
Why Axis2: The Future of Web Services

Eran Chinthaka

Apache Software Foundation & WSO2

About Me ...

- PMC Member – Apache Web Services
- Apache Axis2 – Committer, Release Manager.
- Apache Synapse - Committer
- Member of W3C Working Group for WS-Addressing
- Working with WSO2 on Axis2, Synapse and Tungsten.

Agenda

- Introduction and Motivation
- Key Features of Axis2
- Other Improvements
- Summary and Conclusion

Motivation for Axis2

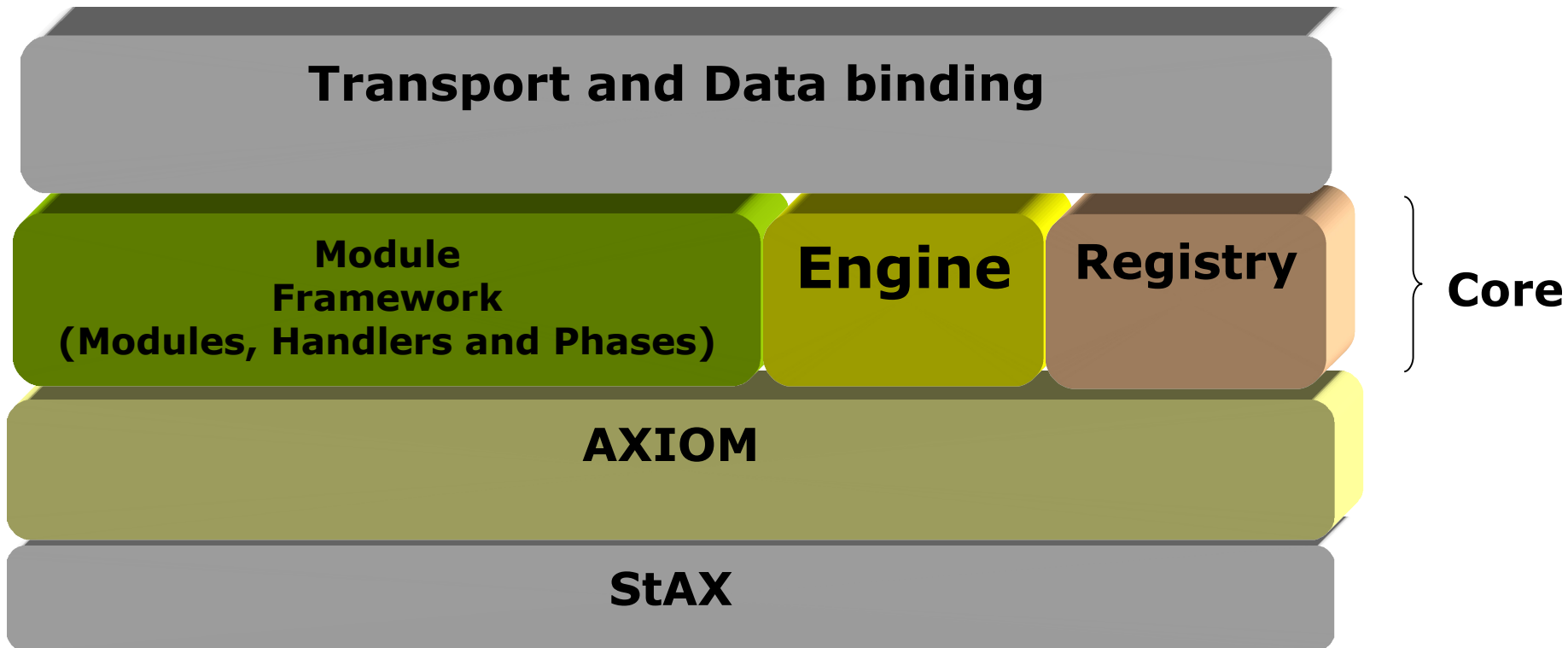
- History of ASF SOAP engines (paradigm)
 - Apache SOAP
 - Axis 1.x designed as a follow-on
 - Why do we need a new SOAP engine?
 - Changes to the Web services landscape
 - » WS-A, WS-RM
 - Performance
 - » Parsers, Optimizing based on use
 - Ease of use
 - » Deployment of new capabilities, service deployment

