MCAP

Modular Compression Architecture Protocol

Support for ISO JPEG 2000

Still Image Compression Standard

The Aerospace Corporation (MCAP Design)

Trident Data Systems (Software Development)

Original Objectives for Modular Compression Development

PRE-JPEG 2000

Dynamic Technologies
(Motivation for Modular Compression)

New Compression Technology

new encoders

new quantizers

specialized rate control

improved image quality

error resiliant coders

new transforms

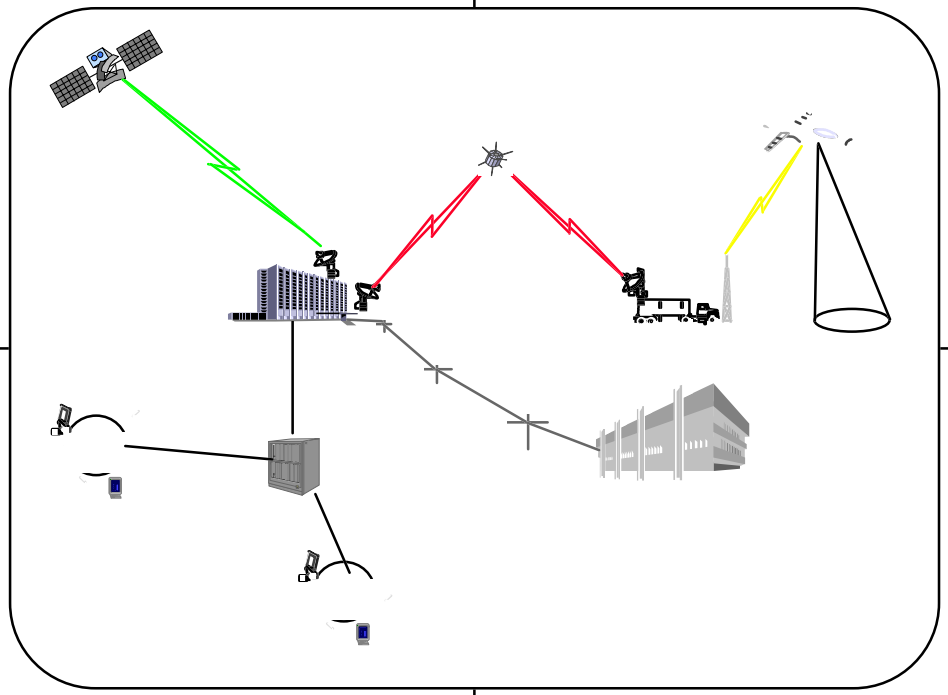
New Exploitation Products

SAR MASINT

fusion products

new modes to existing products

hyperspectral MASINT



New Data Sources

upgrades to existing systems

complex SAR data

commercial sensors

multispectral data

hyperspectral data

New Users

fielding new hardware

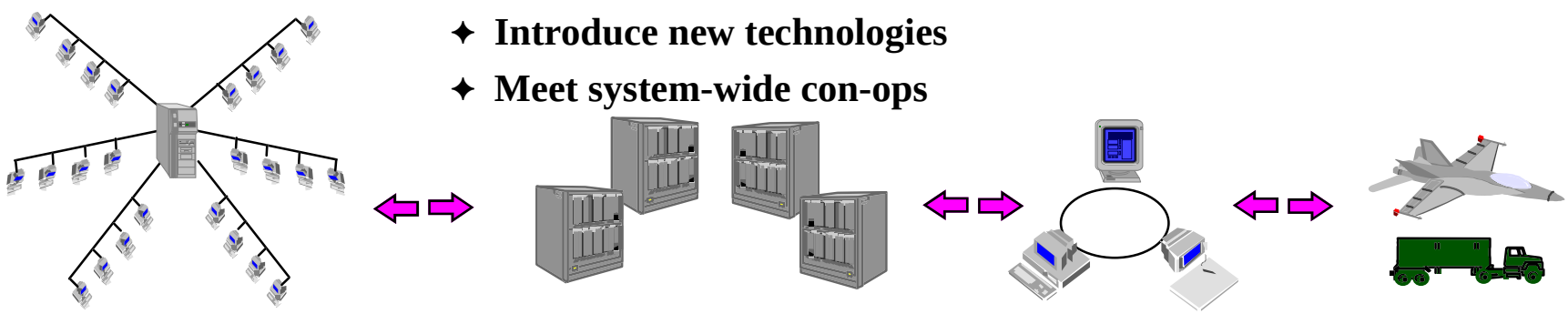
new programs

changes to existing programs

broadened user base

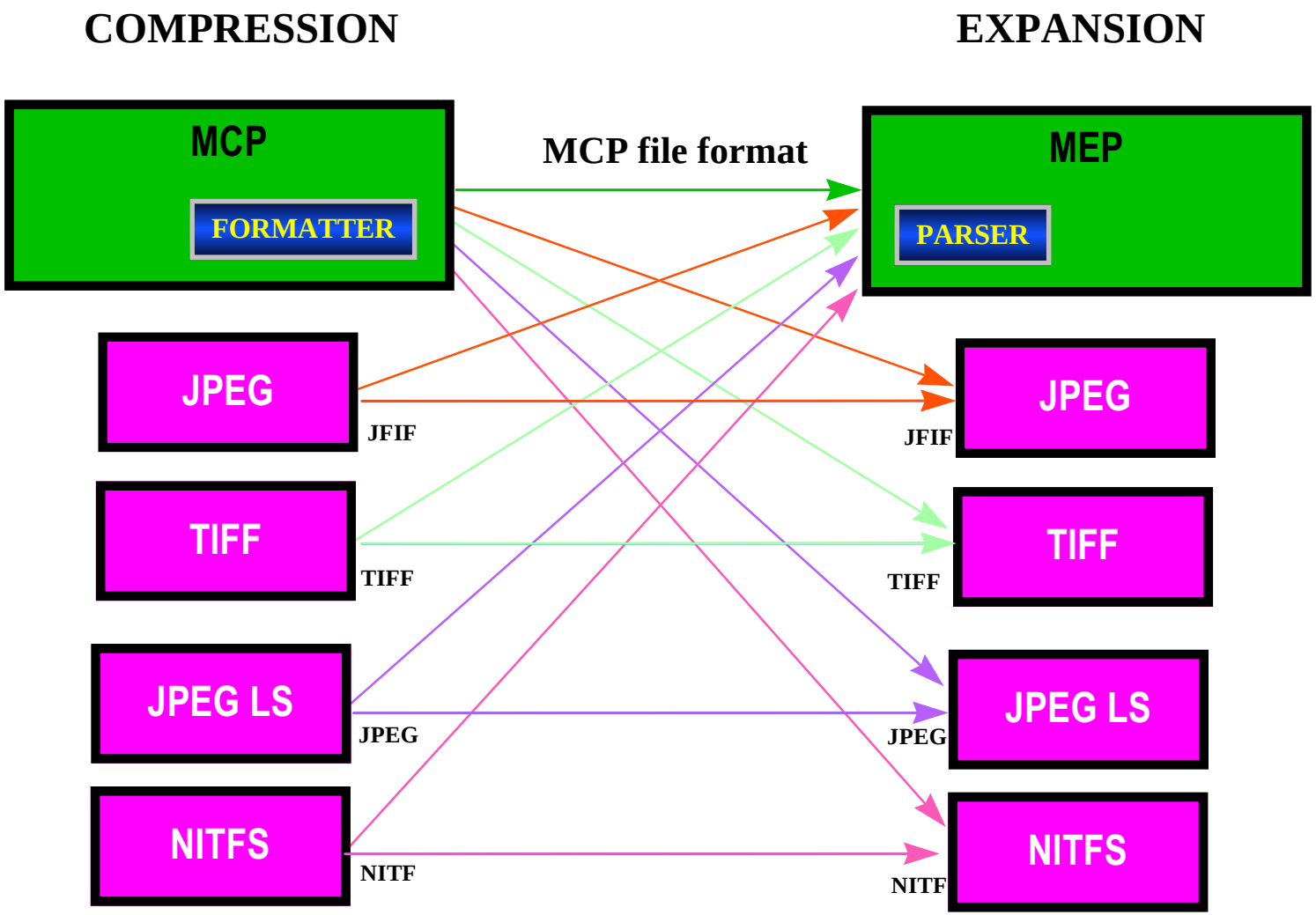
Original Motivation for Modular Architecture

