GIPC Extensibility

Instructor: Prasun Dewan (FB 150, dewan@unc.edu)

Goals

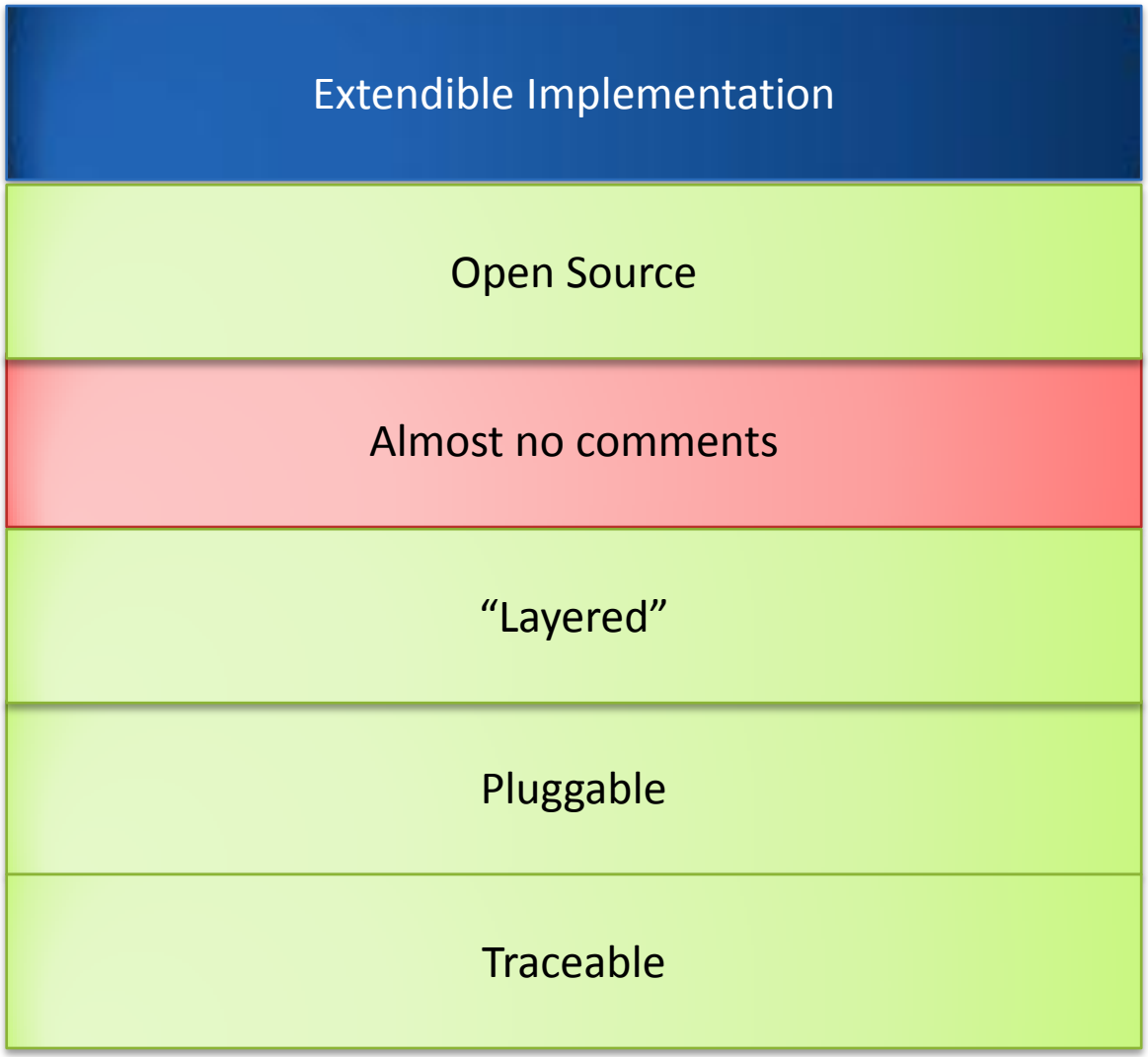
Customizable Implementation

Unlike RMI runs on Android

Informed by the IPC Design Space

How can we “improve” distributed IPC without language support?

Implementation Goal



Multi-Dimension Layering?

```
public enum PortKind {
  CLIENT_INPUT_PORT,
  SERVER_INPUT_PORT,
  SESSION_PORT,
  MULTI_SERVER_PORT
}
```

???

```
public enum PortMessageKind {
  BUFFER,
  OBJECT,
  RPC
}
```

extends

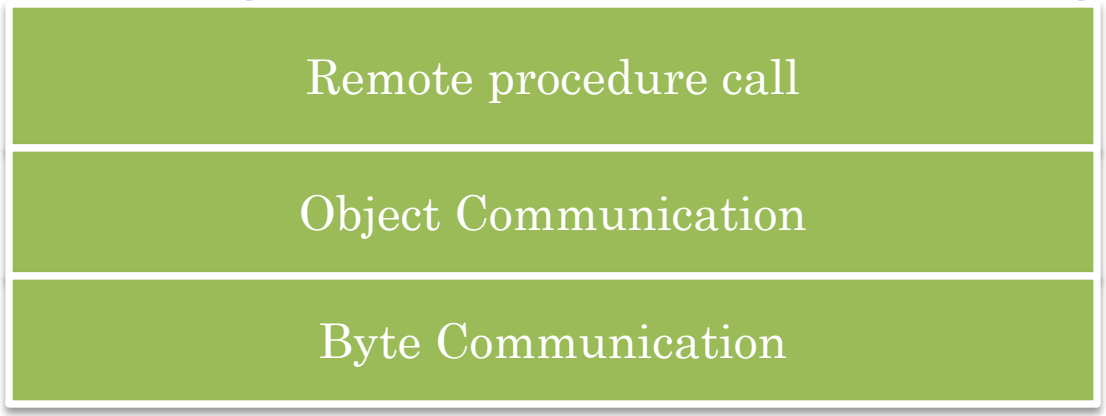
```
public enum PortAccessKind {
  SIMPLEX,
  DUPLEX,
  GROUP
}
```

extends

```
public interface PortDescription {
  PortKind getPortKind();
  void setPortKind(PortKind aPortKind);
  PortAccessKind getPortAccessKind();
  setPortAccessKind(PortAccessKind aPortAccessKind);
  PortMessageKind getPortMessageKind();
  setPortMessageKind(PortMessageKind aPortMessageKind);
  …
}
```

GIPC Message-Based "Layering" Dimension

```
public enum PortMessageKind {
  BUFFER,
  OBJECT,
  RPC
```

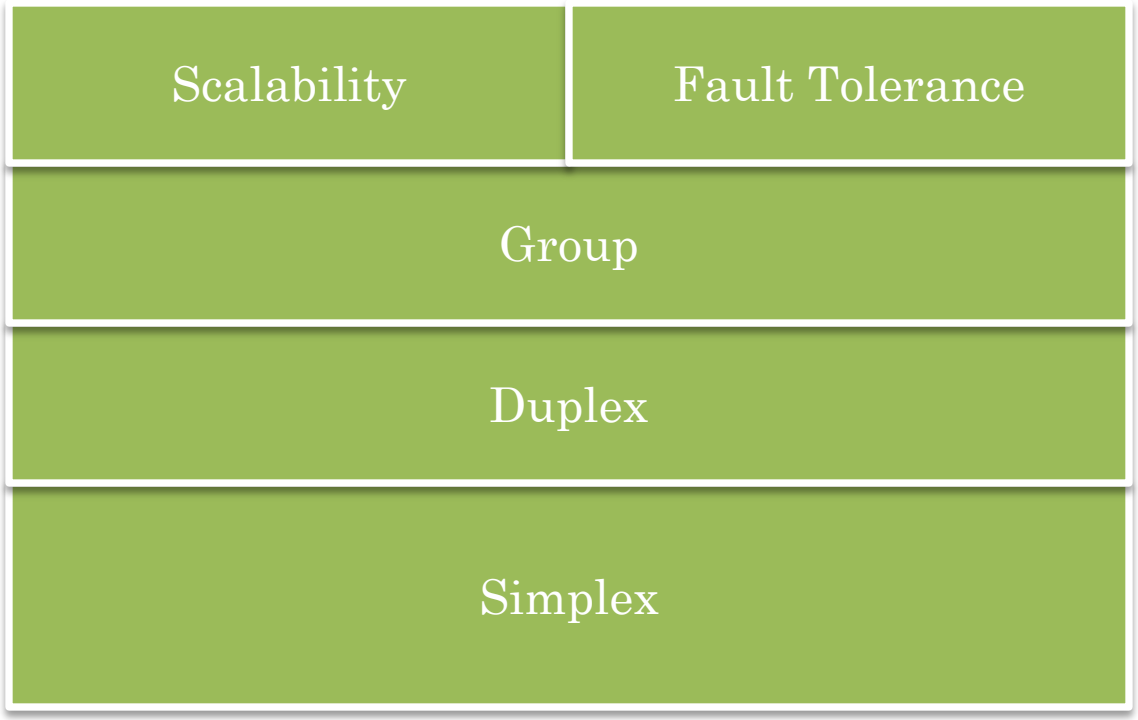


Common generic interfaces for object and byte communication

Object hides byte but RPC sees Object

GIPC Access-Kind Based Layering

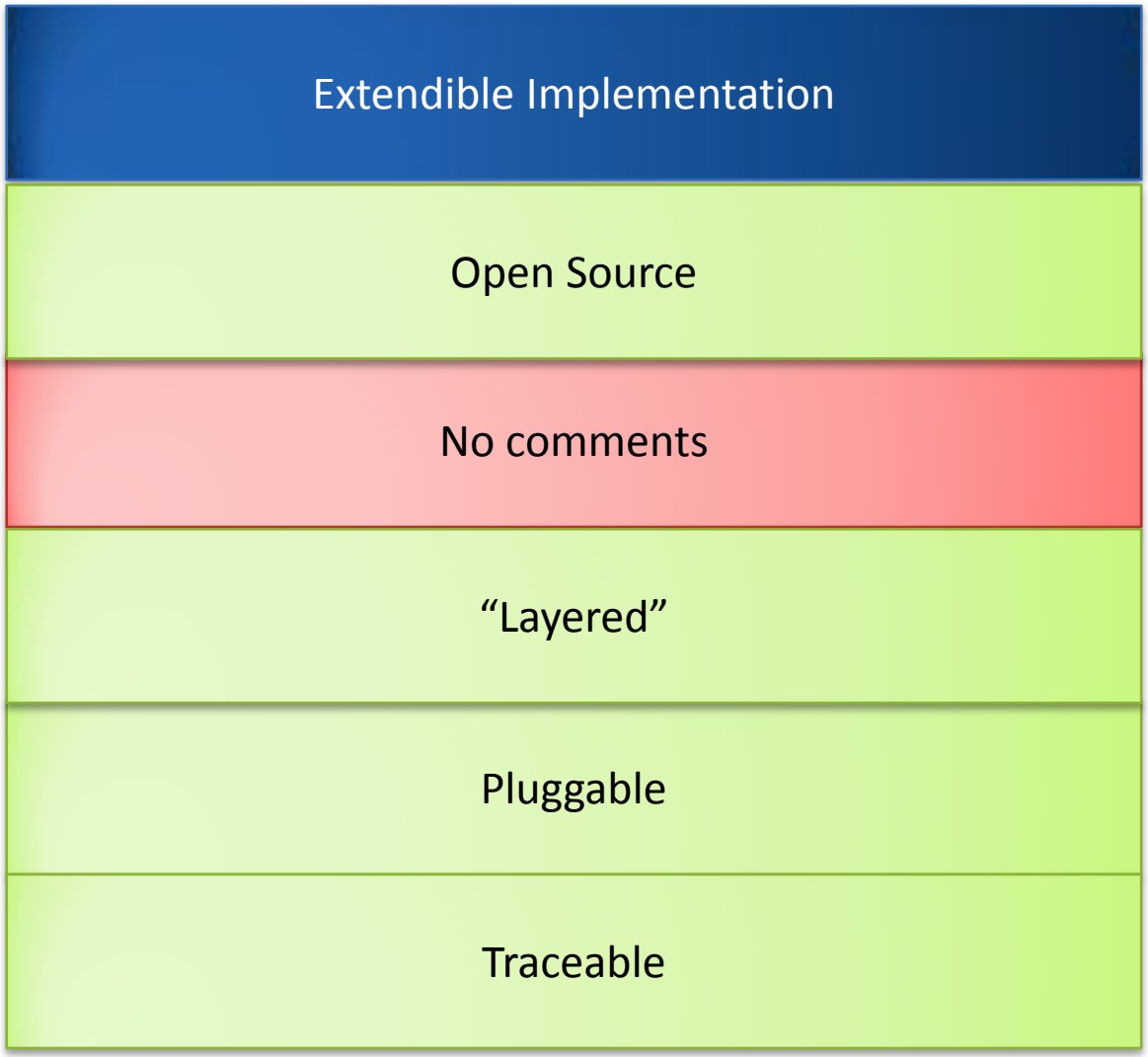
```
public enum PortAccessKind {
  SIMPLEX,
  DUPLEX,
  GROUP
}
```



Port-Kind

```
public enum PortKind {
  CLIENT_INPUT_PORT,
  SERVER_INPUT_PORT,
  SESSION_PORT,
  MULTI_SERVER_PORT
}
```

