NSPF: Designing a Notification Service Provider Framework for Web Services

Bahman Kalali, Paulo Alencar and Donald Cowan

University of Waterloo, CANADA
School of Computer Science
Computer Systems Group
{bkalali, palencar, dcowan}@csg.uwaterloo.ca

Outline

- Introduction
- Background: Three-role service-oriented architecture
- Motivation to extend the three-role architecture
- NSPF: Use Case Scenarios
- Design of the NSPF
- Patterns applied to the design of NSPF
- Web service hierarchy of events
- Conclusions and future work

Introduction

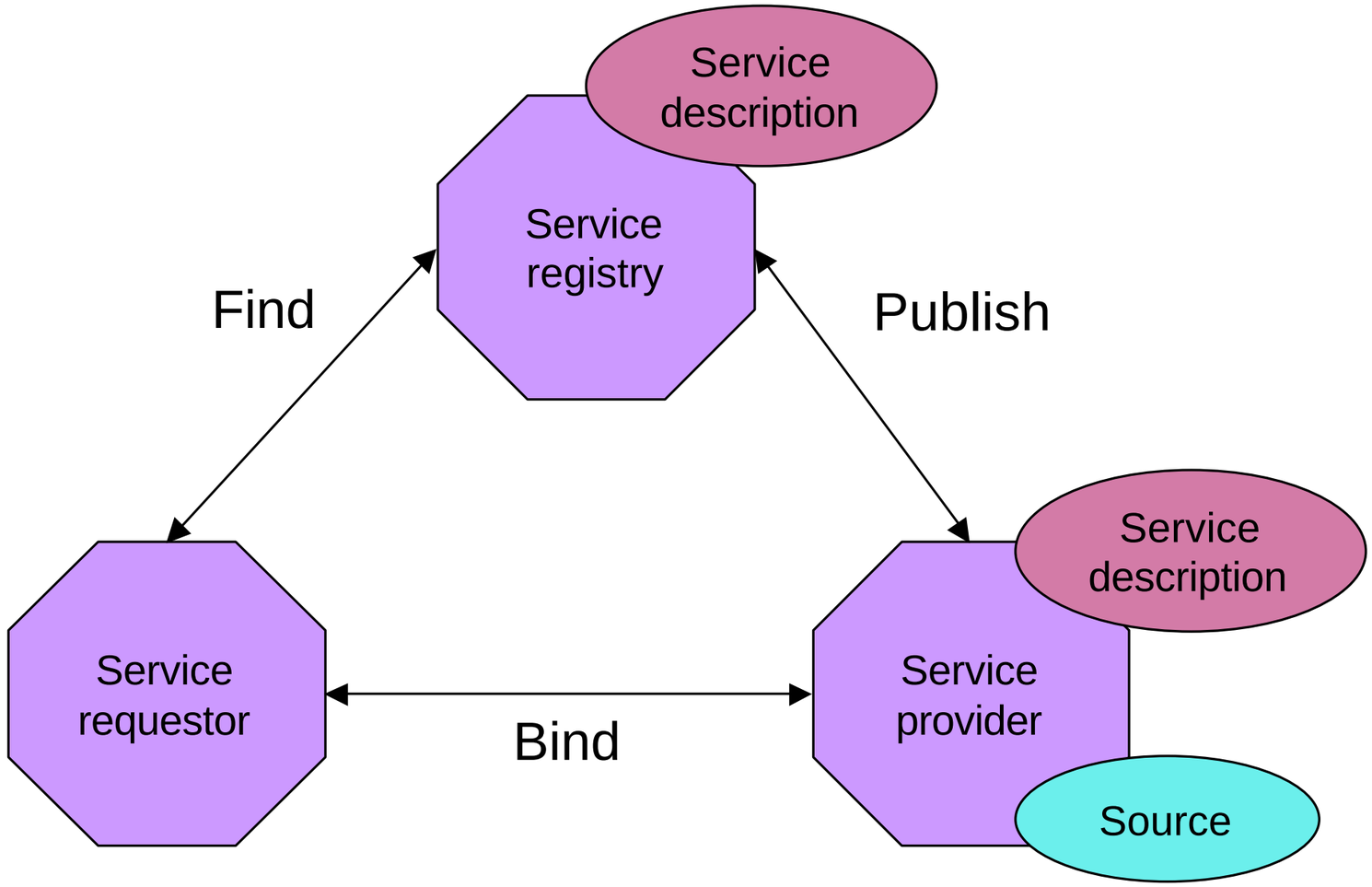
Focus Of This Research Project

- Finding problems related to changes in the interface of Web services and availability of Web services
- Providing a solution in the form of a service-oriented notification provider framework
- Using object-oriented design patterns to design and document a service-oriented framework

Service-oriented Architecture

- A typical Web service architecture represents three roles:
  - Service provider
  - Service requestor
  - Service registry
- It describes three basic operations:
  - Publish
  - Find
  - Bind
- It also has two artifacts:
  - Service
  - Service description