- **From Sensor to Shooter - Dealing with Interoperability & Futureproofing**
  - **Image dissemination architecture diversity**
    - **System-wide compatibility**
    - **Hardware and Software requirements**
    - **User requirements**
    - **Implementation and upgrade timelines**
  - **Future Friendly solution**
    - **New hardware architectures**
    - **Introduce new technologies**
    - **Meet system-wide con-ops**



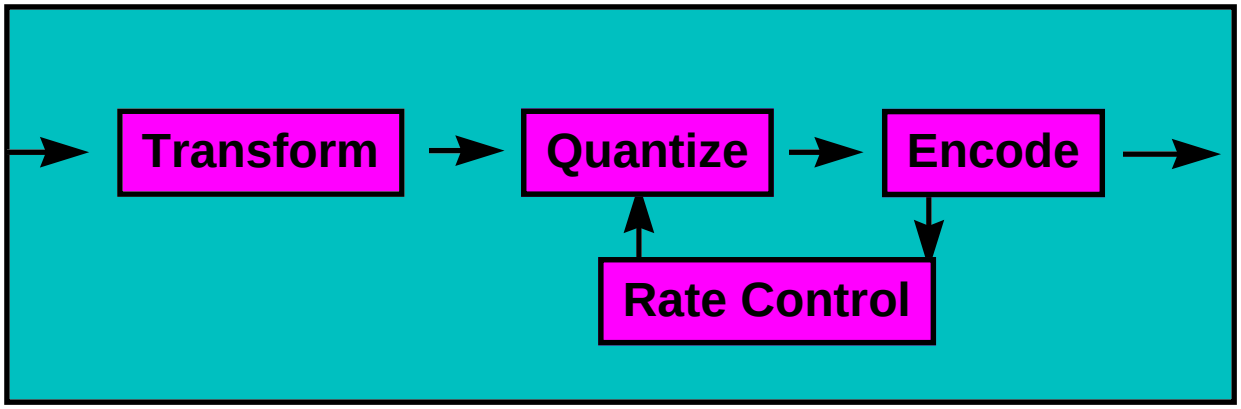
Client/Server (Multi-CPU, Heterogeneous) | Distribution (MPP, High Capacity) | Network (Single CPU) | Tactical (Low Power, Low BW)

Interoperability and MCAP



Two-component Architecture

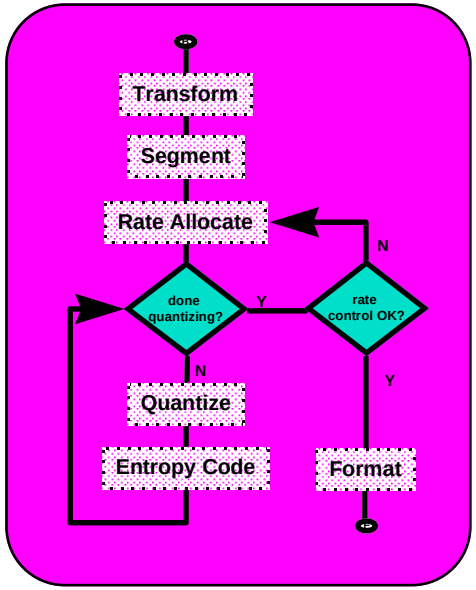
- **Fixed Framework**
  - **Controls I/O, Algorithm flow & structure**
  - **Ties compression to hardware architecture**



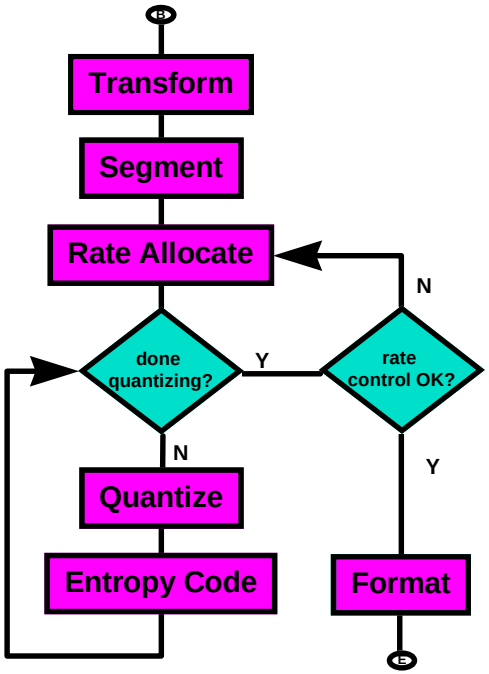
  - **Programmable Modules**
    - **Provides functional processing**
    - **Provides flexible, changeable compression function**

Two Component System Requirements

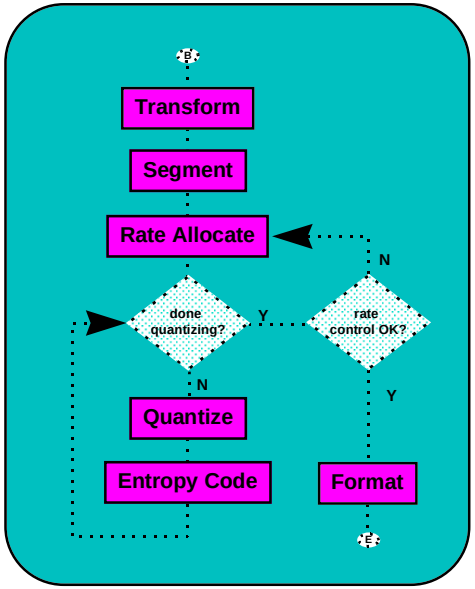
- **Fixed Framework**
  - Abstract Datatypes
  - Hardware configuration
  - Common algorithm flow
  - Distributed processing