Implementation Goal



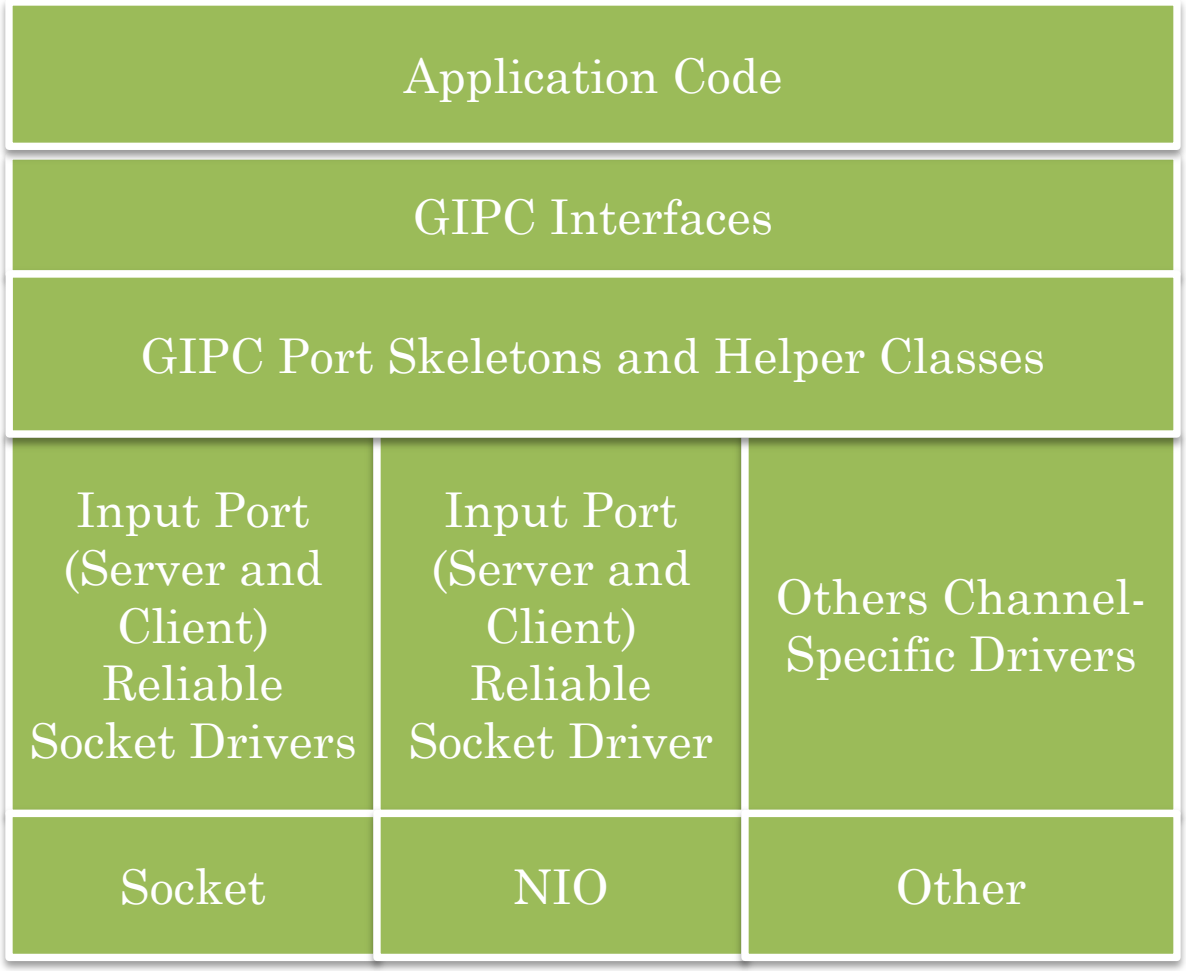
Plug in Models

Add Observer

Replace existing layer

Interpose new layer between two existing layers

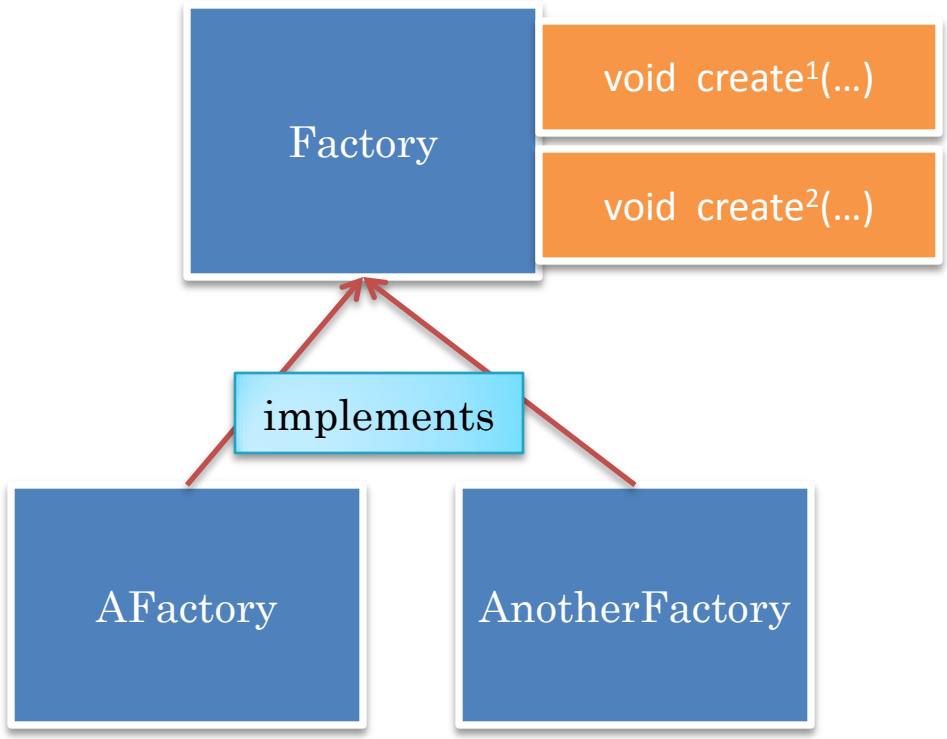
Replace Example



Channel: Mechanism that delivers and receives messages on the network

May want to choose which driver to use

Factory-Based Instantiation



Factory instantiates a set of related types such as ports

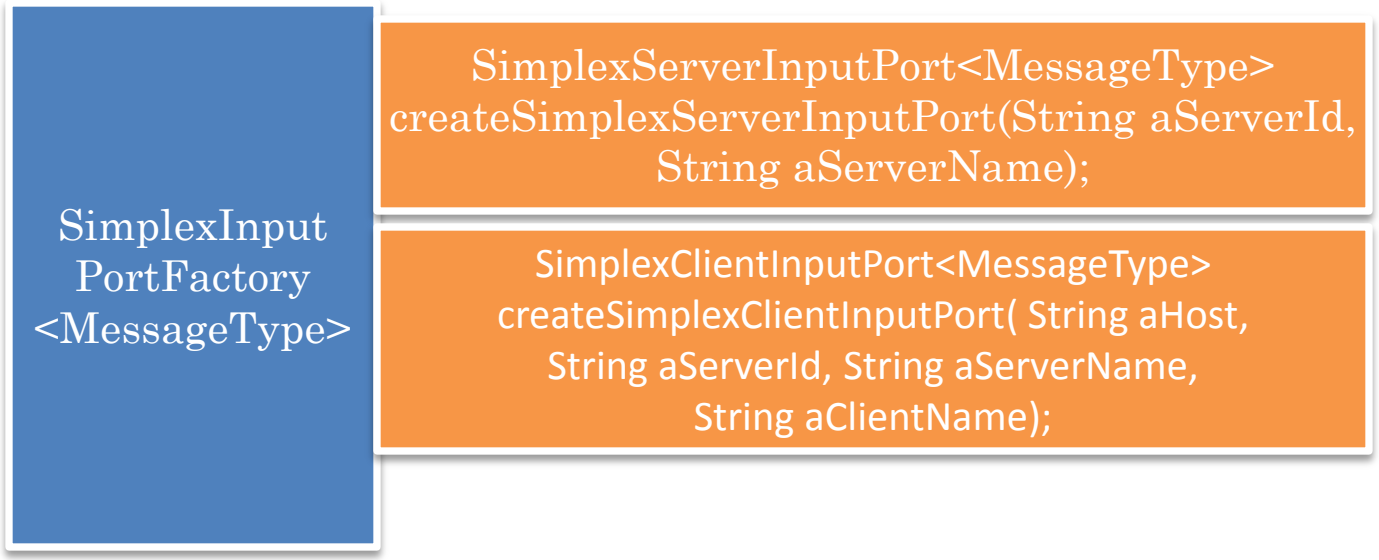
Provides a method for creating each type of object

Each method takes instantiation parameters

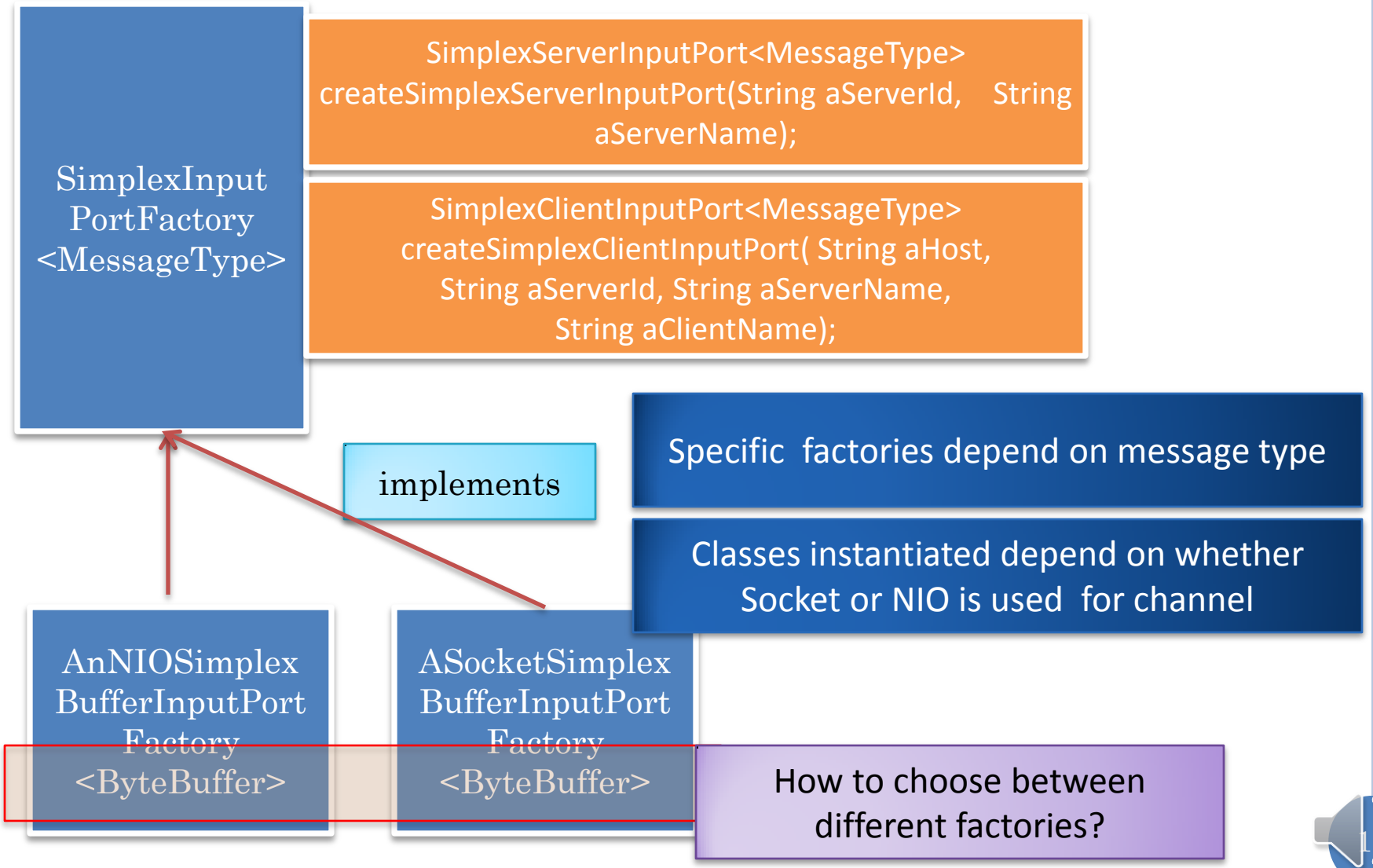
Different implementations can instantiate different classes