Service-Oriented Architecture



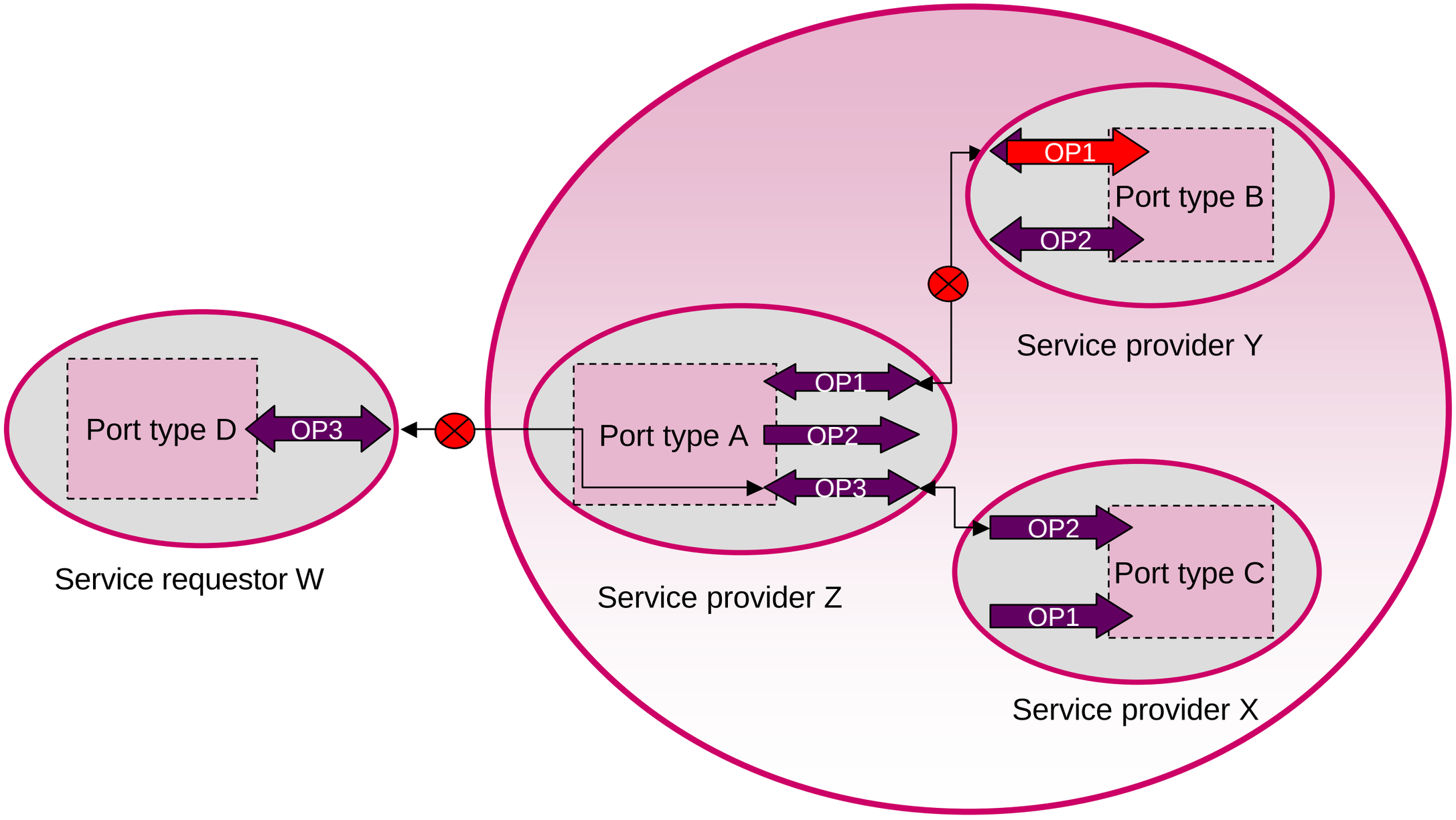
A typical three-role architecture

Motivation to Extend the Three-Role Architecture

- What happens when a Service Provider
  - changes its Web Service interface description?