Key Features of Axis2

- New XML Infoset Representation
 - Deferred building
 - StAX integrated
- Extensible Messaging Engine
- Pluggable Module Architecture
- Improved Deployment Model
- New Client API
- Optional Pluggable Data Binding
- REST Support

Axis2 Architecture - The “*big picture*”

The Component View



New XML Infoset Representation

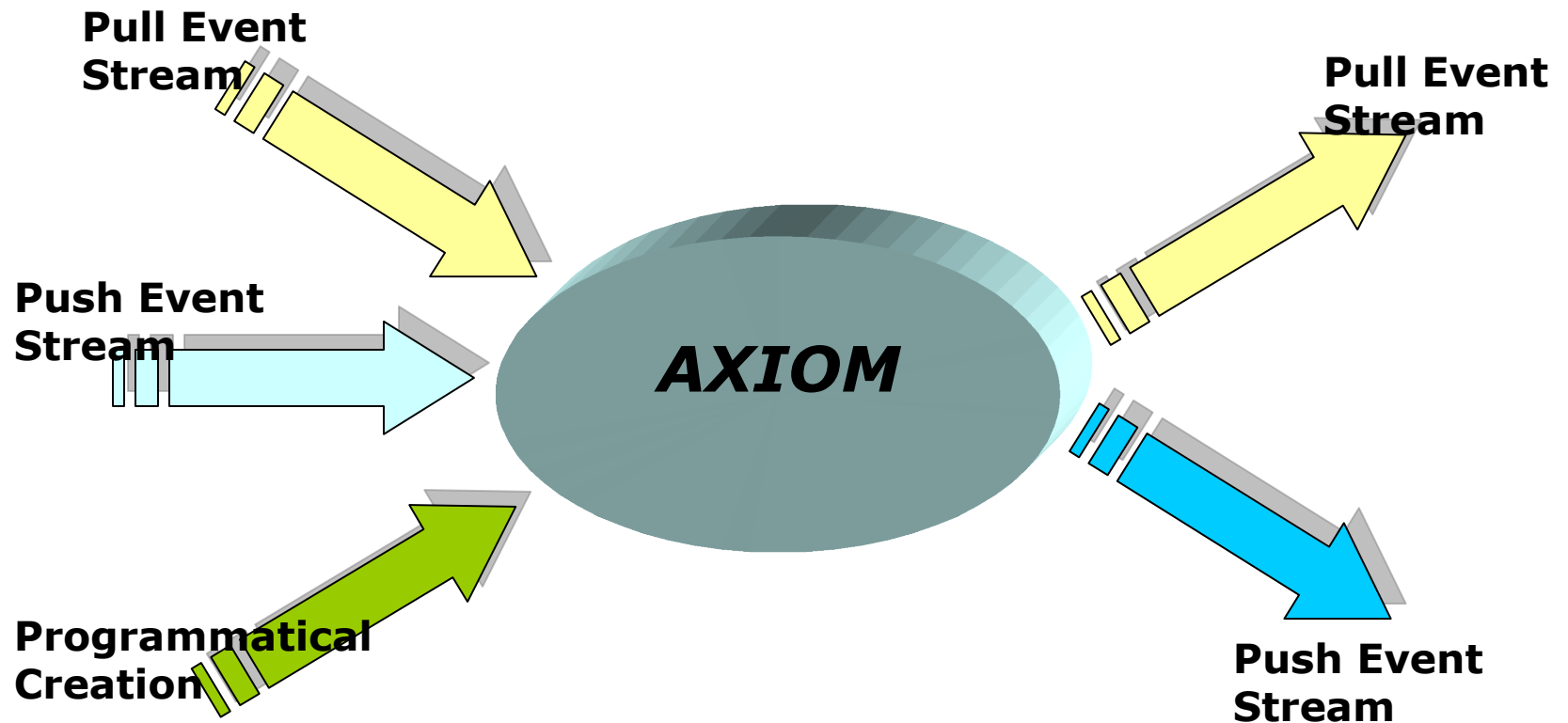
New XML Infoset Representation

- Known as AXIOM (**A**xis **O**bject **M**odel)
- NOT, Yet another XML object model
 - API is more like a simplified DOM
- Fundamental Difference ?
 - Objects are made “on demand” using a pull model
 - Allows direct access to the underlying pull stream with or without building the tree
 - Support for storing binary values

New XML Infoset Representation (Cont...)