Typically instantiation parameters become constructor parameters

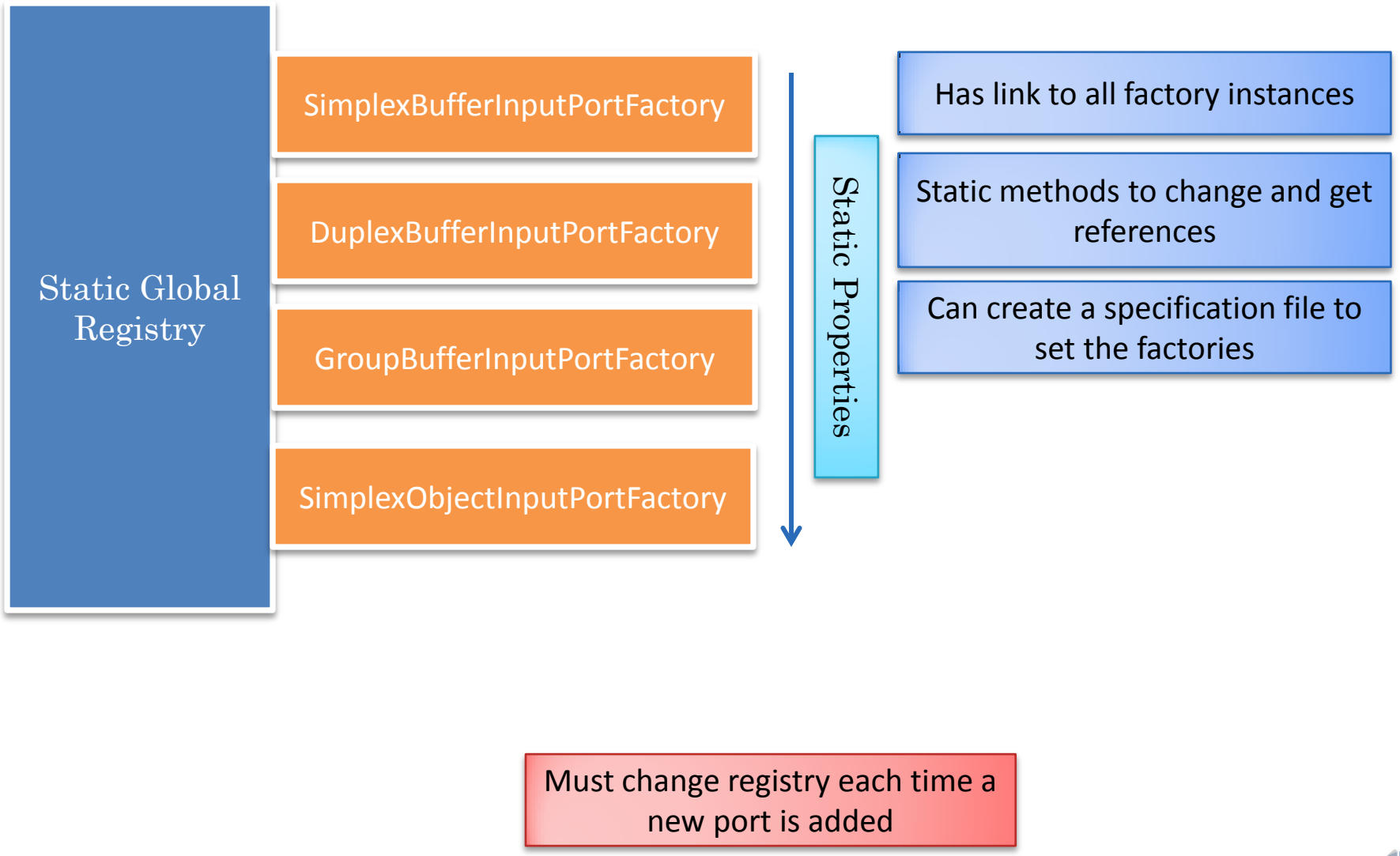
Simplex Input Port Factory Interface



SIMPLEX INPUT PORT FACTORY

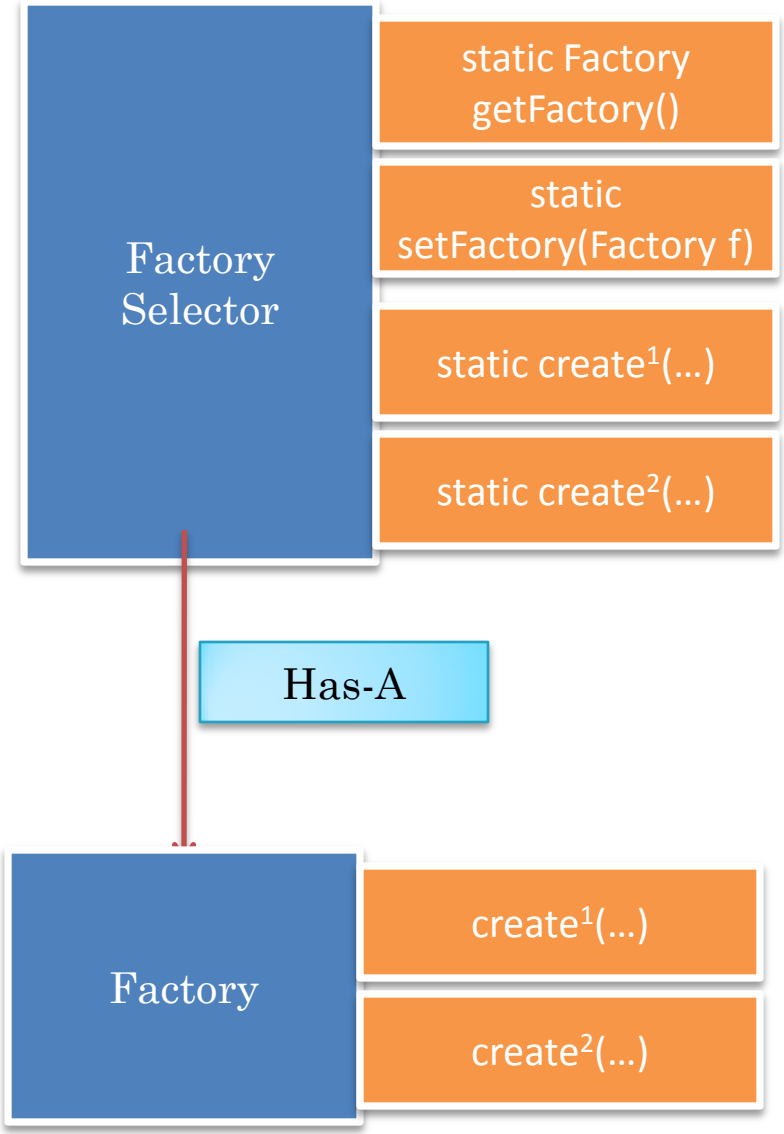


Global Abstract Factory



Must change registry each time a new port is added

Multiple GIPC Abstract Factories



General Idea

Helps choose between factories

GIPC

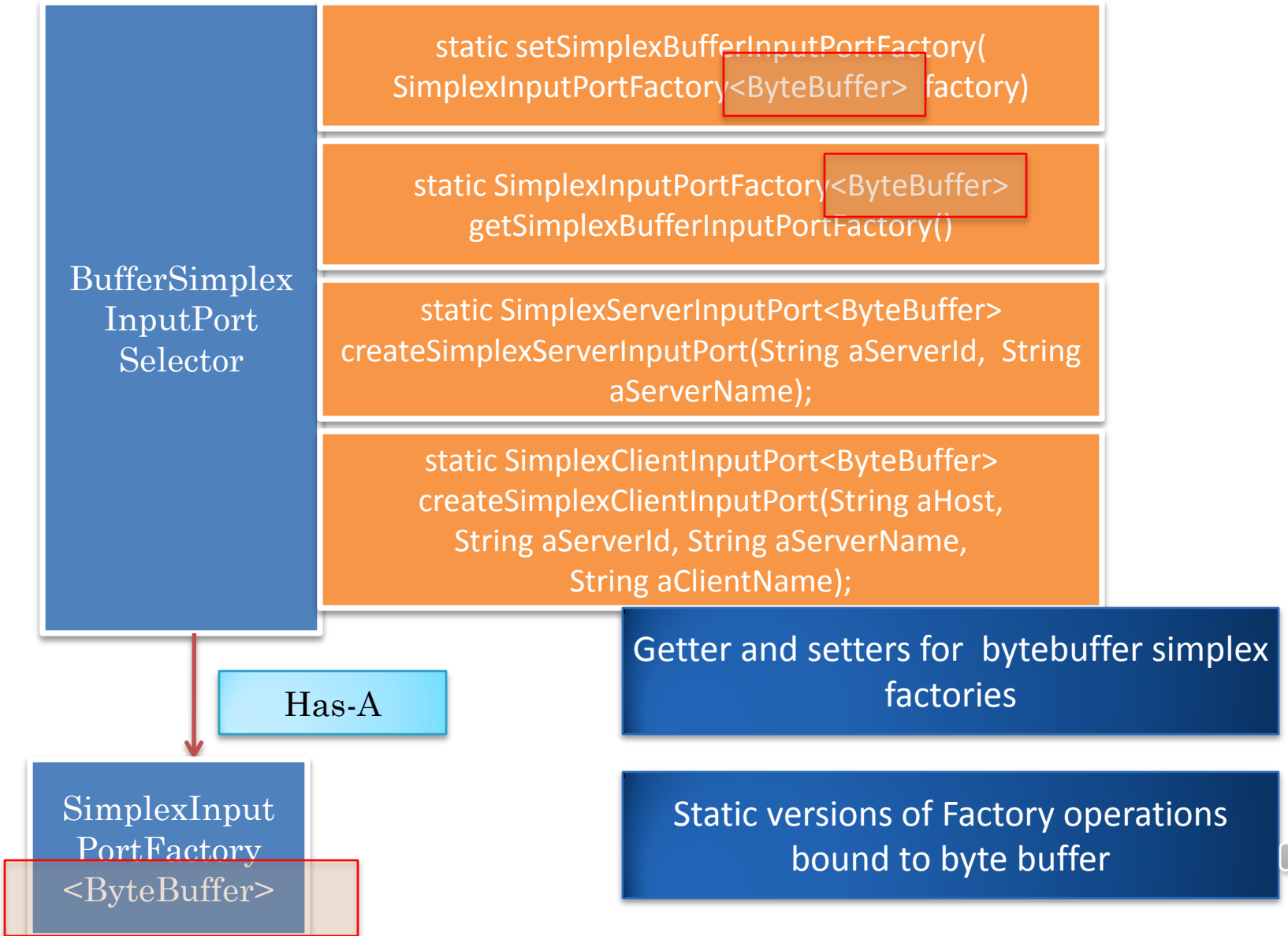
Has a link to a Factory instance

Static methods to change and get reference

Can get reference to factory and invoke factory methods

Abstract factory provides static versions of factory creation methods for convenience

GIPC Simplex Abstract Factories



Selector

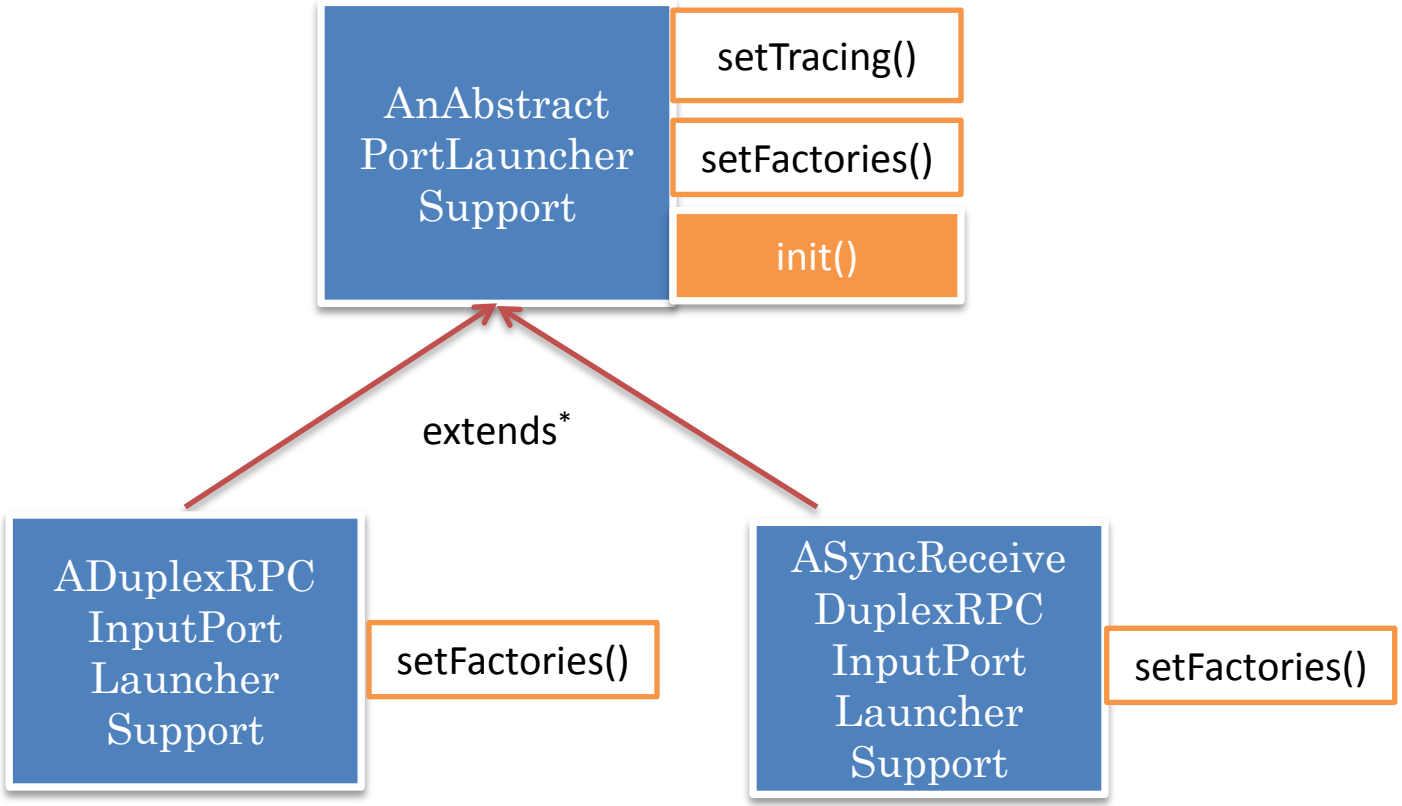
```
SimplexServerInputPort<ByteBuffer> aServerInputPort =
      BufferSimplexInputPortSelector.createServerSimplexInputPort(
                SERVER_PORT, SERVER_NAME);
aServerInputPort.connect();
```

```
SimplexClientInputPort<ByteBuffer> clientInputPort =
      BufferSimplexInputPortSelector.createClientSimplexInputPort(
                SERVER_HOST, SERVER_PORT, SERVER_NAME, clientName);
clientInputPort.connect();
```

How to assign factories to selectors?

How to reduce overhead of setting related factories for each configuration?