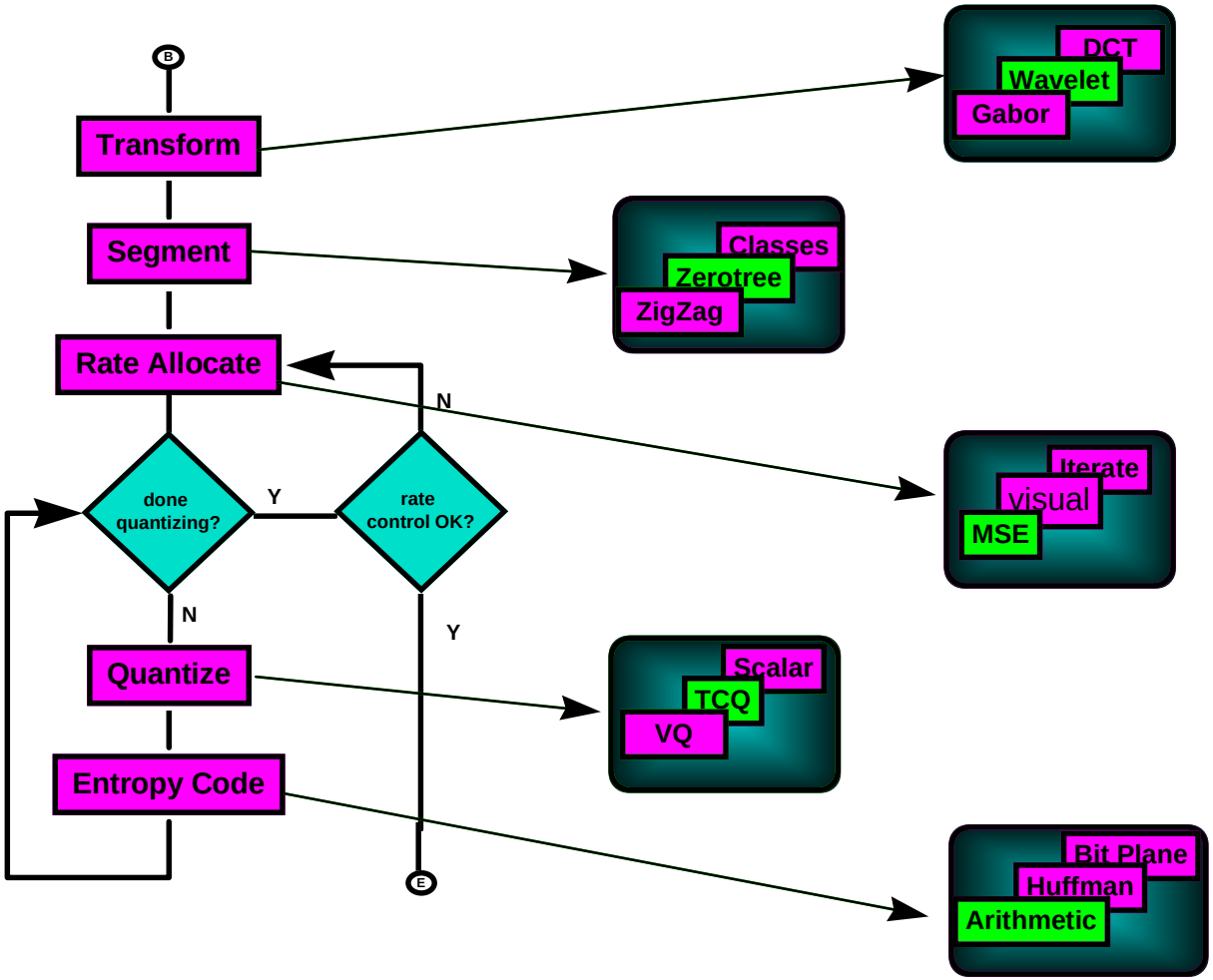
Algorithm



- **Modules**
  - Programmable
  - Language independent



Dynamic Algorithm Configuration

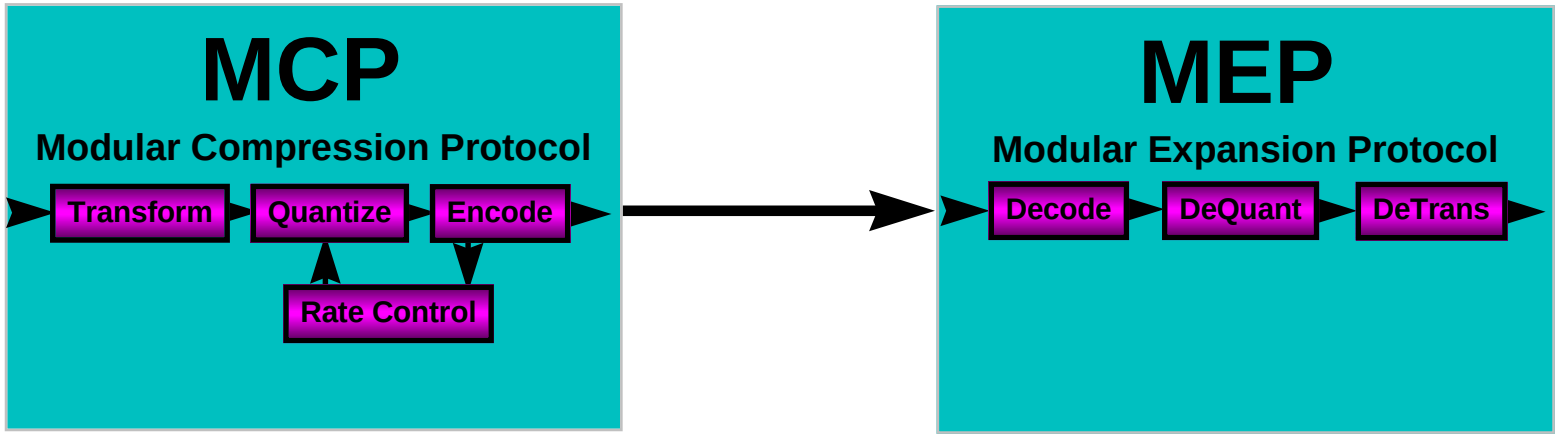


- **Flexibility**
  Select modules to define specific compression function. Multiple functions maintained within single framework.

- **Interoperability**
  Maintain sets of modules to broaden capability. Transmit and receive from multiple sources.