- API also provides a StAX parser interface at any element
 - Allows the event based navigation of the OM tree.

New XML Infoset Representation (Cont...)



New XML Infoset Representation (cont..)

- In built binary storage support
 - Can store binary (unchanged)
 - Natively supports XOP/MTOM
- XOP? MTOM??

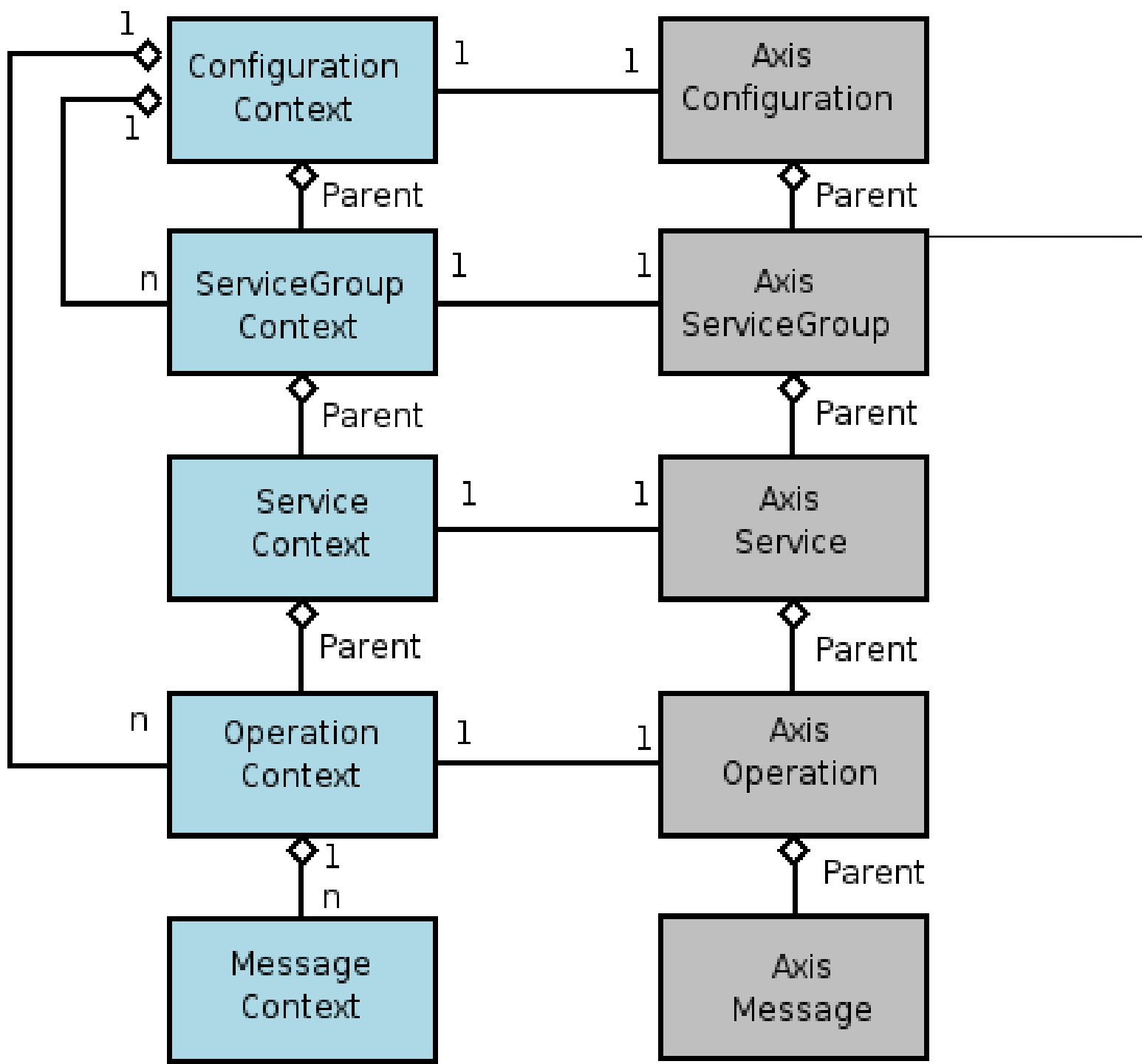
Use of AXIOM in Axis2

- AXIOM is the primary means of representing/manipulating the XML message inside Axis

Extensible Messaging Engine

Extensible Messaging Engine

- It is the “core”
 - A pure SOAP processor