For each Port: Port Launcher Support

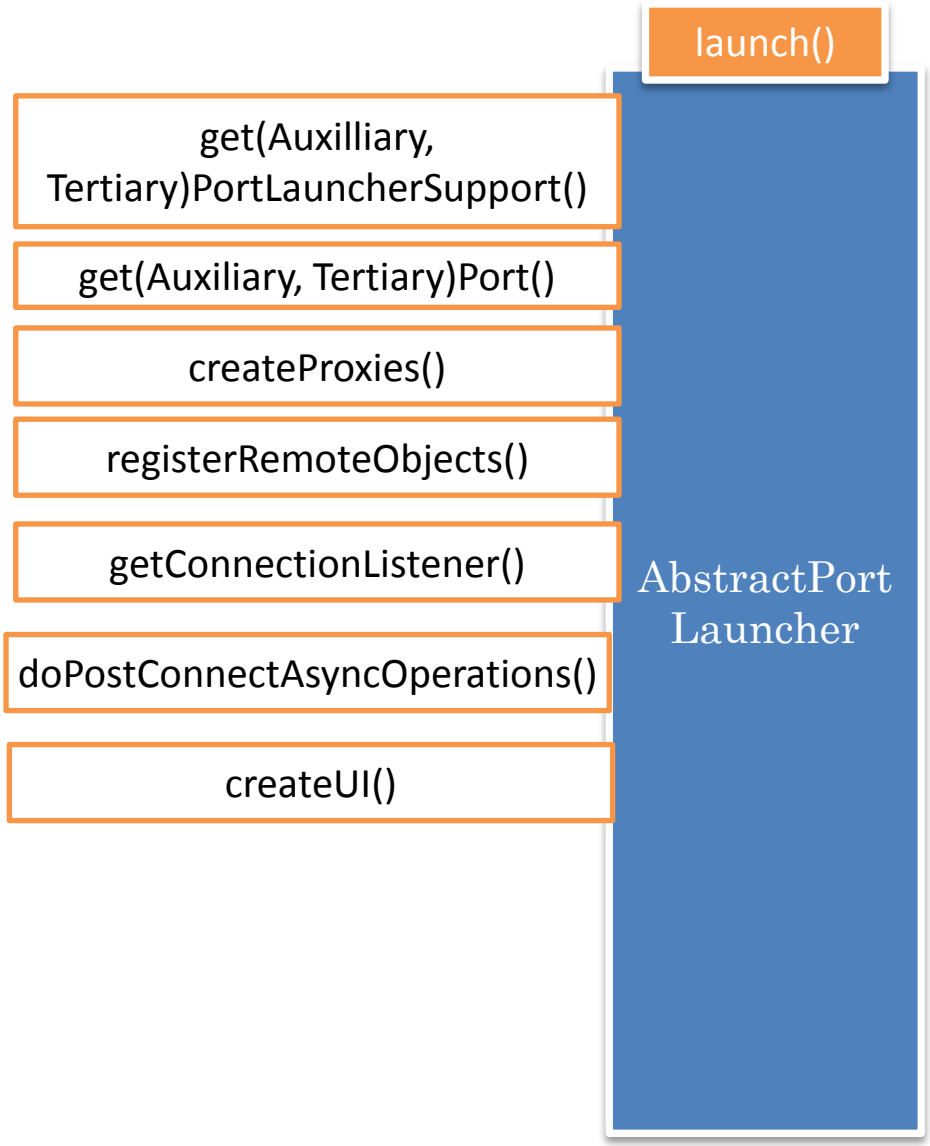


Port launcher support sets factories for a set of related selectors

Can do additional actions such as set tracing for objects created by these selectors

Before a port is created, the init() call is expected to be called on one or more PortLauncherSupport objects, which are expected to set the factories needed to instantiate that port and set constructor parameters of the objects created by the factories

Launcher vs. Launcher Support



gets port launcher support for main, auxilliary and tertiary ports) and initializes them (to create factories)

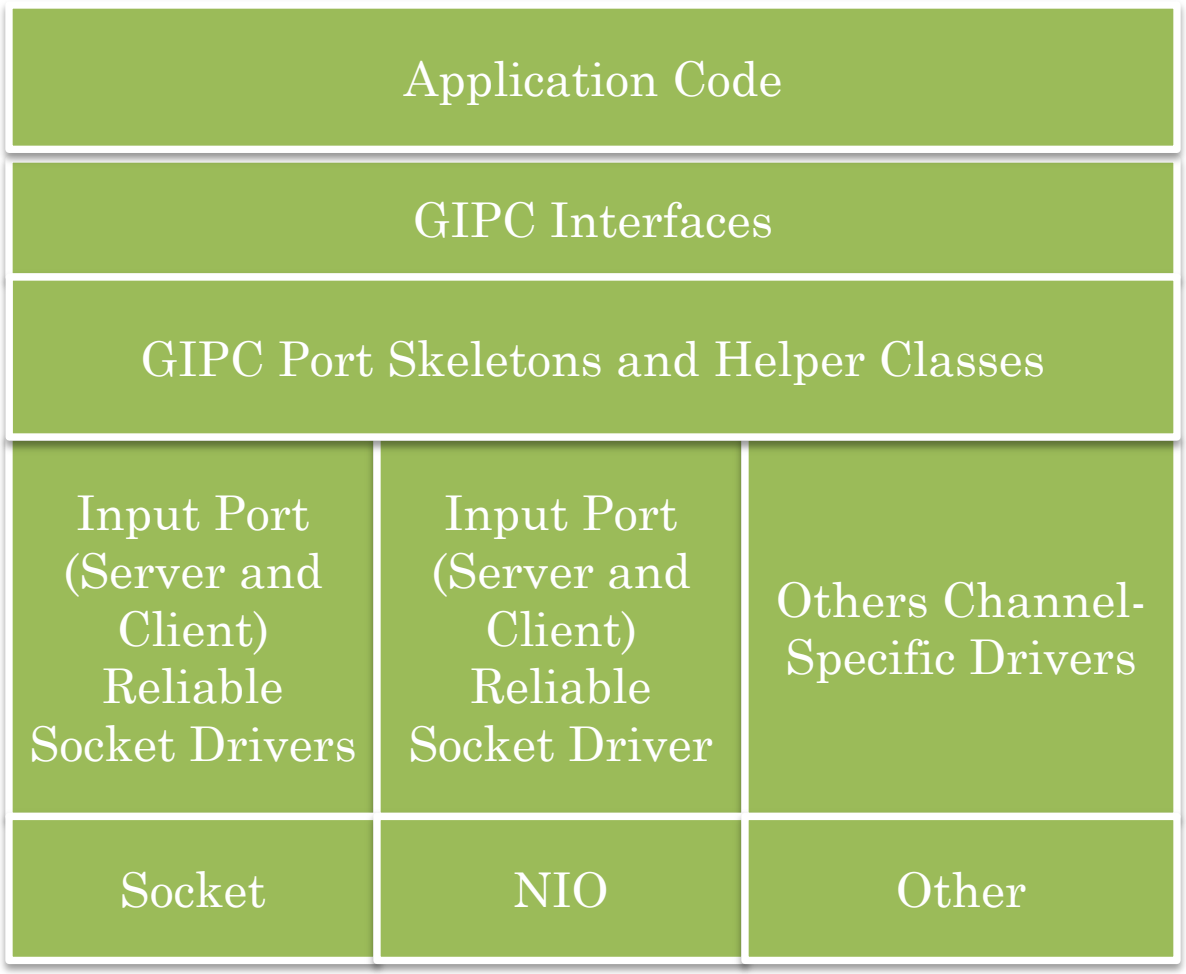
Plug in Models

Add Observer ✓

Replace existing layer ✓

Interpose new layer between two existing layers

Interposition Motivation



What if we want to add delays with messages?

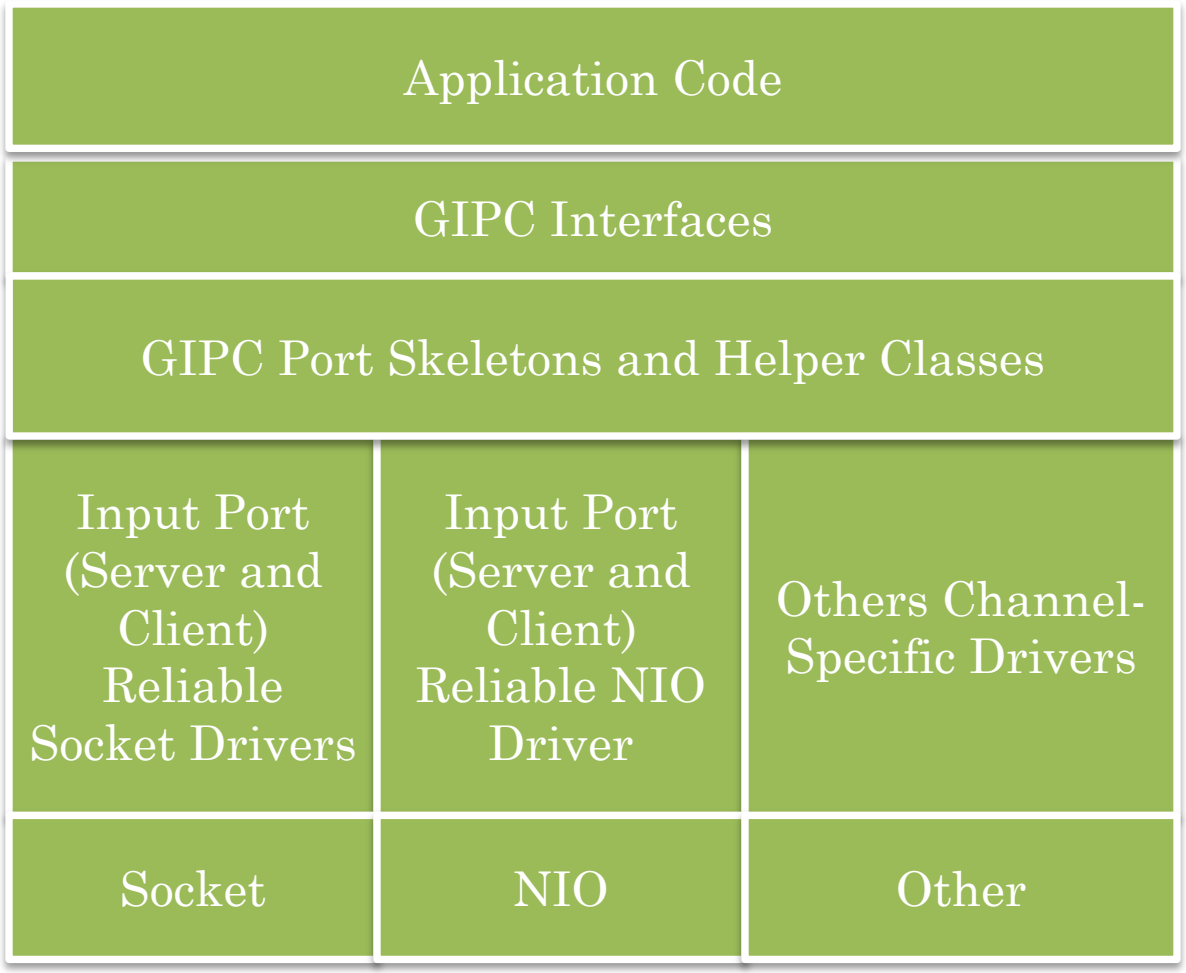
Do not want to duplicate code in different drivers

