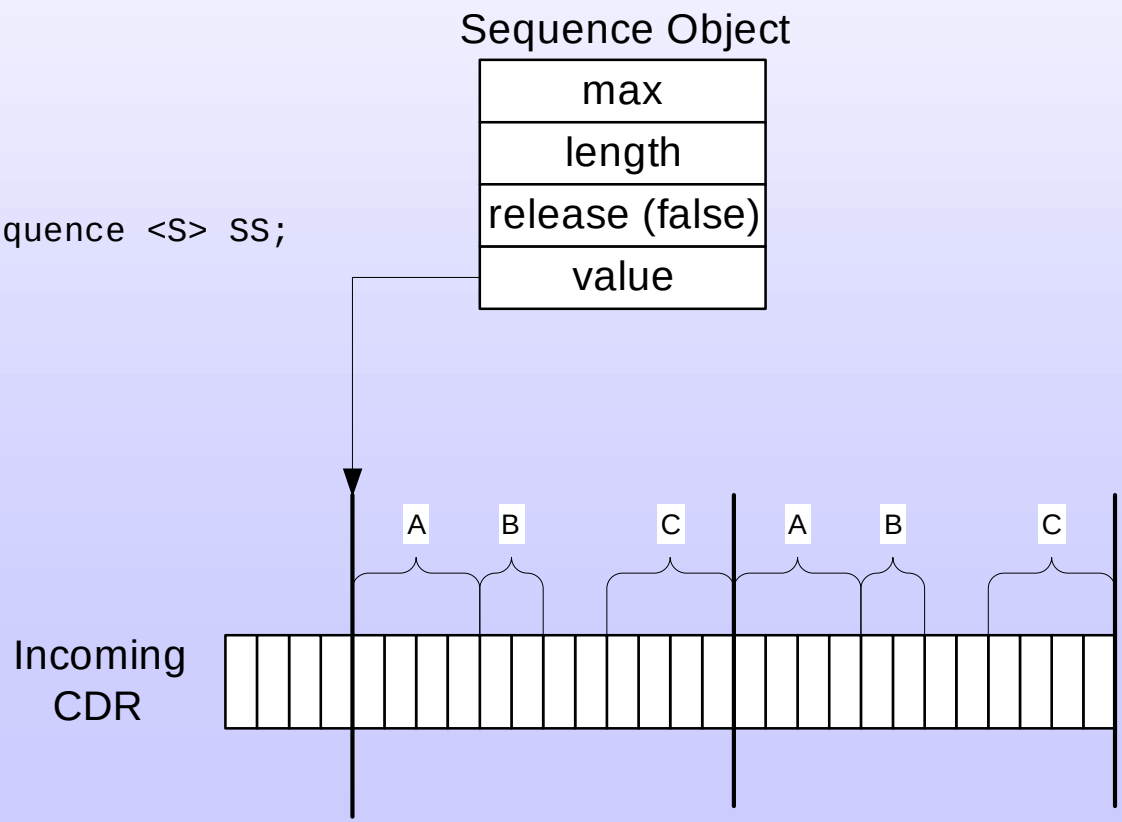


RT-CORBA for soft-radio

- Optimized reference to the CDR buffer of a sequence of structs
  - To a avoid a costly allocation and copy, the rules for alignment, padding and endianness of the compiler and processor must all match the corresponding attributes for the OMG standard CDR rules.
  - Fortunately, these rules often do match and the optimization can be applied.

```
struct S {
  long A;
  short B;
  long C;
};

typedef sequence <S> SS;
```



RT-CORBA in soft-radio (SPS)

- **CORBA in the real-time payload chain**
  - **Operation name 'TFM' is optimal for the wire – three characters plus the null termination character fit nicely into a single DWord of the CDR stream.**

```
IDL: void oneway TFM(in DataSeq Data, in ControlSeq Control);
```

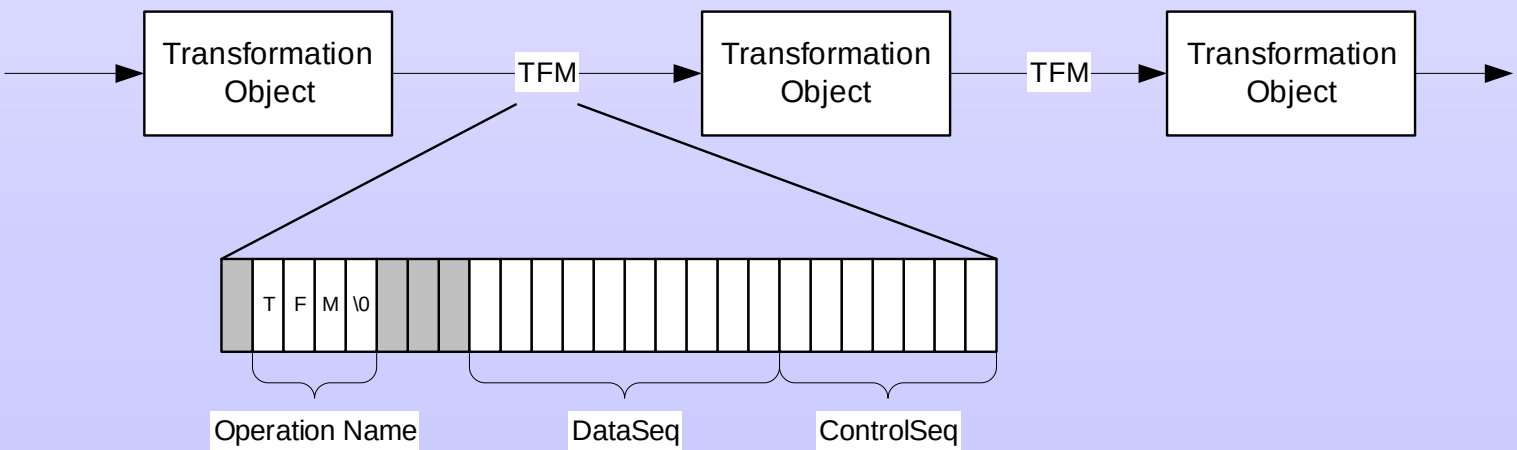
```
Implementation Pseudo-code:

void TFM(DataSeq Data, ControlSeq Control)
{
  // Perform transformation on Data either in-place or new copy

  TransformedData = Transform(Data);

  pNext -> TFM(TransformedData, Control)
}
```

signal processing chain….



RT-CORBA in soft-radio (SPS)

- **CORBA in the real-time payload chain**
  - **Operation name 'TFM' is optimal for the wire – three characters plus the null termination character fit nicely into a single DWord of the CDR stream.**

```
IDL: void oneway TFM(in DataSeq Data, in ControlSeq Control);
```

```
Implementation Pseudo-code:

void TFM(DataSeq Data, ControlSeq Control)
{
  // Perform transformation on Data either in-place or new copy

  TransformedData = Transform(Data);

  pNext -> TFM(TransformedData, Control)
}
```