 - Not aware of any java web service specifications (e.g. JAX-RPC)
 - Contains
 - A Context Hierarchy
 - Concept of descriptors and contexts
 - Phases



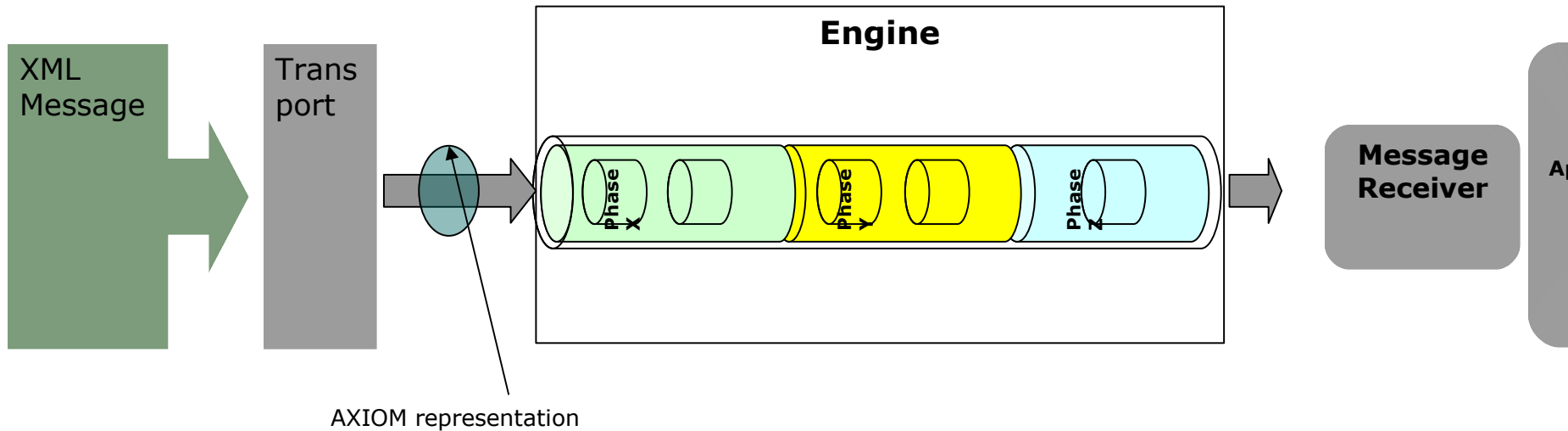
The Messaging Engine - Pipe view

Can be viewed as a pipe



Axis2 “big picture” (Cont..)

The “Dynamic” View



The Messaging Engine - Message Exchange Patterns

- Describes the exchange pattern of SOAP messages per given operation.
- E.g.
 - In – Out
 - In only
 - In – In – Out !
- WSDL 2.0 defines 8 standard MEPs.

The Messaging Engine - Message Exchange Patterns (cont..)