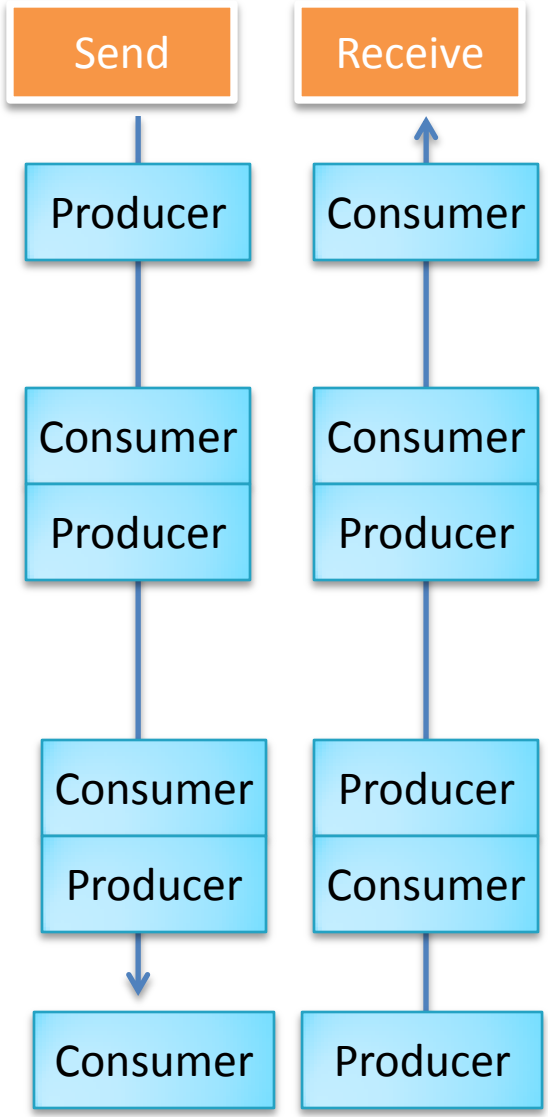
Can change or subclass skeletons

Need to worry about several unrelated concepts such as connection

Sometimes all we want to do is trap receives and sends

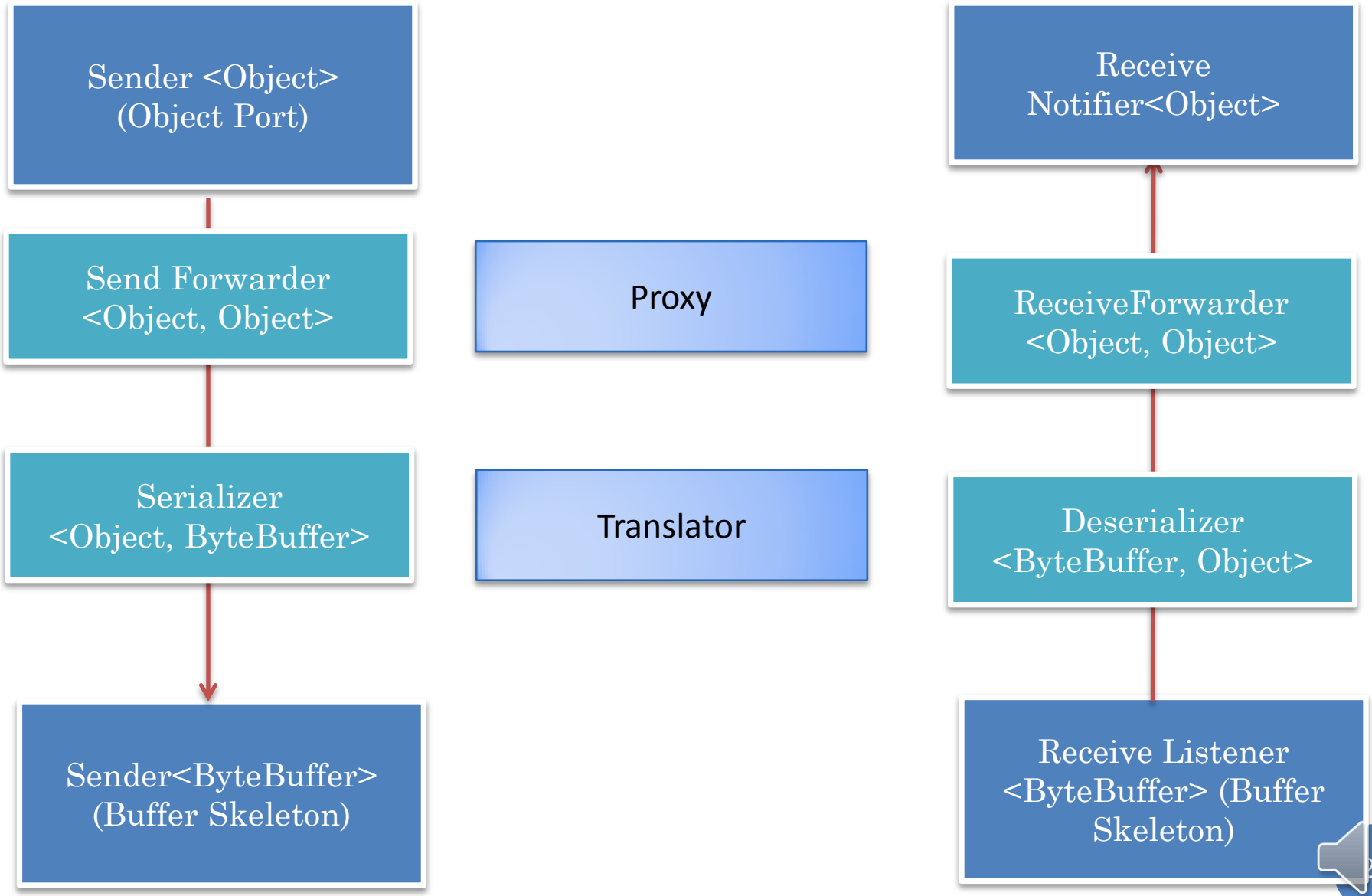
Pipe-Based Extensibility



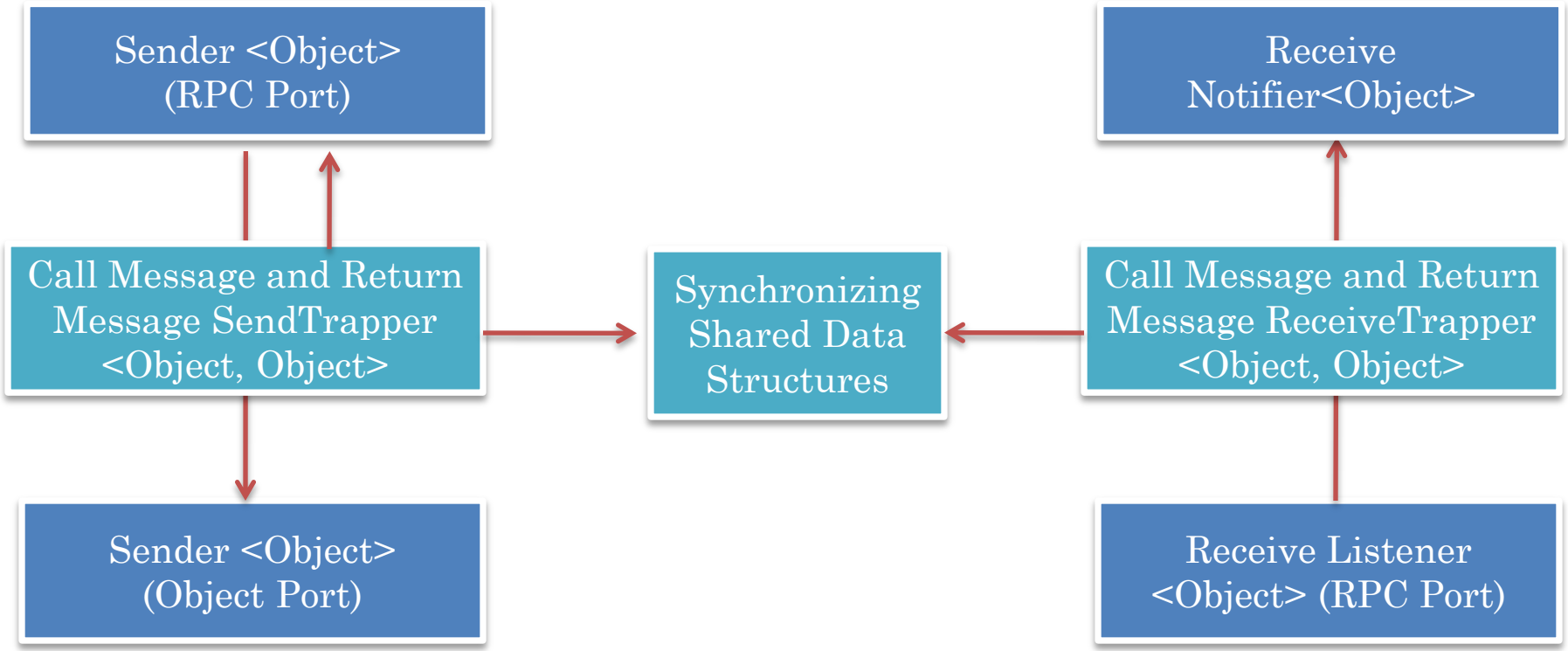


Can add arbitrary long chains of application message trappers between a producer/consumer pair

Proxies vs. Translators



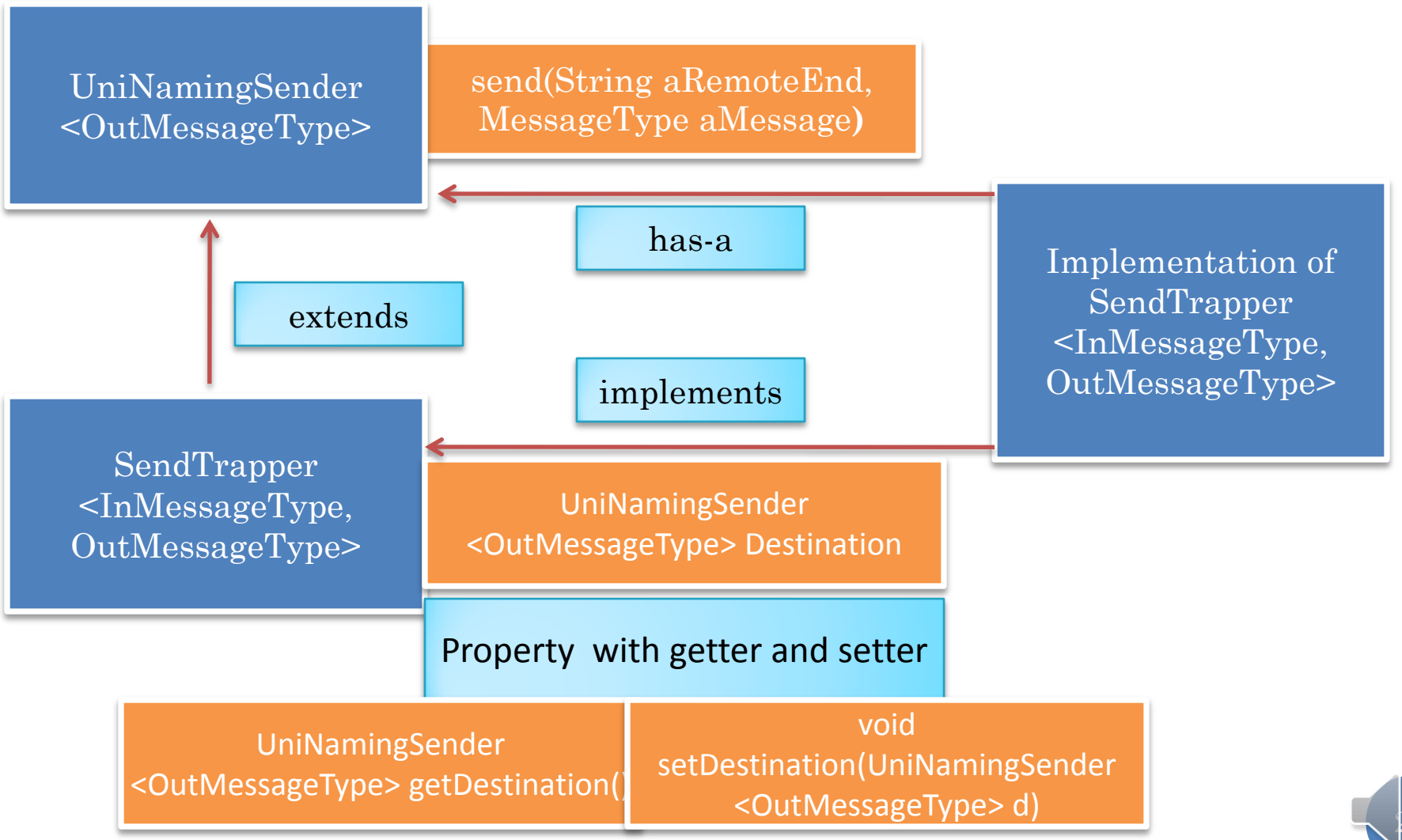
Shared Trappers and Synchronizations



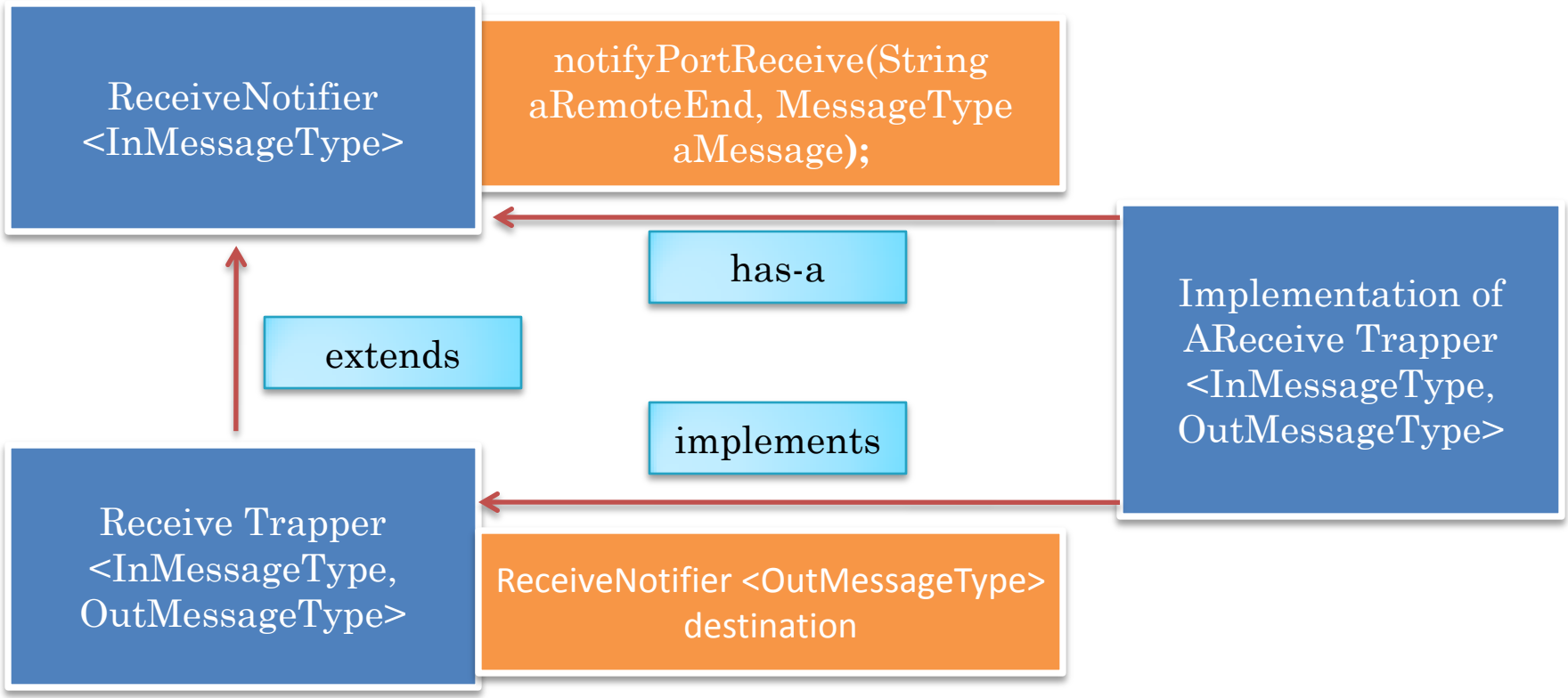
Send trapper may need to block sender to implement synchronous operation

Send and receive trapper may need to share information about remote references sent and received