  - registers a business in UDDI without providing any service?
  - becomes disabled?
  - provides a new service?

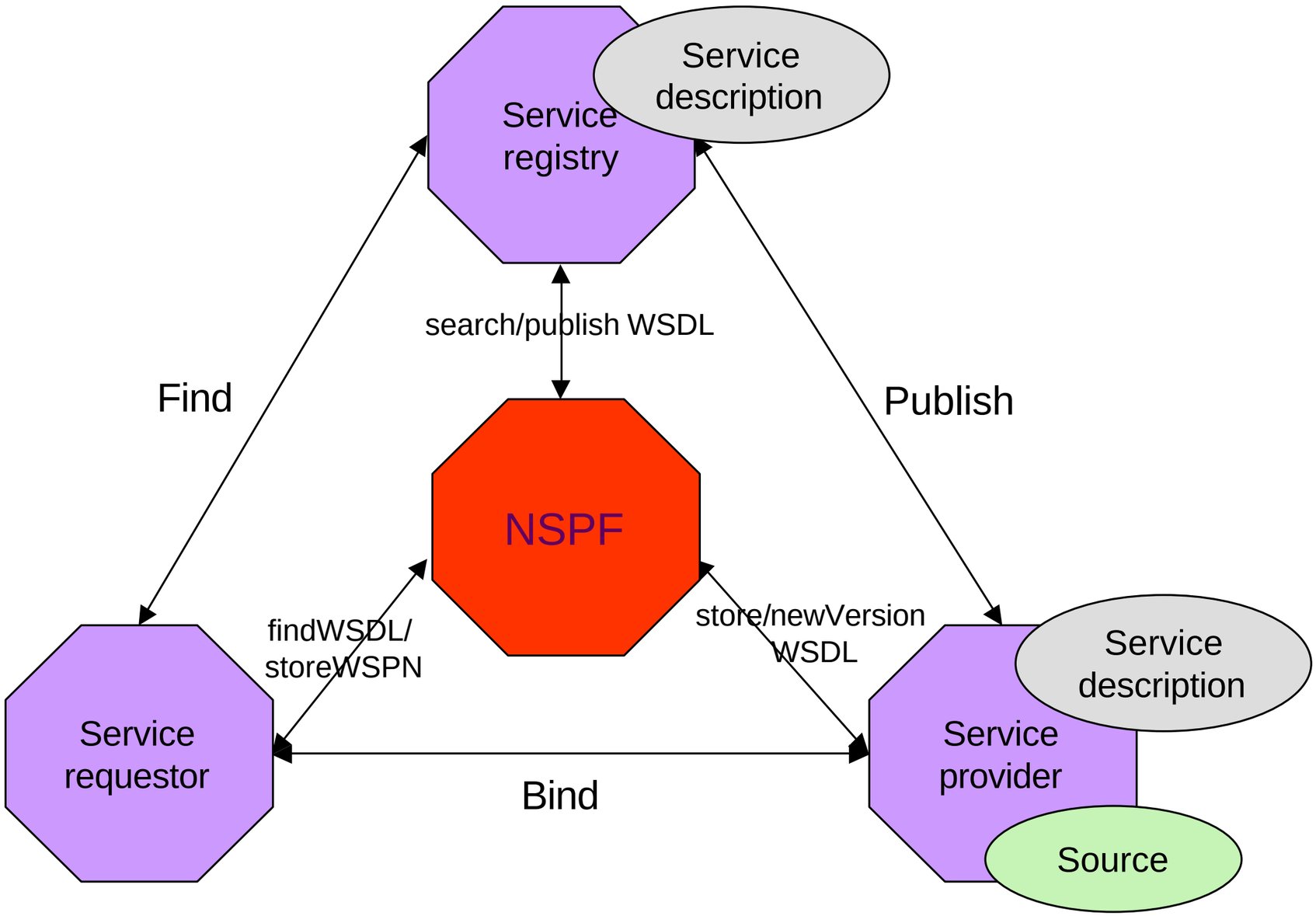
- Proposed solution
  - A Notification Service Provider Framework (NSPF)