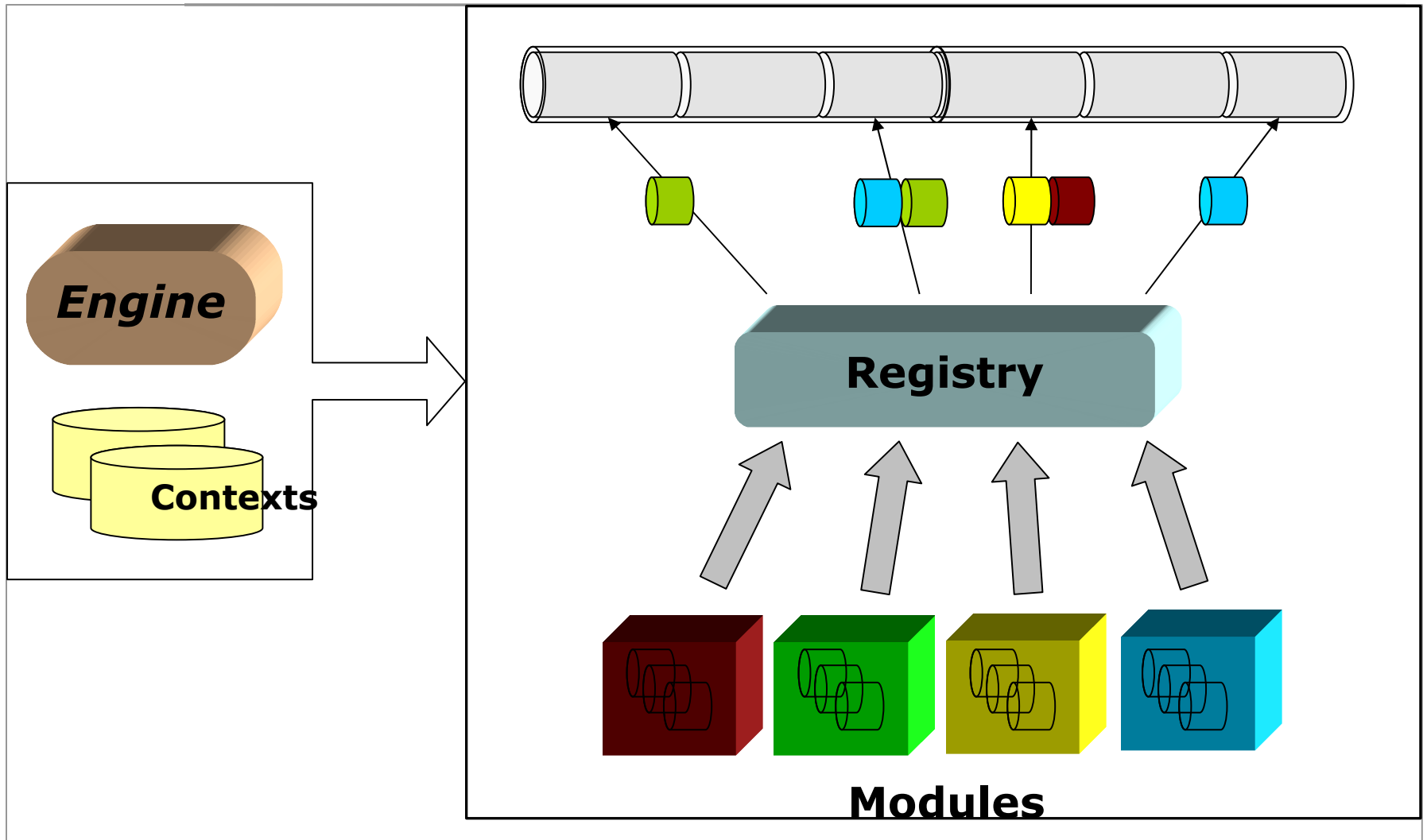
- Axis2 supports MEPs with the piped model of the engine
- Can be easily extended to support custom MEPs

The Messaging Engine - Phases (Cont...)

○ Concept of Phases

- A phase is a particular stage in execution.
 - E.g. pre-dispatch, Transport
- Each Phase contains a group of Handlers
- Handlers are linked to rules that say where the handler should go

The Messaging Engine – Detailed Components View



The Messaging Engine - Phases (Cont...)