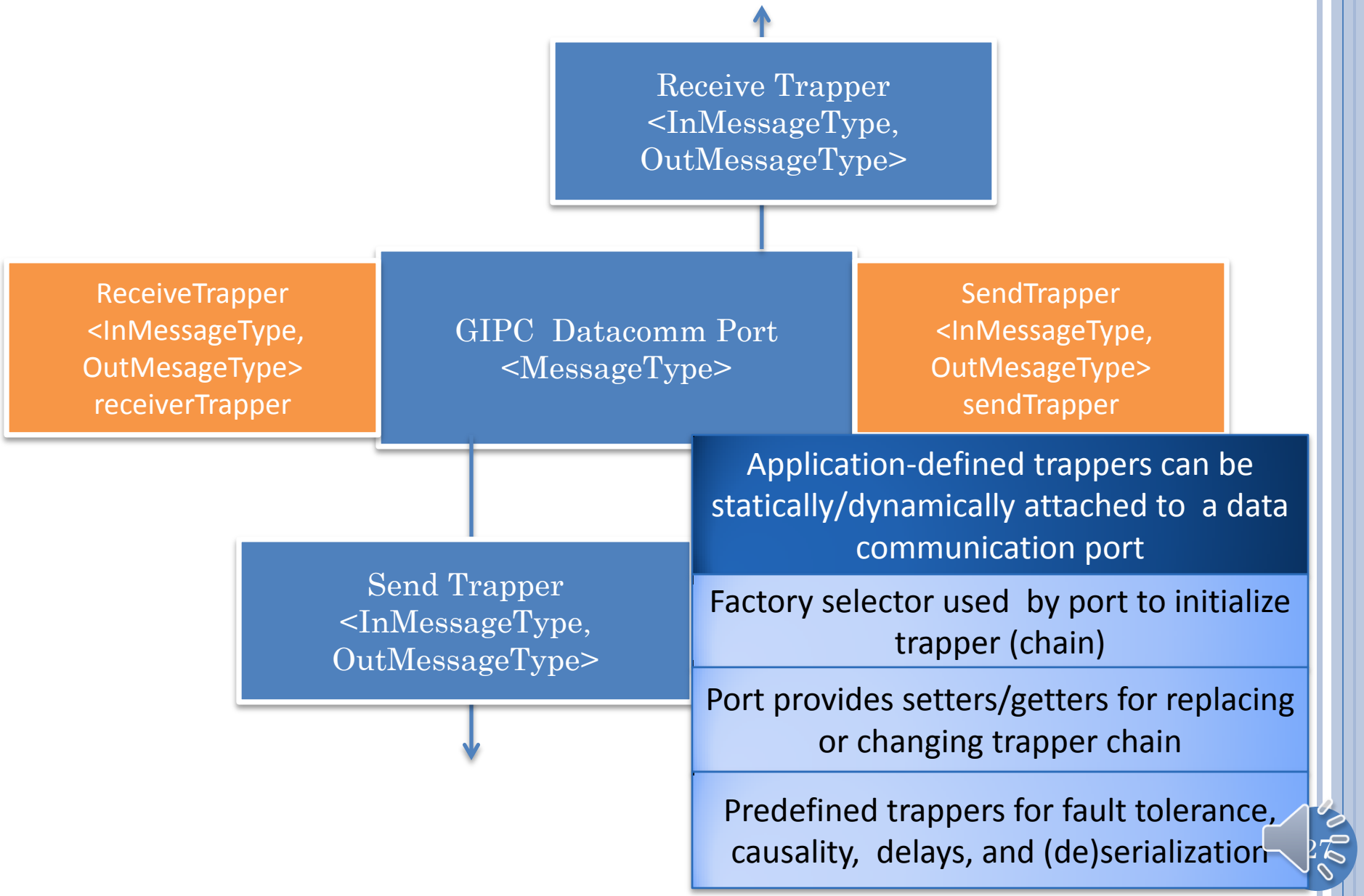
Send Trapper



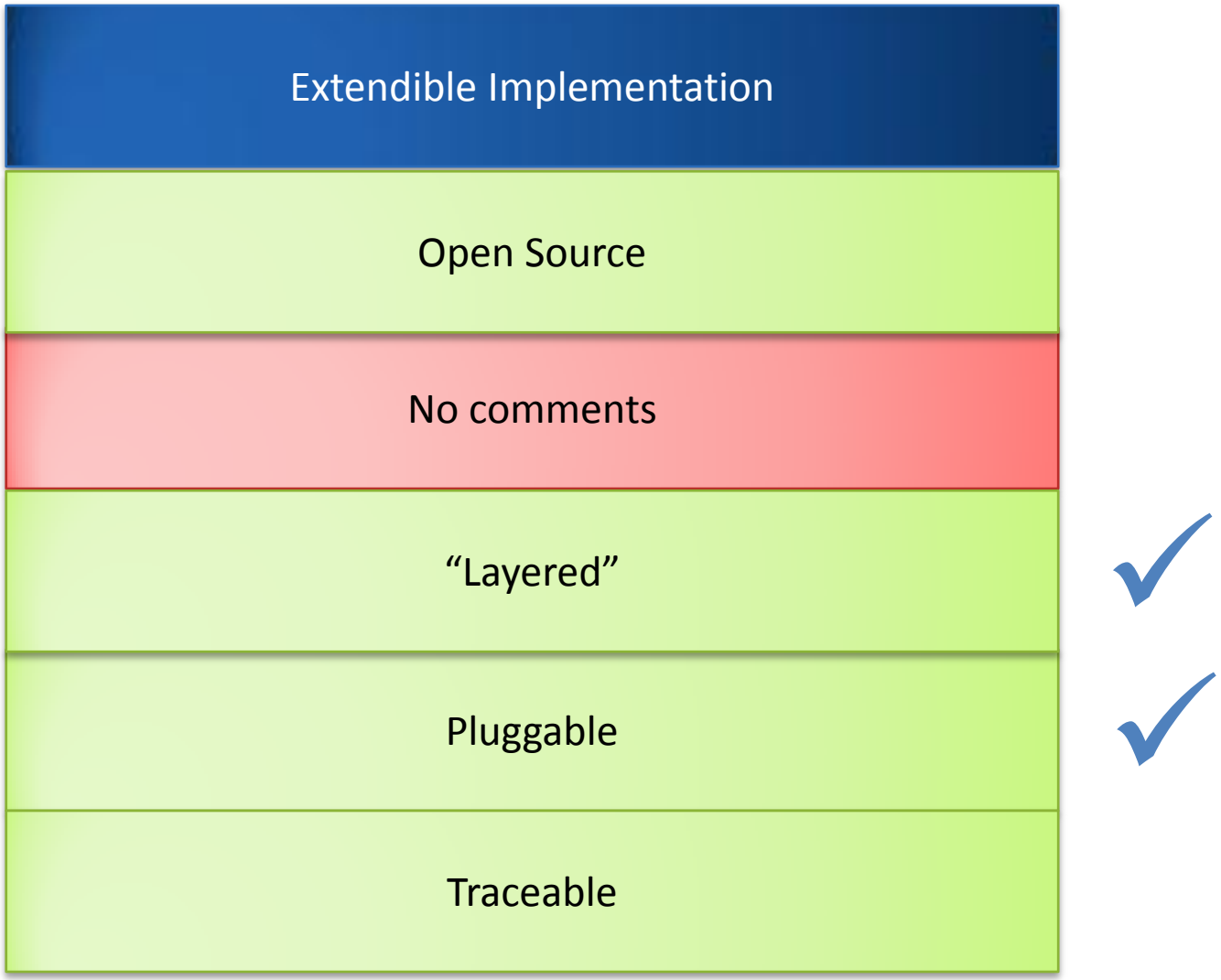
Receive Trapper



Attaching Trappers



Implementation Goal



Tracing with Debugger

Debugger makes it difficult to test race conditions

All threads and processes mapped to a single code window

Cannot see the history of actions taken by a thread

Break points do not transfer to another computer

Cannot use a mechanism to set multiple related debug points from a program

TRACE

```
Please enter message:
The woods are lovely
I***(inputport.datacomm.simplex.buffer)Forwarding message to send trapper:inputpor
I***(inputport.datacomm.simplex)Forwarding sent message java.nio.HeapByteBuffer[po
I***(inputport.datacomm.simplex.buffer.nio)Sending messsage: java.nio.HeapByteBuff
I***(inputport.datacomm.simplex.buffer.nio)ABufferedWrite with id:0 contents:java.
I***(inputport.datacomm.simplex.buffer.nio)Started storing of buffered write with
I***(inputport.da                                                te op for:java.nio.chann
I***(inputport.da                                                g of buffered write with
Please enter mes
I***(inputport.da                                                unblocks
I***(inputport.da                                                te op for:java.nio.chann
I***(inputport.da                                                select
I***(inputport.da                                                unblocks
I***(inputport.da                                                heljava.nio.channels.Soc
I***(inputport.da                                                ers about write
I***(inputport.da                                                heljava.nio.channels.Soc
I***(inputport.datacomm.simplex)notifying to send listners message:java.nio.HeapBy
Alice-->Echo Server:(0)java.nio.HeapByteBuffer[pos=0 lim=20 cap=20]
I***(inputport.datacomm.simplex.buffer.nio)notified listeners about write
I***(inputport.datacomm.simplex.buffer.nio)Selector registering read as no pending
I***(inputport.datacomm.simplex.buffer.nio)channel not connected or no pending wri
I***(inputport.datacomm.simplex.buffer.nio)Selector calls select
```

Tracer.*showInfo*(`true`);

May not want to to know about nio

Filtering: Selecting which events to print?

ONLY INPUTPORT.DATACOMM.SIMPLEX.BUFFER

```
AServerSimplexBufferInputPortLauncher [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (Oct 17, 2011 4:22:27
I***(inputport.datacomm.simplex.buffer)Retrieved from global state receive trapp
I***(inputport.datacomm.simplex.buffer)Changing connection status and asking dri
I***(inputport.datacomm.simplex.buffer)Received message:java.nio.HeapByteBuffer[
I***(inputport.datacomm.simplex.buffer)Associating Alice with java.nio.channels.
I***(inputport.datacomm.simplex.buffer)ServerInputPort connected to:java.nio.cha
Echo Server<-->Alice (Opened)
I***(inputport.datacomm.simplex.buffer)Received message:java.nio.HeapByteBuffer[
I***(inputport.datacomm.simplex.buffer)ServerInputPort received message java.nio
```

```
AnAliceSimplexBufferInputPortLauncher [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (Oct 17, 2011 4:22:33
I***(inputport.datacomm.simplex.buffer)Retrieved from global state send trapper:
I***(inputport.datacomm.simplex.buffer)Set my send trapper to:inputport.datacomm
I***(inputport.datacomm.simplex.buffer)Adding send listener:inputport.datacomm.s
I***(inputport.datacomm.simplex.buffer)Asking driver to connect and changing sta
I***(inputport.datacomm.simplex.buffer)Received connected notificaton from drive
I***(inputport.datacomm.simplex.buffer)Sending to server my name: Alice
I***(inputport.datacomm.simplex.buffer)Forwarding message to send trapper:inputp
Alice<-->Echo Server (Opened)
I***(inputport.datacomm.simplex.buffer)Received sent notification from driver
I***(inputport.datacomm.simplex.buffer)Notifying to send listeners message:java.
Alice-->Echo Server:(0)java.nio.HeapByteBuffer[pos=0 lim=5 cap=5]
Please enter message:
The woods are lovely
I***(inputport.datacomm.simplex.buffer)Forwarding message to send trapper:inputp
Please enter message:
I***(inputport.datacomm.simplex.buffer)Received sent notification from driver
I***(inputport.datacomm.simplex.buffer)Notifying to send listeners message:java.
```

Explicit Keywords

```
public static void info(String keyWord, String info);
```

```
Tracer.info("inputport.datacomm.simplex.buffer", "Asking
driver to connect and changing status");
```

```
public static void setKeywordPrintStatus(
    String keyWord,
    Boolean status);
```

```
Tracer.setKeywordPrintStatus("inputport.datacomm.simplex.
buffer", true)
```

```
Tracer.setKeywordPrintStatus(Tracer.ALL_KEYWORDS, false);
```

Have to specify package name each time

Package name can change

Implicit KeyWords

```
public static void info(Object object, String info);
```

```
Tracer.info(this, "Asking driver to connect and changing status");
```

```
public static void setKeywordPrintStatus(Class c, Boolean status);
```

```
Tracer.setKeywordPrintStatus(
            AGenericSimplexBufferClientInputPort.class,
            false);
```

Showing Package Names

AServerSimplexBufferInputPortLauncher [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (Oct 17, 2011 4:22:27

Cannot identify classes of the objects printing out messages

Need to not only specify which events to display but what information to display about each event

AnAliceSimplexBufferInputPortLauncher [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (Oct 17, 2011 4:22:33

```
I***(inputport.datacomm.simplex.buffer)Retrieved from global state send trapper:
```

```
Tracer.showInfo(true);
Tracer.setKeywordPrintStatus(Tracer.ALL_KEYWORDS, false);
Tracer.setKeywordPrintStatus(
     AGenericSimplexBufferClientInputPort.class, true);
Tracer.setMessagePrefixKind(MessagePrefixKind.PACKAGE_NAME);
```

```
I***(inputport.datacomm.simplex.buffer)Notifying to send listeners message:java.
Alice-->Echo Server:(0)java.nio.HeapByteBuffer[pos=0 lim=5 cap=5]
Please enter message:
The woods are lovely
I***(inputport.datacomm.simplex.buffer)Forwarding message to send trapper:inputp
Please enter message:
I***(inputport.datacomm.simplex.buffer)Received sent notification from driver
I***(inputport.datacomm.simplex.buffer)Notifying to send listeners message:java.
```

Showing Short Class Names

```
AServerSimplexBufferInputPortLauncher [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (Oct 17, 2011 12:52:32 PM)
I***(AGenericSimplexBufferServerInputPort)Retrieved from global state receive trapper:inputport.da
I***(AGenericSimplexBufferServerInputPort)Changing connection status and asking driver to connect
I***(AGenericSimplexBufferServerInputPort)Received message:java.nio.HeapByteBuffer[pos=4 lim=9 cap
I***(AGenericSimplexBufferServerInputPort)Associating Alice with java.nio.channels.SocketChannel[
I***(AGenericSimplexBufferServerInputPort)ServerInputPort connected to:java.nio.channels.SocketCha
Echo Server<-->Alice (Opened)
I***(AGenericSimplexBufferServerInputPort)Received message:java.nio.HeapByteBuffer[pos=4 lim=24 ca
I***(AGenericSimplexBufferServerInputPort)ServerInputPort received message java.nio.HeapByteBuffe
Echo Server<--Alice:The woods are lovely
```

```
Tracer.showInfo(true);
Tracer.setKeywordPrintStatus(Tracer.ALL_KEYWORDS, false);
Tracer.setKeywordPrintStatus(
    AGenericSimplexBufferClientInputPort.class, true);
Tracer.setMessagePrefixKind(MessagePrefixKind.SHORT_CLASS_NAME)
```

```
I***(AGenericSimplexBufferClientInputPort)Received sent notification from driver
I***(ASendRegistrarAndNotifier)Notifying to send listeners message:java.nio.HeapByte
Alice-->Echo Server:(0)java.nio.HeapByteBuffer[pos=0 lim=5 cap=5]
Please enter message:
The woods are lovely
I***(AGenericSimplexBufferClientInputPort)Forwarding message to send trapper:inputpo
Please enter message:
I***(AGenericSimplexBufferClientInputPort)Received sent notification from driver
I***(ASendRegistrarAndNotifier)Notifying to send listeners message:java.nio.HeapByte
Alice-->Echo Server:(1)java.nio.HeapByteBuffer[pos=0 lim=20 cap=20]
```