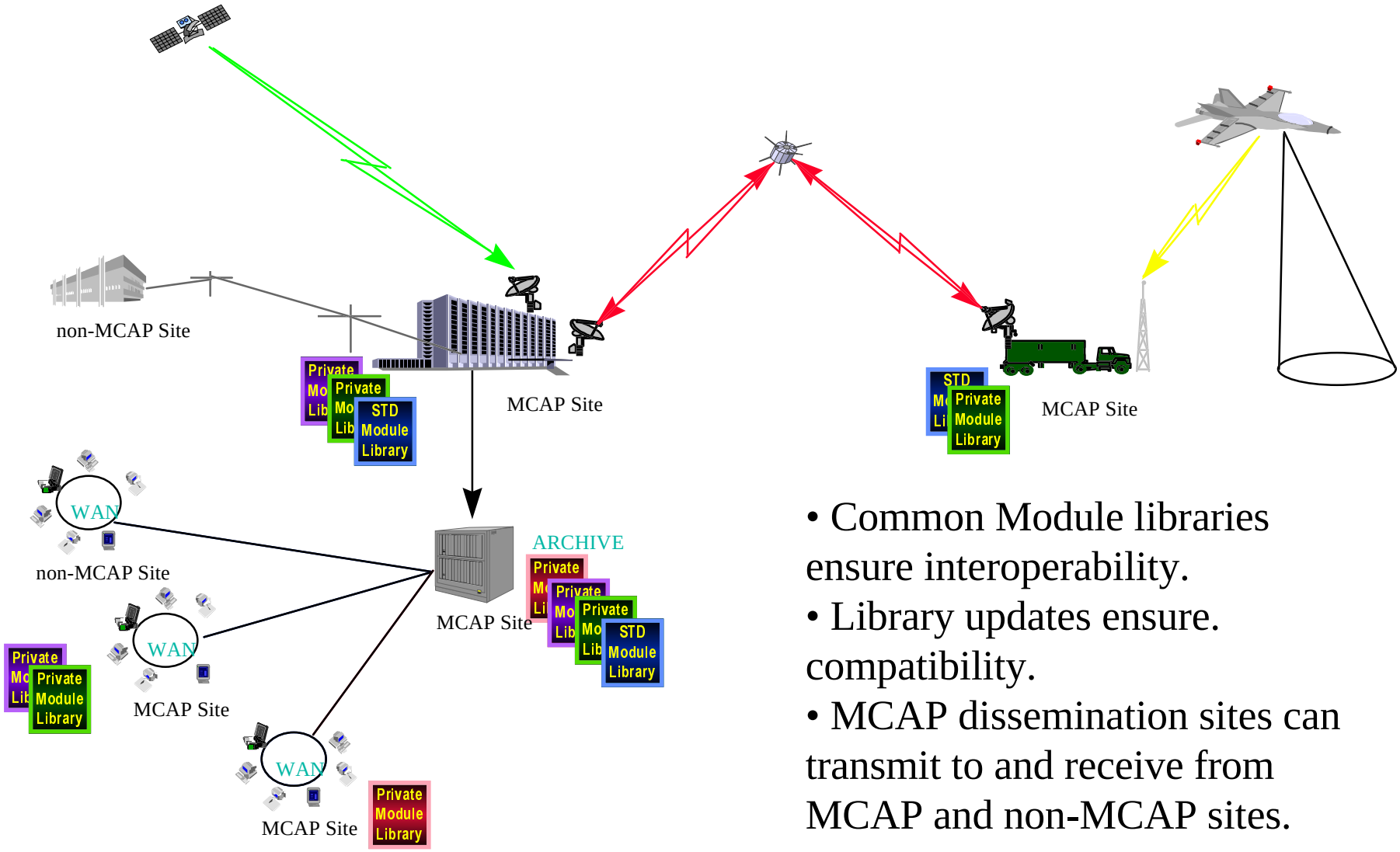
- **Future Proofing**
  Addition of new modules allows update or change in capability without system modification.

MCAP



- **A mechanism to implement and easily change compression algorithms in a modular fashion.**
  - Software framework
  - Programmable modules
  - Run-time configuration
  - Object Oriented language (C++)
  - Multiple data types
  - Supports various hardware
  - Future friendly

Concept of Operations



- Common Module libraries ensure interoperability.
- Library updates ensure. compatibility.
- MCAP dissemination sites can transmit to and receive from MCAP and non-MCAP sites.

STATUS - Pre JPEG 2000

- **MCP and MEP functional**
  - **C++**
- **Multiple algorithms -**
  - **Wavelet/TCQ, JPEG, Gabor/TCQ, CREW (similar), CCITT T.4 FAX**
- **CPU usage -**
  - **Multi-threaded execution**
- **Operating system -**
  - **Solaris, Windows 95, NT**