○ Standard Phases

- Transport phase
- Pre-dispatch phase
- Dispatch phase
- Post-dispatch phase
- Policy determination phase

Pluggable Module Architecture

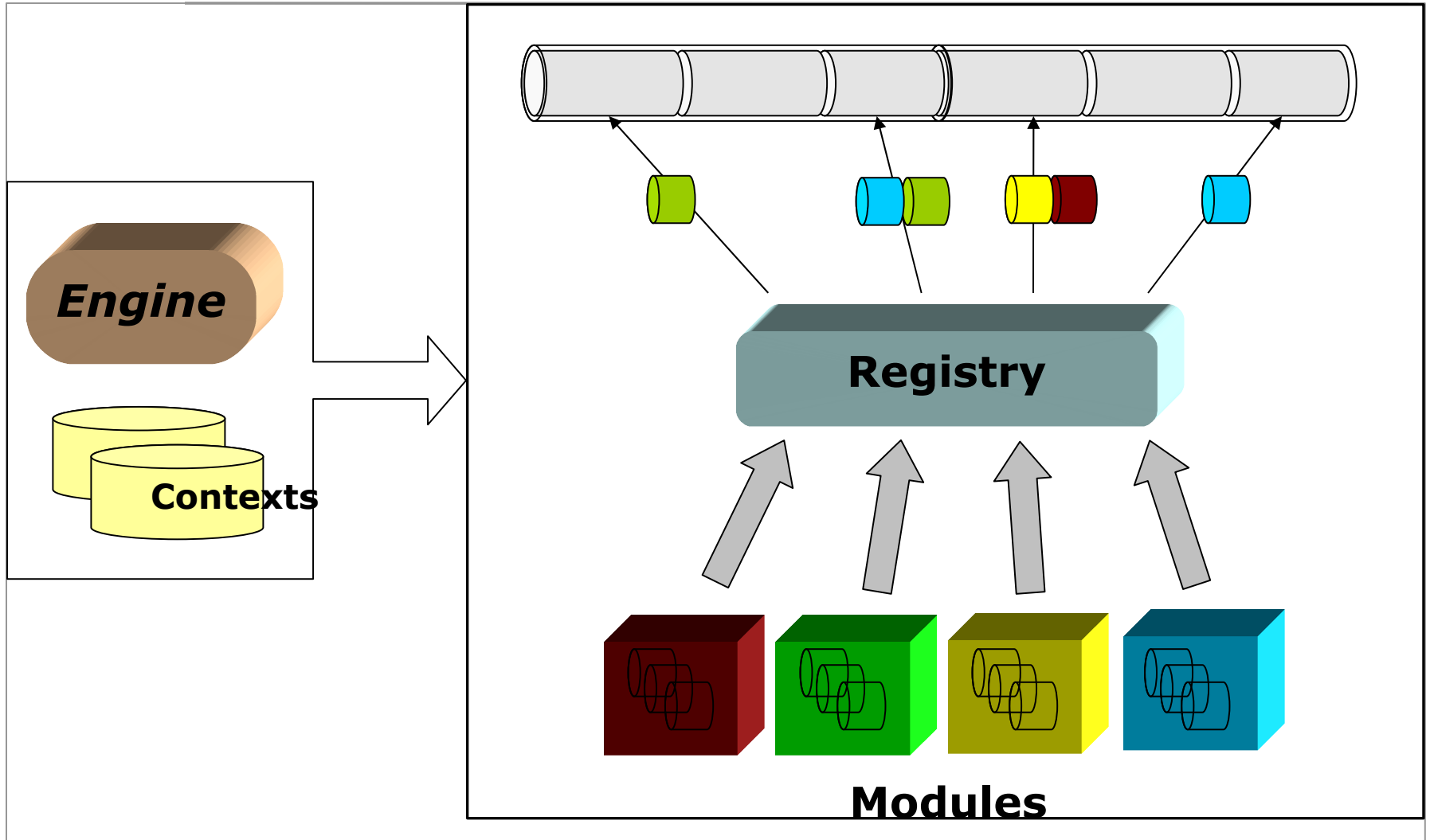
Pluggable Modules

- Extensions for the server
- Meant to provide a specific functionality
 - E.g. Security, Reliability etc
- Consists of
 - Handlers
 - Special Module class
 - A Descriptor (module.xml)

Pluggable Modules (Cont..)