MessagePrefix

```
public static void setMessagePrefixKind(
                        MessagePrefixKind newValue)
```

```
public enum MessagePrefixKind {
   NONE,
   OBJECT_TO_STRING,
   SHORT_CLASS_NAME,
   FULL_CLASS_NAME,
   PACKAGE_NAME
}
```

Displaying All Classes in Package

```
AServerSimplexBufferInputPortLauncher [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (Oct 17, 2011 12:52:32 PM)
I***(AGenericSimplexBufferServerInputPort)Retrieved from global state receive trapper:inputport.d
I***(AGenericSimplexBufferServerInputPort)Changing connection status and asking driver to connect
I***(AGenericSimplexBufferServerInputPort)Received message:java.nio.HeapByteBuffer[pos=4 lim=9 ca
I***(AGenericSimplexBufferServerInputPort)Associating Alice with java.nio.channels.SocketChannel[
I***(AGenericSimplexBufferServerInputPort)ServerInputPort connected to:java.nio.channels.SocketCh
Echo Server<-->Alice (Opened)
```

```
Tracer.showInfo(true);
Tracer.setKeywordPrintStatus(Tracer.ALL_KEYWORDS, false);
Tracer.setKeywordPrintStatus(
    AGenericSimplexBufferClientInputPort.class, true);
Tracer.setMessagePrefixKind(MessagePrefixKind.SHORT_CLASS_NAME)
```

```
I***(AGenericSimplexBufferClientInputPort)Sending to server my name: Alice
I***(AGenericSimplexBufferClientInputPort)Forwarding message to send trapper:inputpo
Alice<-->Echo Server (Opened)
I***(AGenericSimplexBufferClientInputPort)Received sent notification from driver
I***(ASendRegistrarAndNotifier)Notifying to send listeners message:java.nio.HeapByte
Alice-->Echo Server:(0)java.nio.HeapByteBuffer[pos=0 lim=5 cap=5]
Please enter message:
The woods are lo
I***(AGenericSim                                                    d trapper:inputpo
Please enter mes
I***(AGenericSim                                                    n from driver
I***(ASendRegist                                                    java.nio.HeapByte
Alice-->Echo Ser
```

What if we want to focus on one class?

Narrow down which events

Controlling Implicit Keyword

```
AServerSimplexBufferInputPortLauncher (1) [Java Application] D:\Program Files\Java\jre1.6.0_04\bin\javaw.exe
Echo Server<-->Alice (Opened)
Echo Server<--Alice:The woods are lovely
```

```
I***(AGenericSimplexBufferClientInputPort)Retrieved from global state send trapper:inputport.datacomm.s
I***(AGenericSimplexBufferClientInputPort)Set my send trapper to:inputport.datacomm.simplex.ASendMessag
I***(AGenericSimplexBufferClientInputPort)Asking driver to connect and changing status
```

```
Tracer.showInfo(true);
Tracer.setKeywordPrintStatus(Tracer.ALL_KEYWORDS, false);
Tracer.setImplicitKeywordKind(ImplicitKeywordKind.OBJECT_CLASS_NAME);
Tracer.setKeywordPrintStatus(
    AGenericSimplexBufferClientInputPort.class, true);
Tracer.setMessagePrefixKind(MessagePrefixKind.SHORT_CLASS_NAME);
```

Implicit Keyword

```
public static void setImplicitKeywordKind(
        ImplicitKeywordKind newValue)
```

```
public enum ImplicitKeywordKind {
   OBJECT_TO_STRING,
   OBJECT_CLASS_NAME,
   OBJECT_PACKAGE_NAME
}
```

Tracer Static Methods Summary

```
setKeywordPrintStatus(String keyWord Boolean status);
```

```
info(String keyWord, String info);
```

```
setImplicitKeywordKind(ImplicitKeywordKind newValue)
```

```
info(Object obj, String info);
```

```
setKeywordPrintStatus(Class cls, Boolean st
```

```
public enum ImplicitKeywordKind {
   OBJECT_TO_STRING,
   OBJECT_CLASS_NAME,
   OBJECT_PACKAGE_NAME
}
```

```
setMessagePrefixKind(MessagePrefixKind n
```

```
public enum MessagePrefixKind {
   NONE,
   OBJECT_TO_STRING,
   SHORT_CLASS_NAME,
   FULL_CLASS_NAME,
   PACKAGE_NAME
}
```

Debugging Capabilities in Tracer?

Blocking but separate windows for different processes

See state at traced actions (with and without blocking)

Separate state for different threads (with and without blocking)

Have application-specific code learn about traced calls (perhaps in different processes)

Tracing Strings→Objects

```
setKeywordPrintStatus(String keyWord Boolean status);
```

```
info(String keyWord, String info);
```

```
setImplicitKeywordKind(ImplicitKeywordKind newValue)
```

```
info(Object obj, String info);
```

```
setKeywordPrintStatus(Class cls, Boolean st
```

```
public enum ImplicitKeywordKind {
   OBJECT_TO_STRING,
   OBJECT_CLASS_NAME,
   OBJECT_PACKAGE_NAME
}
```

```
setMessagePrefixKind(MessagePrefixKind n
```

```
public enum MessagePrefixKind {
   NONE,
   OBJECT_TO_STRING,
   SHORT_CLASS_NAME,
   FULL_CLASS_NAME,
   PACKAGE_NAME
}
```

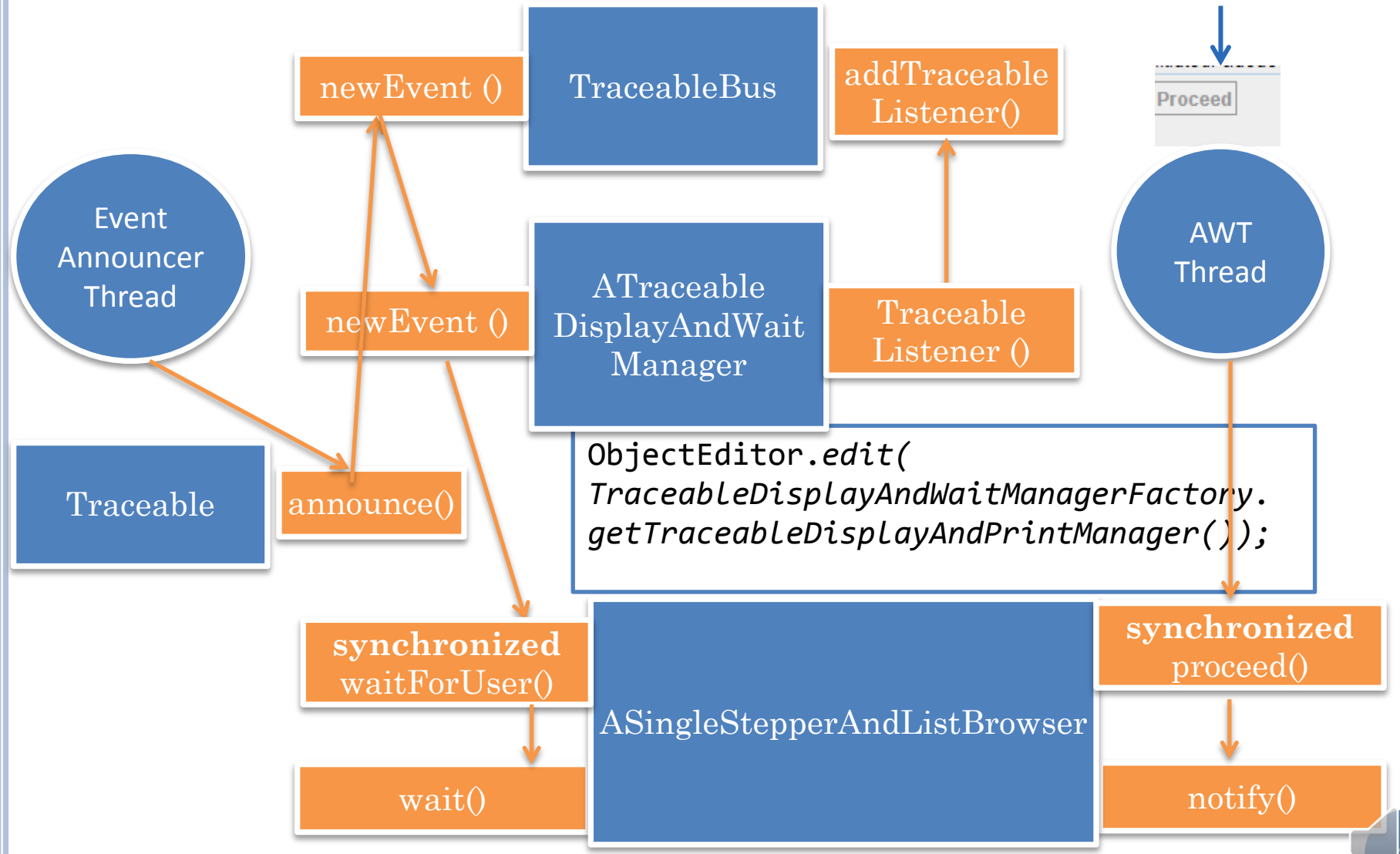
STRING→OBJECT

```
@DisplayToString(true)
@ComponentWidth(1000)
public class MVCTraceableInfo extends TraceableInfo{
  public MVCTraceableInfo(String aMessage, Object anAnnouncer) {
    super(aMessage, anAnnouncer);
 }
  public static MVCTraceableInfo newInfo(String aMessage, Object aFinder) {
    MVCTraceableInfo retVal = new MVCTraceableInfo(aMessage, aFinder);
    retVal.announce();
    return retVal;
  }
}
```

```
Tracer.info(this, "MVC structure built")
```

```
MVCTraceableInfo( "MVC structure built", this);
```