Architectures for Compression Standards

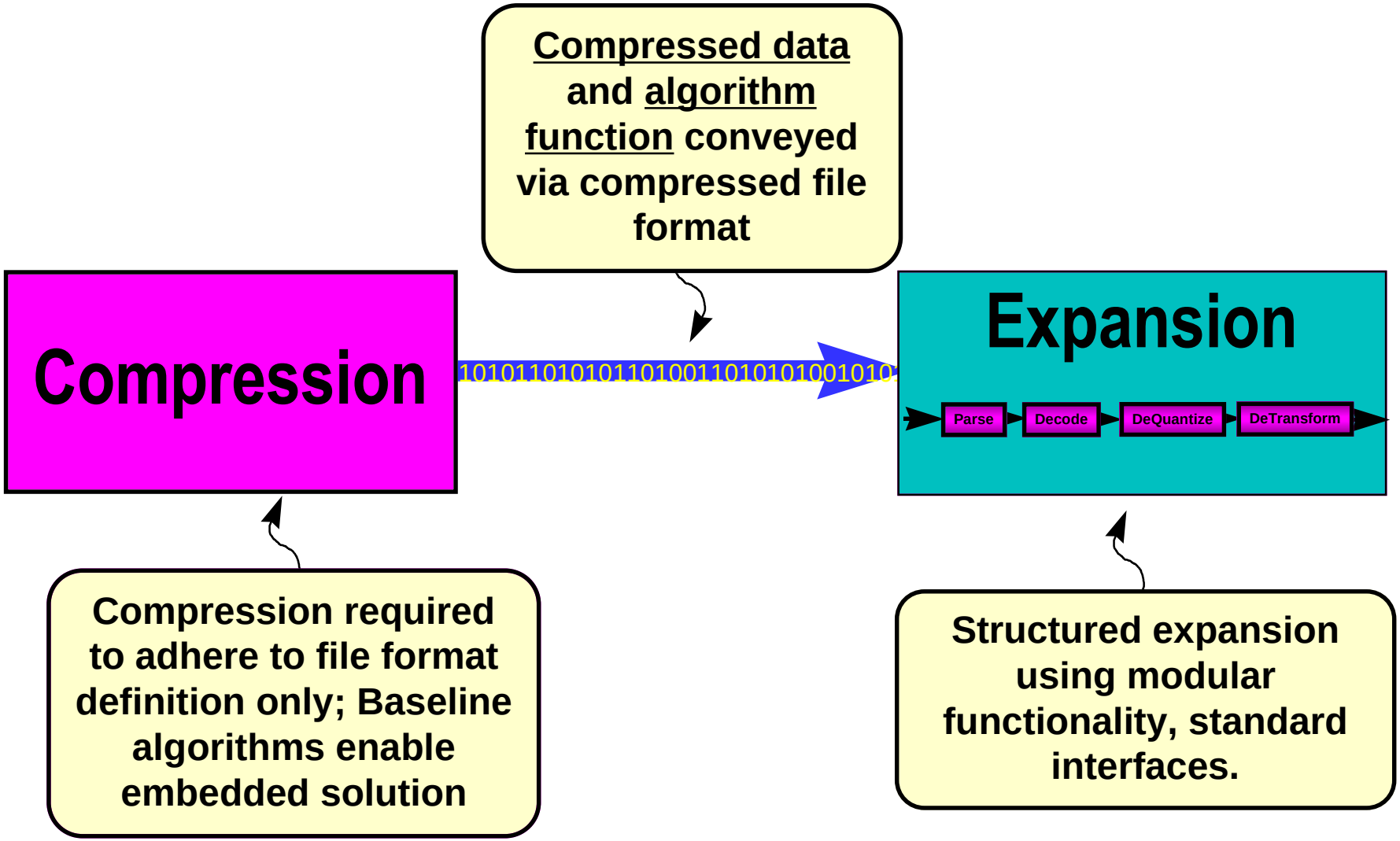
MCAP support to JPEG 2000

JPEG 2000 Requirements

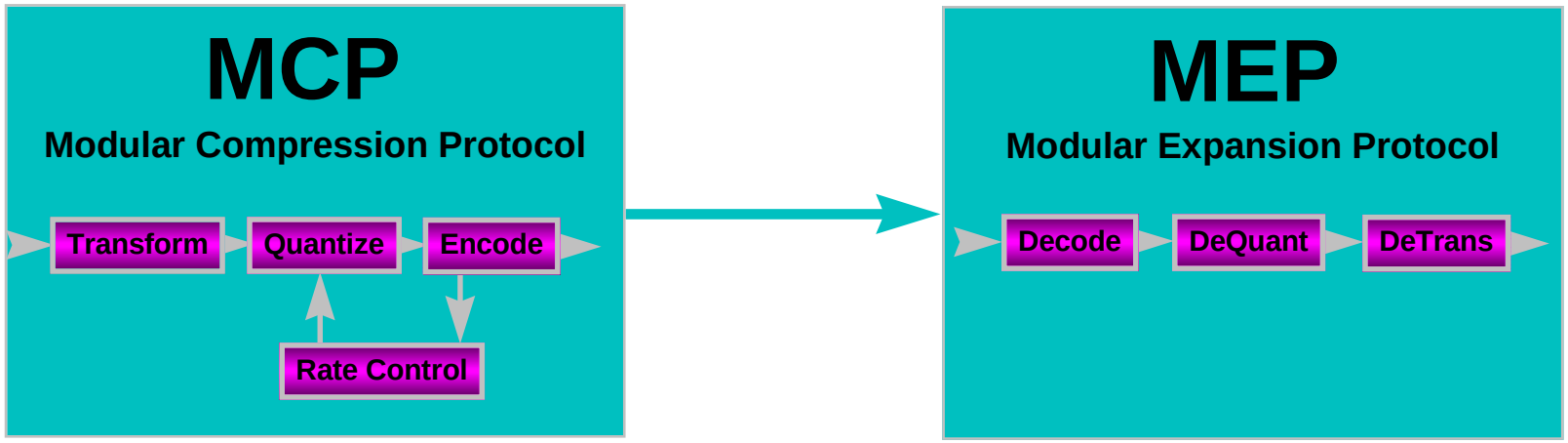
Superior low bit-rate performance

- Continuous-tone and bi-level compression
- Lossless and lossy compression
- Progressive transmission by pixel accuracy and resolution
- Fixed-rate, fixed-size, limited workspace memory
- Random codestream access and processing
- Robustness to bit-errors
- Open architecture
- Sequential build-up capability (real time coding)
- Backwards compatibility with JPEG
- Content-based description:
- Protective image security
- Interface with MPEG-4
- Side channel spatial information (transparency)

Modified MCAP for JPEG 2000



MCAP Support to JPEG 2000



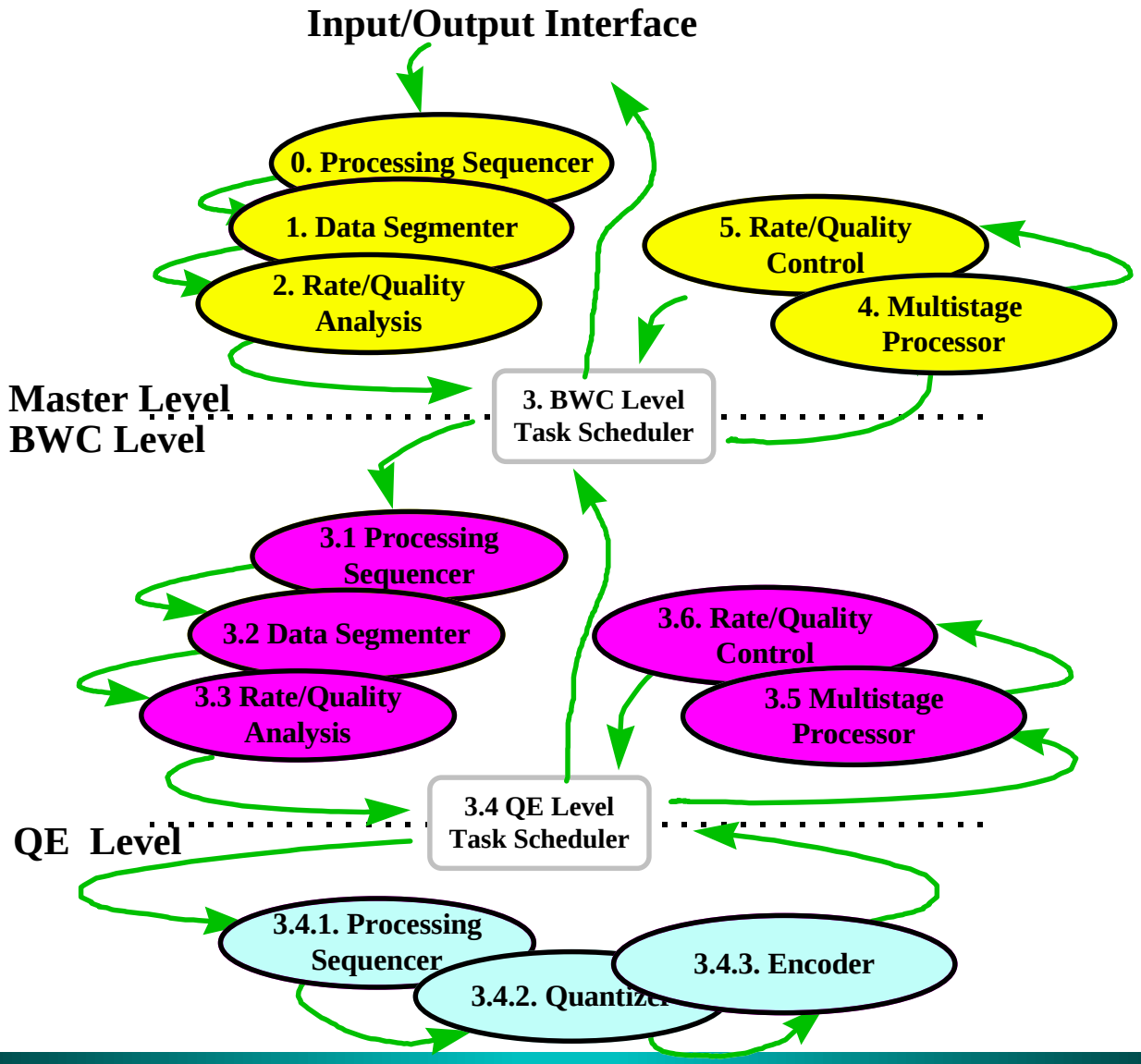
- pre-JPEG 2000
  - multiprocessor
  - old version options
  - integrated framework and modules
- modified to support convergence
  - modular file format support