- Archived to make a single bundle
 - Known as a “.mar” file
- Concept of ***Engaging***
 - Engaging a module means activating it
 - Can be done
 - Per System
 - Per Service
 - Per Operation

The Messaging Engine – Detailed Components View



Pluggable Modules - Example

- The WS-Addressing module
 - Contains
 - Addressing-in handler
 - Placed in the **pre-dispatch** phase – in pipe
 - Addressing out handler
 - Placed in the **message-out** – out pipe

Improved Deployment Model

Improved Deployment Model

- Faster and Easier Deployment
 - Service Archive files (.aar files)
 - A collection of resources needed for a service
 - Includes
 - Service implementation
 - Handlers (optional)
 - Service descriptor (services.xml)

Improved Deployment Model (Cont...)

- Hot Deployment
 - “Drop in” deployment
 - Uploaded through
 - File system
 - Axis2 web application

New Client API

New Client API

- Supports both blocking and non-blocking invocations models
 - Concept of callbacks for the client for non-blocking case
- Can handle both transport dependent and transport independent asynchrony.
- Based on the “MEP Client”
 - Default support to IN only and IN-OUT MEPs
 - Can be extended to support custom MEPs

Pluggable Data Binding

Pluggable Data Binding

- How to provide data binding support?
 - Use a well established framework
 - Avoid reinventing the wheel!
 - Flexible
 - Adds more functionality
 - E.g. XMLBeans supports the complete XML Schema
- Features an API that can bind to another data binding framework

Pluggable Data Binding (Cont ...)

- We have Axis data binding (ADB) framework by default.
 - Sufficient and proven to support simple data binding needs
- Currently supported frameworks
 - XMLBeans

REST Support

REST Support in Axis

- Services deployed in Axis2 can be invoked in REST manner
- A system wide switch to enable REST
- Selection of messages depending on WSDL 2.0 REST binding rules.
 - All REST messages will be treated similarly to SOAP messages inside the system

Other Improvements

- WSDL 2.0 compatible object model (WOM)
 - We have feeders to pump both WSDL 1.1 and WSDL 2.0 files in to WOM.
 - We have complete support for WSDL 1.1, full WSDL 2.0 support coming soon.

Improvements

○ Tools

- Complex operations need user friendly tools
 - Service and module archive generator tools
 - Axis Administration web application
 - Improved WSDL2Code tool
 - Eclipse and IDEA plug-ins

Improvements (cont ..)

- Other transports
 - SMTP / POP based transport
 - TCP based transport
 - JMS based transport
- Can easily switch between different transports
- Service Groups
- JMX Management Console (Google SoC project)

Improvements (cont ..)

- WSS4J support
 - Successfully completed interoperability with Indigo using Axis2 with WS-Security, MTOM and WS-Addressing
- Sandesha2
 - WS-Reliable Messaging implementation on Axis2
- DOOM – DOM over OM implementation

Improvements (Cont...)

- Support for other languages
 - Ability to write implementations in various JVM languages
 - Groovy
 - Client stub generation for languages other than Java !!!
 - Tested on IKVM based C# and VB.Net code !

Next Release

- Expect 1.0 very soon...
- 0.93 released on 2nd December, 2005.
- M1, M2, 0.90, 0.91 and 0.92 already released.....

Join with us

- Use Axis2 and give us feed back
- Let us know your specific requirements
- Join axis-dev@ws.apache.org or axis-user@ws.apache.org (remember to prefix subject with [Axis2])
- Enter in to Code and Samples competition and win nano-Pods from WSO2

Questions ?

Visit us at

<http://ws.apache.org/axis2/>

Thank you