Sample Problem



Web Service Aggregation

Extension to Three-Role Architecture



Four-Role Architecture

NSPF: Use Case Scenarios

- Notification of changing Web Service Interface
  - Service providers send new version of their interfaces to NSPF
  - NSPF is responsible to keep track of web service descriptions
  - NSPF notifies service requestors about new version of an interface

- Notification of Inactive Web service provider and replacement
  - NSPF periodically re-binds to each service provider to find disabled services
  - NSPF notifies all the service requestors about disabled services
  - NSPF finds replacement for disabled services and notifies requestors

NSPF: Use Case Scenarios (Cont'd…)

- Notification of a new available Web service Interface
  - Service providers for the first time send their interfaces to NSPF
  - NSPF notifies all requestors that used similar interfaces before
- Notification of top-ranked Interfaces to Web service requestor
  - NSPF periodically searches for each service requestor in its local native database
  - NSPF finds *portType* that is used by a service requestor
  - NSPF finds all the service providers that implemented the same *portType*
  - NSPF selects the service provider that has the highest number of service requestors associated with the *portType*
  - NSPF notifies the service requestor about the top-ranked service provider

Design of the NSPF

- Based on Multi-Tier Architecture : Four Layers
  - Proxy Layer
  - Web Server Layer
  - Application Notification Server Layer
  - Application Worker Layer
- Basic Operations
  - *storeWSDL*
  - *findWSDL*
  - *storeWSPN*
  - *newVersionWSDL*

Proxy Layer

Web Server Layer

Application Notification Server Layer

Application Worker Layer

<<Interface>>
HTTPInterface

HTTPServer

SOAPProcessor

1

<<interface>>
WManager
interface

WSNSProvider

+ storeWSDL ()
+ findWSDL ()
+ storeWSPN ()
+ newVersionWSDL ()

- subscribers

*

<<interface>>
Subscriber

+ inform ()

1

EventService

+ subscriber
+ filter
# subscription list
- event type

+ publish ()
+ subscribe ()
+ unsubscribe ()

Event

*

1

*

1

*

- filters

*

<<interface>>
Filter

+ apply ()