- JPEG 2000 modified
  - modular file format
  - expansion framework
  - standardized module interfaces
  - options
  - interdependencies
- reflects proposed architecture ideas

JPEG 2000 MCP and MEP Architecture Description

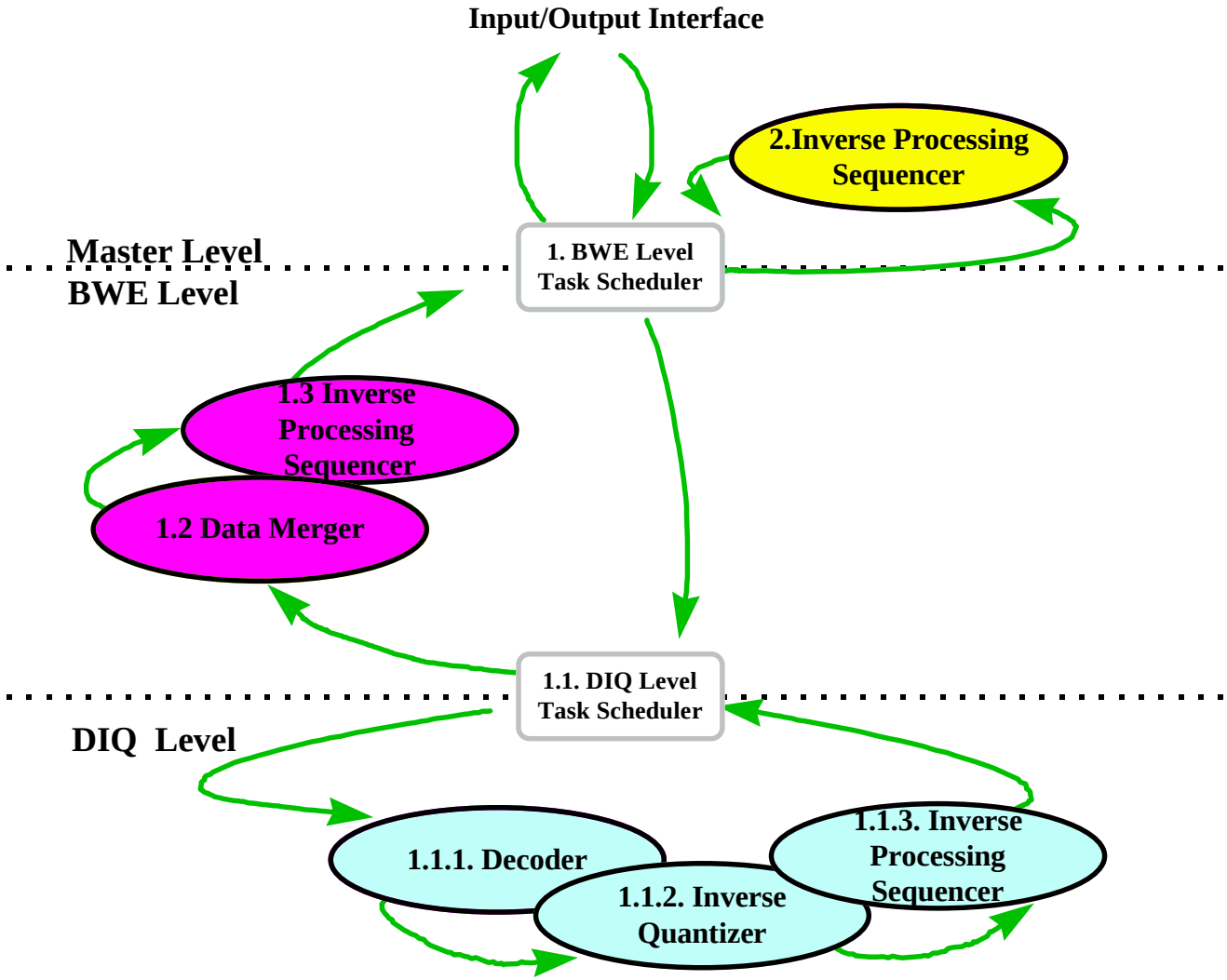
MCP Framework Sequence

(For reference only, not for architecture standard)

