Message Exchange Patterns

UNIT-IV

Introduction

- *Message exchange patterns (MEPs) represent a set of templates that provide a group of already mapped out sequences for the* exchange of messages
- Each MEPs addresses a common message exchange requirement
- **Primitive MEPs**
- Used before contemporary SOA
- one pattern that defines synchronous communication
- Most common example is a request and response pattern – receiver upon successful delivery responds to the initiator

Fire-and-forget

- Simple asynchronous pattern - based on the unidirectional transmission of messages from a source to one or more destinations

- **Variations of the fire-and-forget MEP :**
- single-destination pattern - source sends a message to one destination only.
- multi-cast pattern - a source sends messages to a predefined set of destinations
- broadcast pattern - similar to the multi-cast pattern, except message is sent to a broader range of recipient destinations.

Complex MEPs

- Primitive MEPs assembled in various configurations to create different types of messaging models – *Complex MEPs*
- *Ex Publish-Subscribe Model*
- Step1: subscriber notifies the publisher to receive messages on a particular topic.
- Step 2: Upon receiving message, the publisher broadcasts messages on the particular topic to all of that topic's subscribers
- Also an example of aggregate primitive MEPs
  - Step1 is implemented by a request-response MEP
  - Step2 using fire-and-forget patterns, allowing the publisher to broadcast a series of unidirectional messages to subscribers

MEPs with SOAP, WSDL & SOA

- **MEPs and SOAP**
- SOAP standard messaging framework supports single-direction message transfer
- Its extensible nature allows numerous messaging characteristics and behaviors via SOAP header blocks
- It also has optional parameter to be set to identify the MEP associated with a message

- **MEPs and WSDL**

- Role of MEPs in WSDL service descriptions is they coordinate the input and output messages associated with an operation
- WSDL operations support different configurations of incoming, outgoing, and fault messages

Contd…

- Configurations are equivalent to MEPs specification and referred as patterns
- Patterns are applied to service operations on service provider side

- 4 Basic patterns supported by WSDL 1.1
- Request-response operation - Upon receiving a message, the service must respond with a standard message or a fault message
- Solicit-response operation - Upon submitting a message to a service requestor, service expects a standard response message or a fault message
- One-way operation - service expects a single message and is not obligated to respond.
- Notification operation - service sends a message and expects no response.

WSDL Extended MEPs

- WSDL specification 2.0 extends MEP with eight patterns:
- *in-out pattern - comparable to the request-response MEP*
- *out-in pattern - reverse of the previous pattern, where service provider initiates the exchange by transmitting* the request (Equivalent to the WSDL 1.1 solicit-response operation)
- *in-only pattern - supports the standard fire-and-forget MEP. (Equivalent to the WSDL 1.1 one-way* operation)
- *out-only pattern - reverse of the in-only pattern, used primarily in support of event notification. (Equivalent to* WSDL 1.1 notification operation.)

Contd…

- *robust in-only pattern - a variation of in-only pattern provides option of launching a fault response message as a* result of a transmission or processing error.
- *robust out-only pattern – same as out-only pattern, has an outbound message initiating the transmission,* difference is a fault message can be issued in response to the receipt of this message
- *in-optional-out pattern - similar to in-out pattern with one exception. This variation introduces a rule stating that* the delivery of a response message is optional
- *out-optional-in pattern is the reverse of the in-optional-out pattern, where the incoming message is optional. Fault* message generation is again supported.

MEP and SOA

- MEPs are highly generic and abstract in nature
- It relates to an interaction between two services
- MEP relevance to SOA = MEP relevance to the abstract Web services framework
- Hence it is a fundamental and essential part of any Web services-based environment, SOA