WSDLPublisher

WManager

1

- workers

*

<<interface>>
WorkerInterface

InterfaceProvider

- provideID
- interfaceID

- InterfaceUsed

0.*

*

1

InterfaceRequestor

- requestorID

*

OODatabase
Worker

1

ServiceController
Worker

WSPN

1

*

NativeXMLDatabase
Worker

*

1

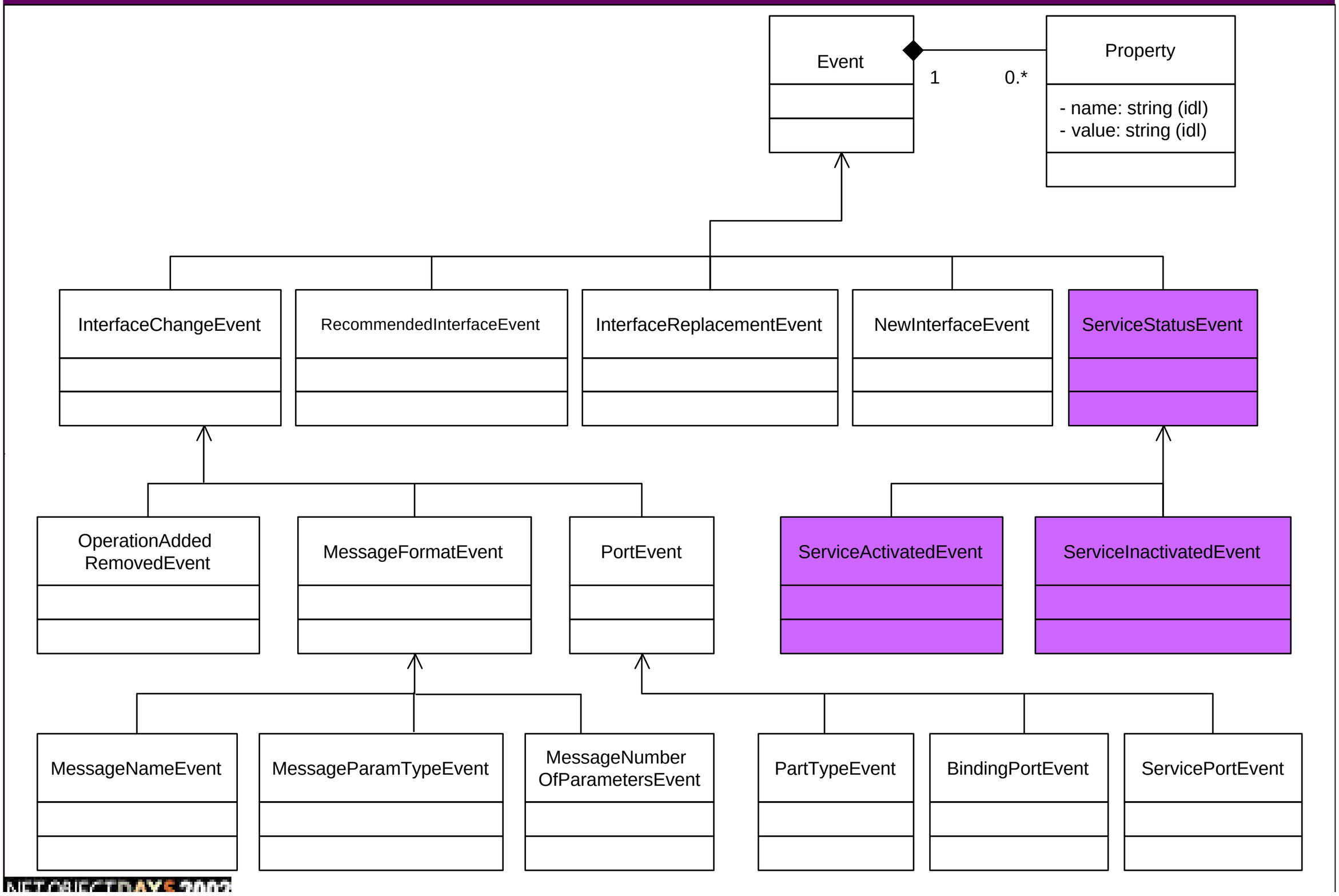
WSDL

ServiceDiscovery
Worker

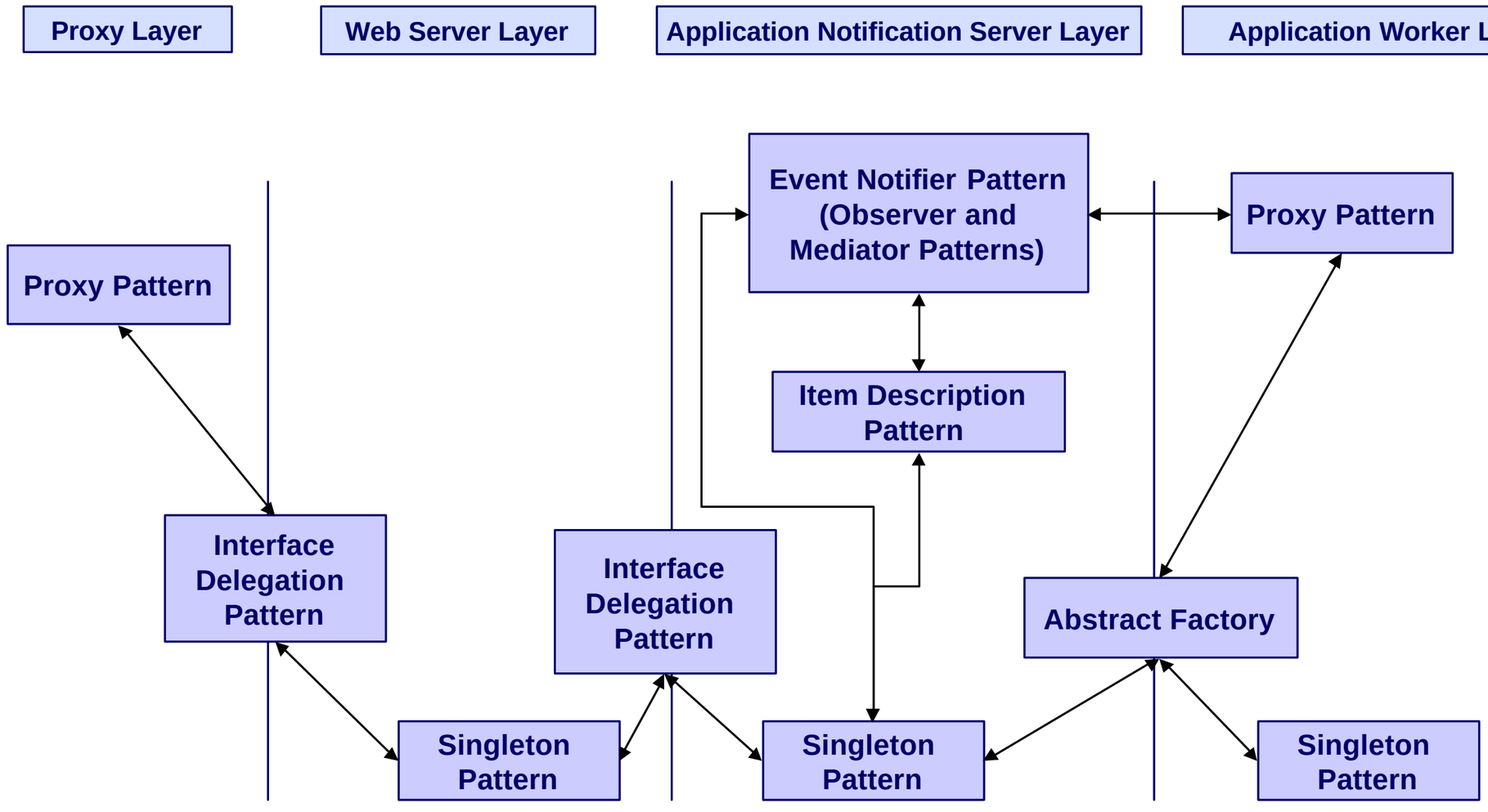
UDDIClient

It contacts UDDS registry
to find services

Web Service Hierarchy of Events



Patterns Applied to the Design of NSPF



Conclusions and Future Work

- NSPF is an extension to the Web service three-role architecture
- NSPF is designed with Software Engineering principles in mind
- A first prototype:
  - Storing WSDL files into Xindice XML Database
  - Using XPath expressions to define WSPN

- Future Work
  - Improving the design to support content-based publish-subscribe
  - Defining XML Schema for Web service event types
  - Improving the design to support scalable distributed native XML databases
  - Implementing a NSPF plug-in prototype using Eclipse
  - Regeneration of Proxies (Auto-Proxy Regeneration)