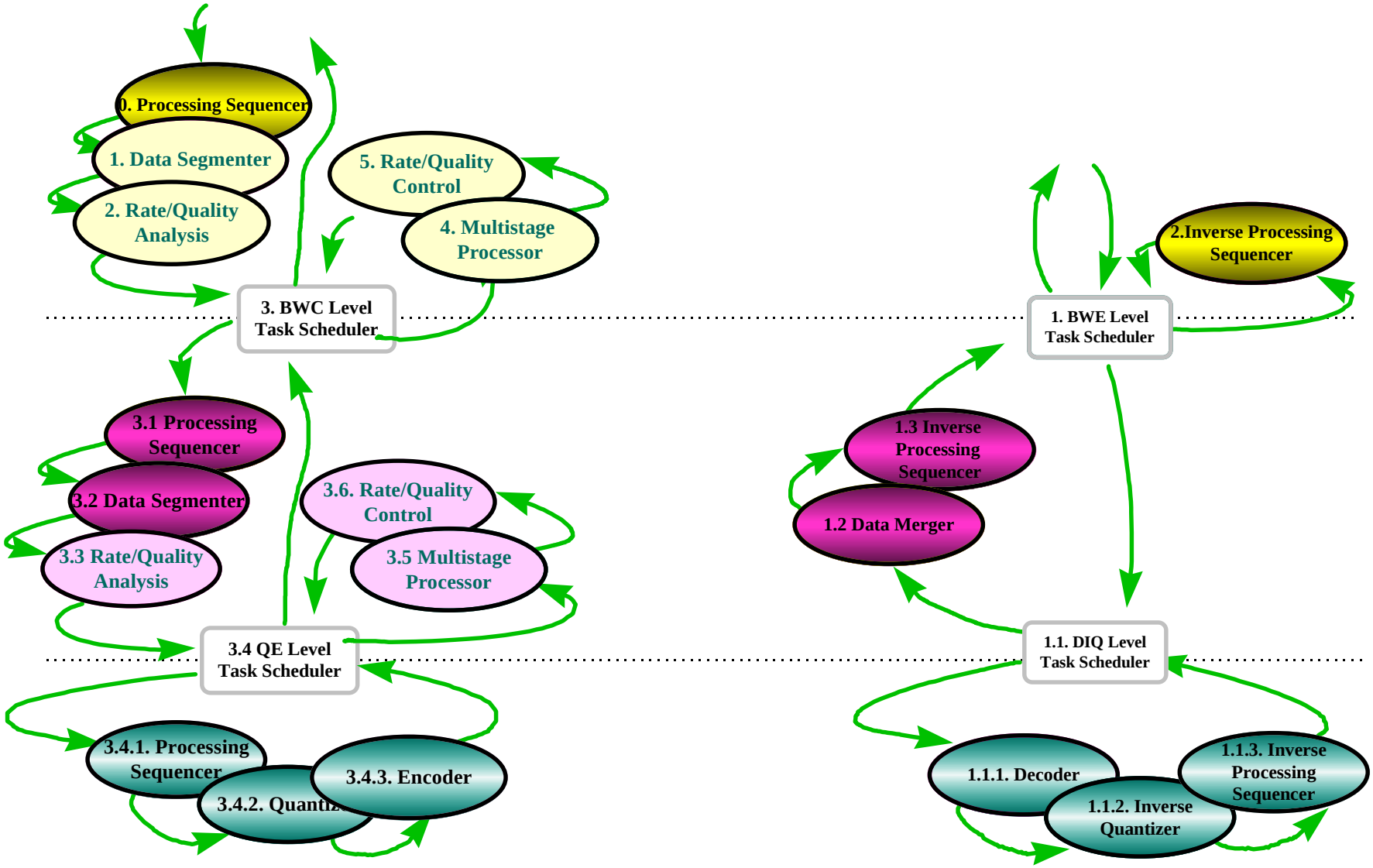


Expansion Framework Sequence

(Standard Framework Definition)



Simplified Expansion Framework



Configuration of Baseline File Format

[Master Header]

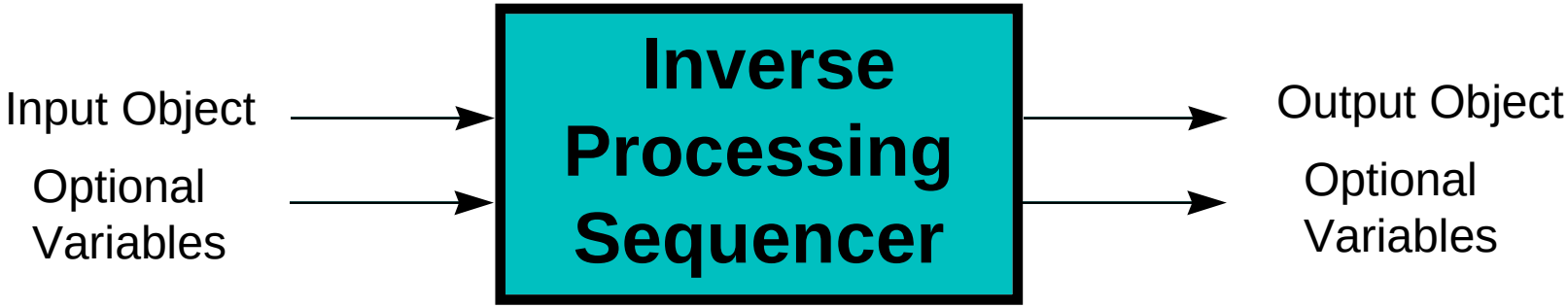
- [BWC 1 Header]
  - [BWC 1 Segment 1 Compressed Data]
  - [BWC 1 Segment 2 Compressed Data]
  - [BWC 1 Segment 3 Compressed Data]
  - [BWC 1 Segment 4 Compressed Data]
  - [BWC 1 Segment 5 Compressed Data]
- [BWC 2 Header]
  - [BWC 2 Segment 1 Compressed Data]
  - [BWC 2 Segment 2 Compressed Data]
  - [BWC 2 Segment 3 Compressed Data]
  - [BWC 2 Segment 4 Compressed Data]
  - [BWC 2 Segment 5 Compressed Data]
- [BWC3 Header]
  - [BWC 3 Segment 1 Compressed Data]
  - [BWC 3 Segment 2 Compressed Data]
  - [BWC 3 Segment 3 Compressed Data]
  - [BWC 3 Segment 4 Compressed Data]
  - [BWC 3 Segment 5 Compressed Data]

Algorithm Profiling

	Baseline1	Baseline2	Baseline3	FLEX1	FLEX2
Inverse Processing Sequencer (Master Level)	MBWC IPS1	MBWC IPS2	MBWC IPS3	MBWC IPS4	MBWC IPS5
Inverse Processing Sequencer (BWE Level)	BWCIPS1	BWC IPS2	BWC IPS2	select baseline algorithm profile	select baseline algorithm profile
Data Merger (BWE Level)	BWC DM1	BWC DM2	BWC DM2		
Decoder* (DIQ Level)	DEC1, DEC2	DEC3, DEC4, DEC1, DEC2	DEC5		
Inverse Quantizer* (DIQ Level)	IQ1, IQ2, IQ3	IQ1, IQ2, IQ3	IQ5		
Inverse Processing Sequencer* (DIQ Level)	IPS1	IPS2	IPS3		

*Multiple modules for different segments

Standardized Module Interface

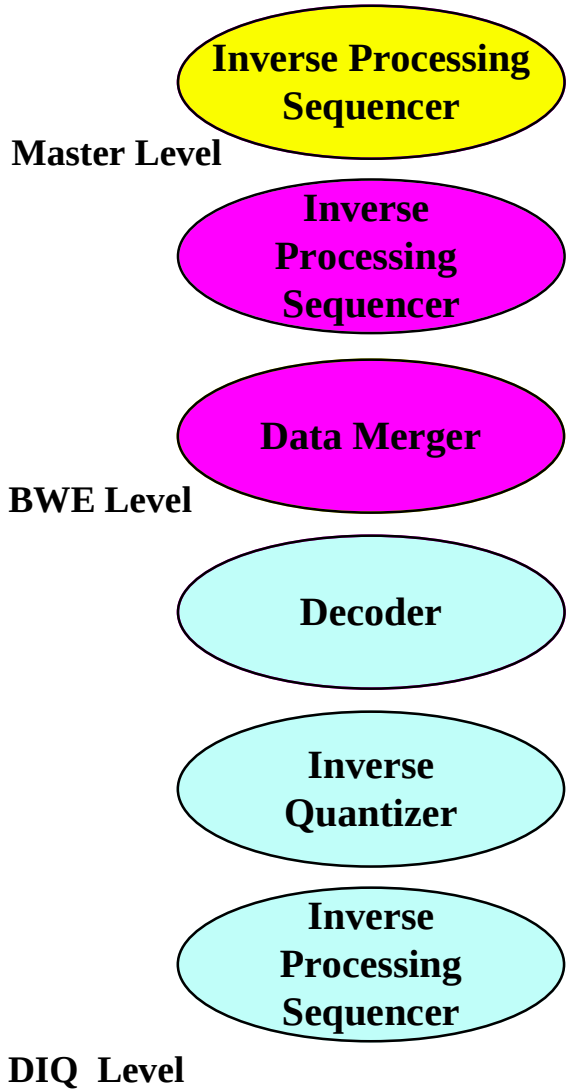



```
struct AnyArray {
  char Dimensions;
  unsigned long Dim1, Dim2, Dim3;
  sequence<any> Representation;
};

 struct BitVector {
   unsigned long Length;
   sequence<octet> Representation;
 };
```

```
interface MBWE_IPS {
  void call(in AnyArray Input,
            out AnyArray Output,
            inout Options);
};
```

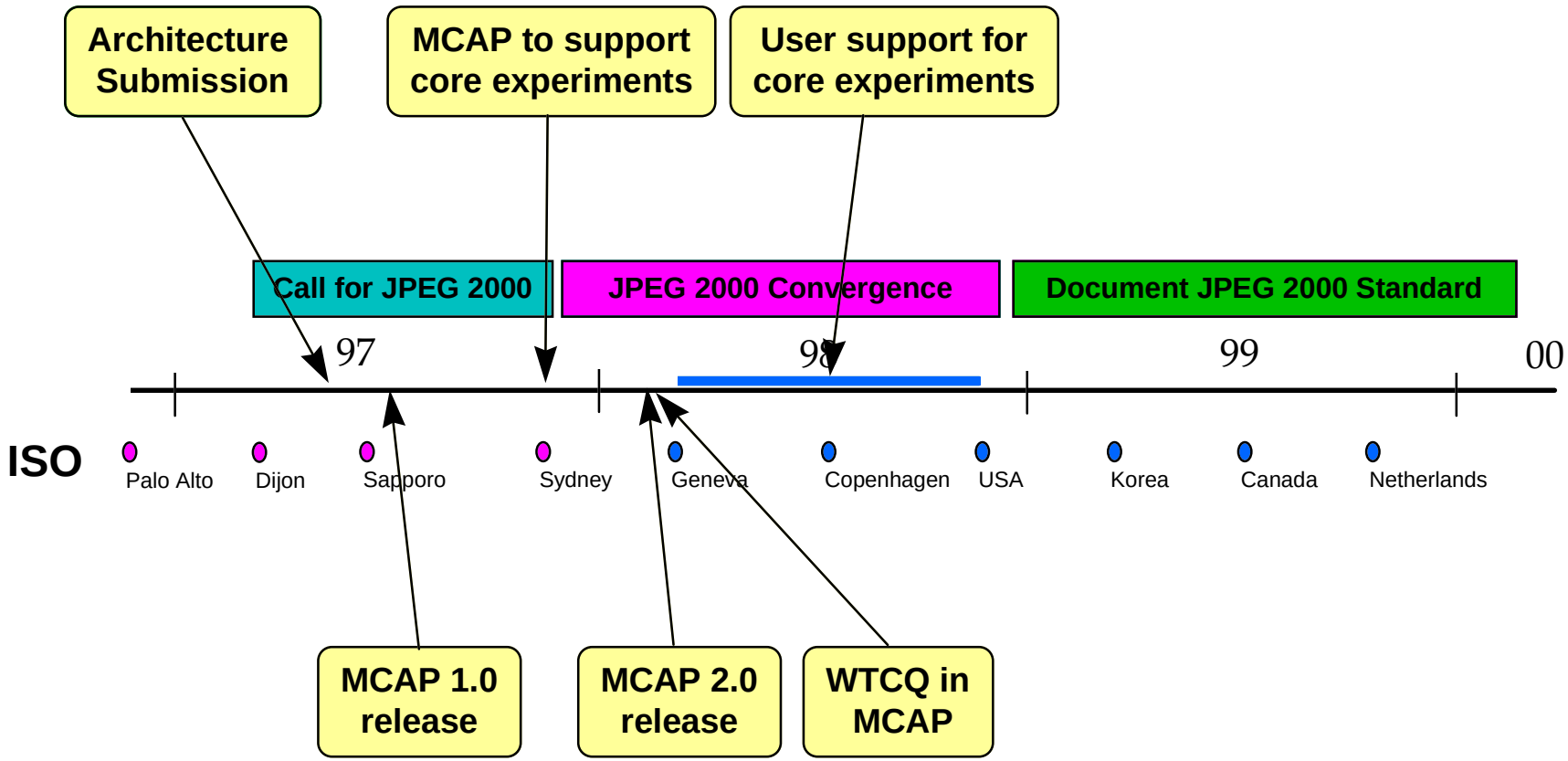
- »IDL - Technology-independent syntax describing object encapsulation
- »Facilitate distributed processing
- »Support C, C++, Ada, Smalltalk, COBOL, JAVA, Objective C

Standardized Module Interface




```
interface MBWE_IPS {
  void call(in AnyArray Input, out AnyArray Output,
            inout Options);
};

interface BWE_IPS {
  void call(in AnyArray Input, out AnyArray Output,
            inout Options);
};

interface BWE_DM {
  void call(in sequence<AnyArray> Input, out AnyArray Output,
            inout Options);
};

interface DIQ_D {
  void call(in BitVector Input, out sequence<BitVector> Output,
            inout Options);
};

interface DIQ_IQ {
  void call(in BitVector Input, out AnyArray Output,
            inout Options);
};

interface DIQ_IPS {
  void call(in AnyArray Input, out AnyArray Output,
            inout Options);
};
```

MCAP Schedule for JPEG 2000



JPEG 2000 Architecture Activities

- **N505 JPEG 2000 Call for contributions**
  - **included call for Architecture contributions**
  - **8+ submissions (July and Nov meeting)**
- **N612 Architecture Requirements Document**
  - **broad set of requirements**
- **N716 Requirements and Profiles**
  - **specific user and data requirements**