Tracing Objects



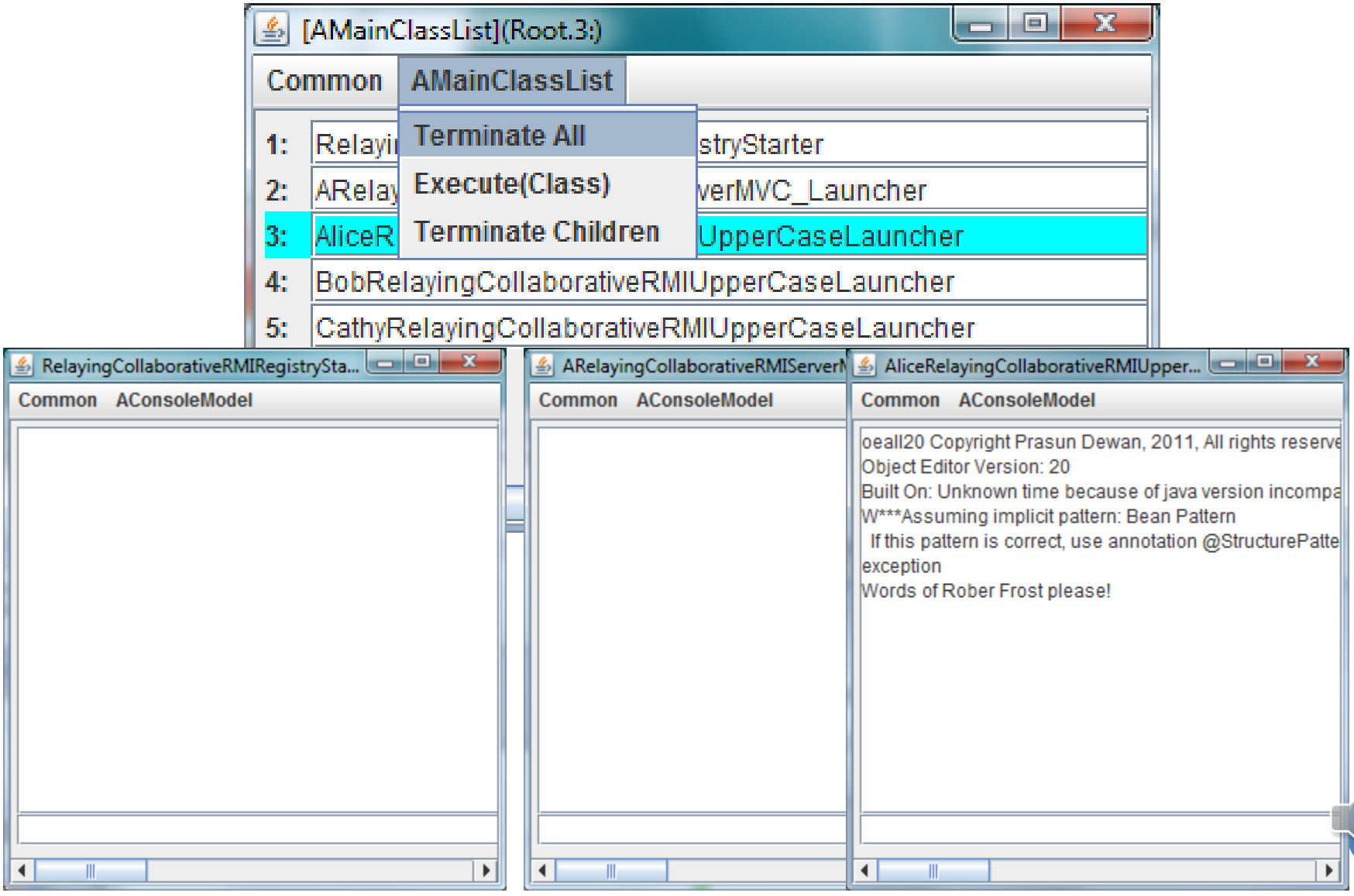
Message Bus

Like an observable it has registration method

Has an announce method

Does not generate events – simply communicates them to observers

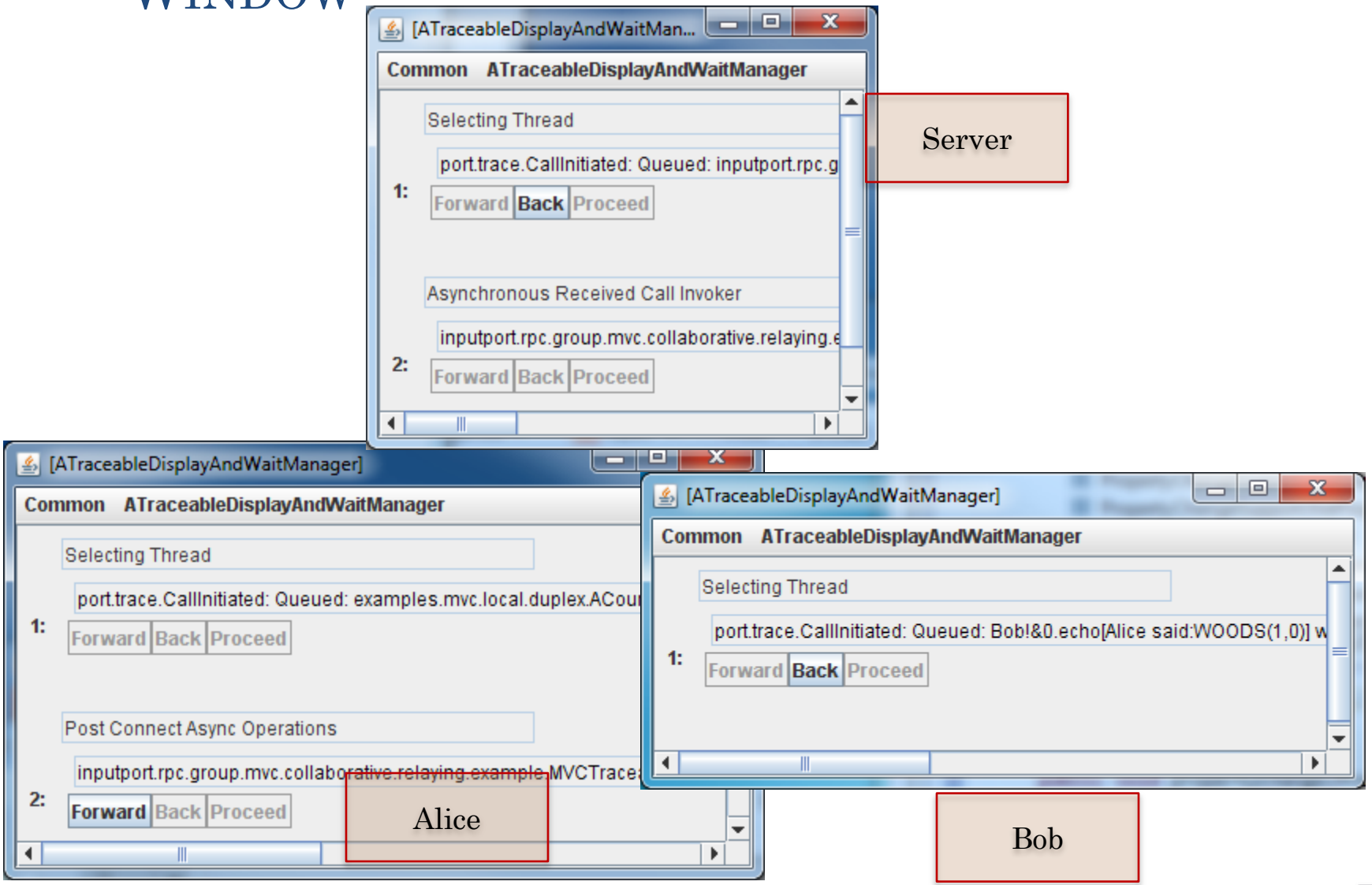
Run Functionality



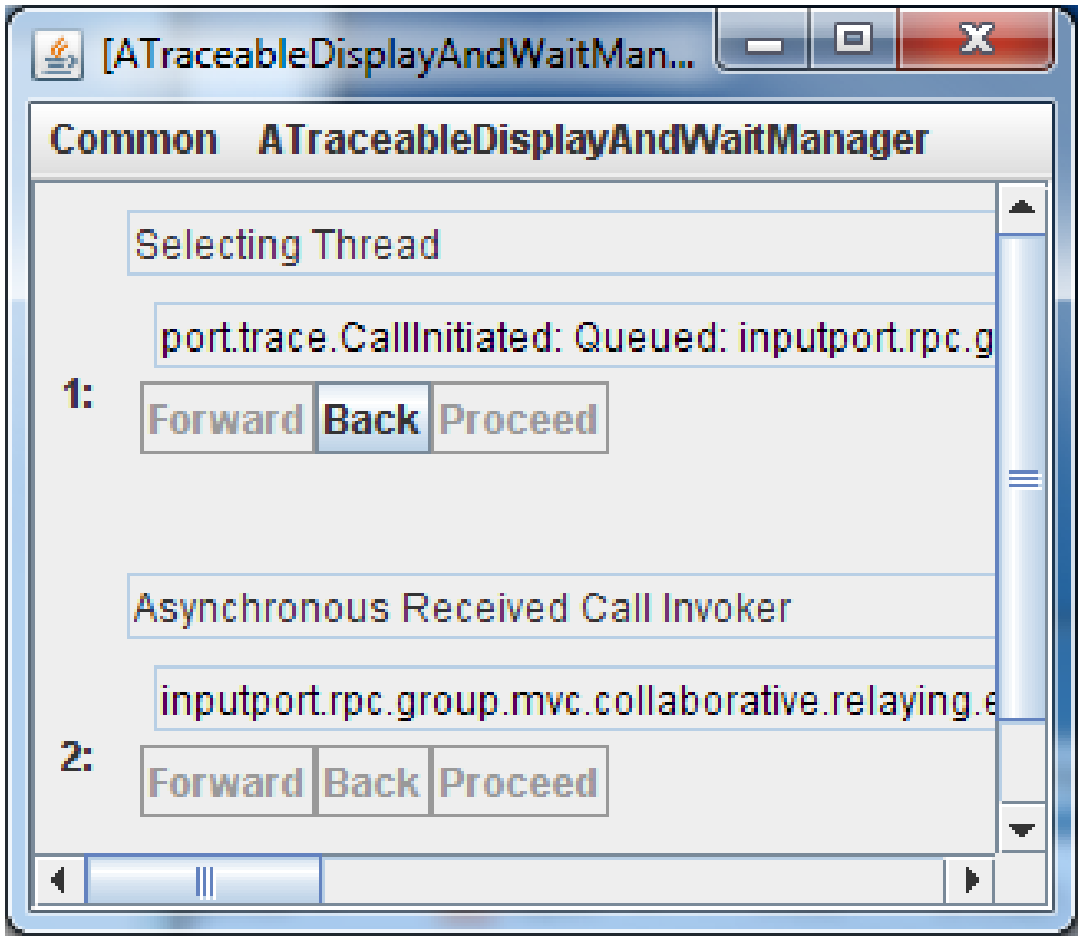
Debug Functionality

```
public ADuplexRPCClientRelayingCollaborativeMVCLauncher(
        String aClientName, String aServerHost,
        String aServerId, String aServerName) {
  super(aClientName, aServerHost, aServerId, aServerName);
  ObjectEditor.edit
      (TraceableDisplayAndWaitManagerFactory.
          getTraceableDisplayAndPrintManager());
  Tracer.setKeywordDisplayStatus(this, true);
}
```

Each Process Can Have Separate Trace Window



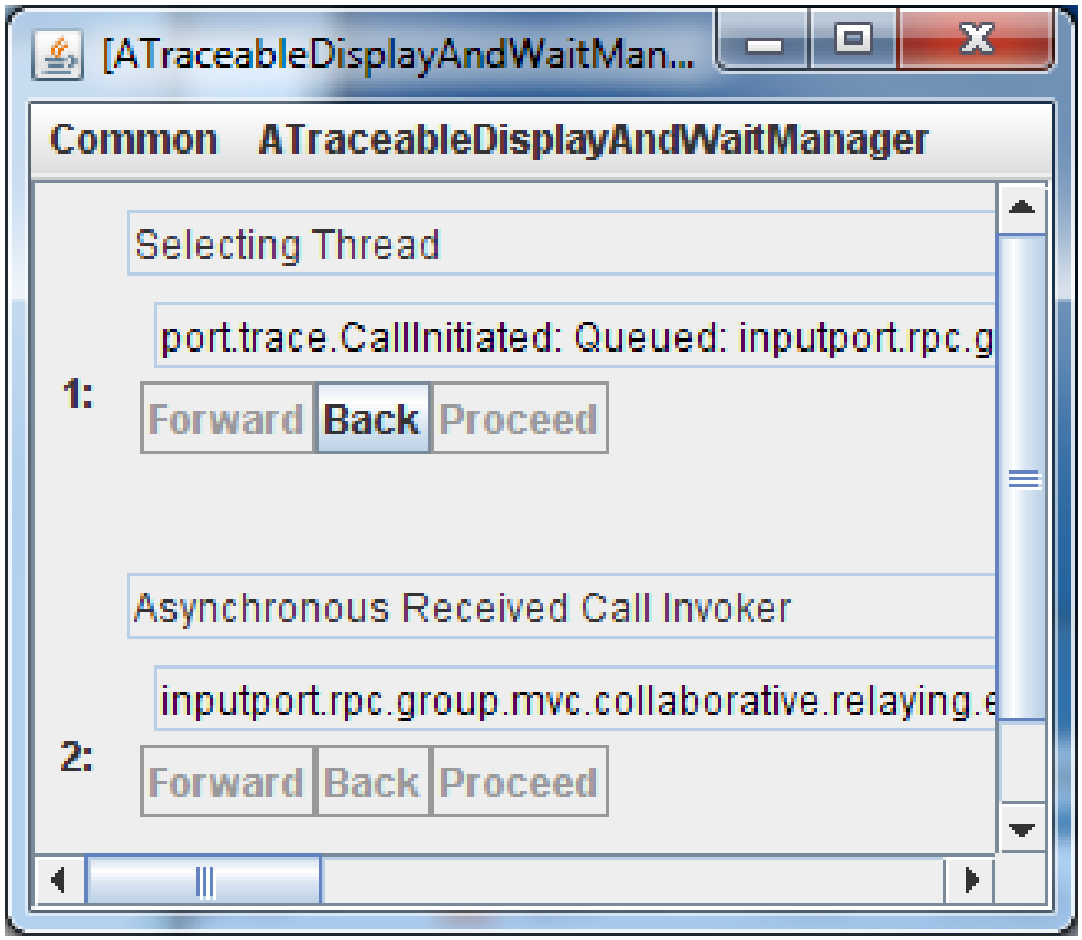
Each Trace Window has Separate Thread Area



Thread interacting with underlying communication channel

Thread invoking remote calls

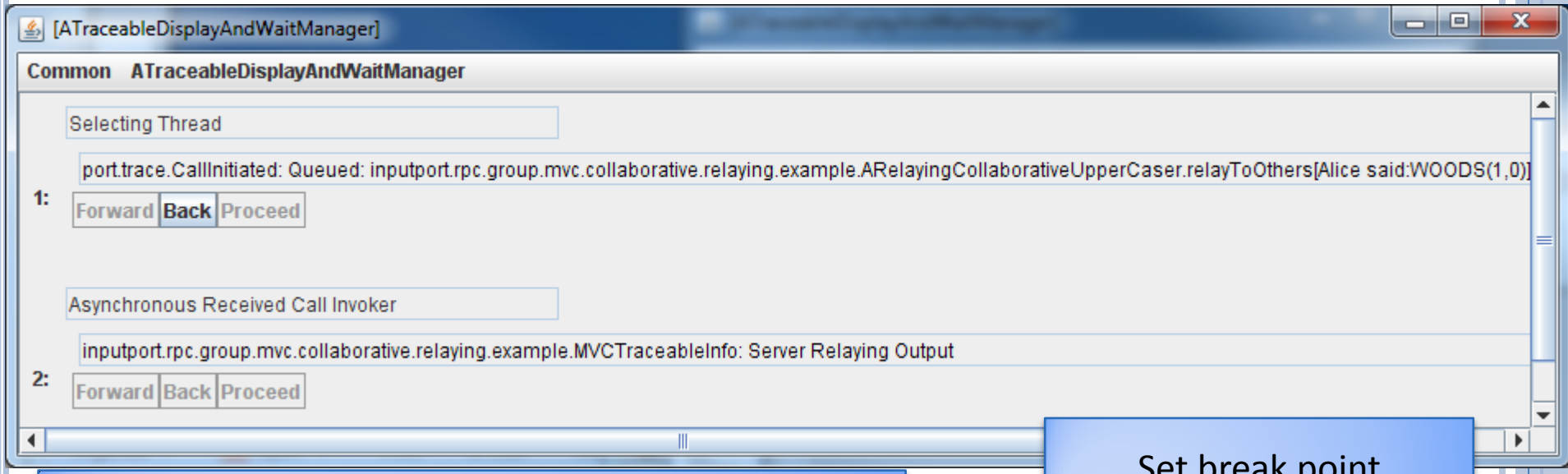
Thread display



Thread name

Other info

Goals



Debugger allows programmer to set breakpoint, resume, and at each breakpoint, pause and see stack trace, variable values, and console message when some line of code is executed

Tracer should provide equivalent

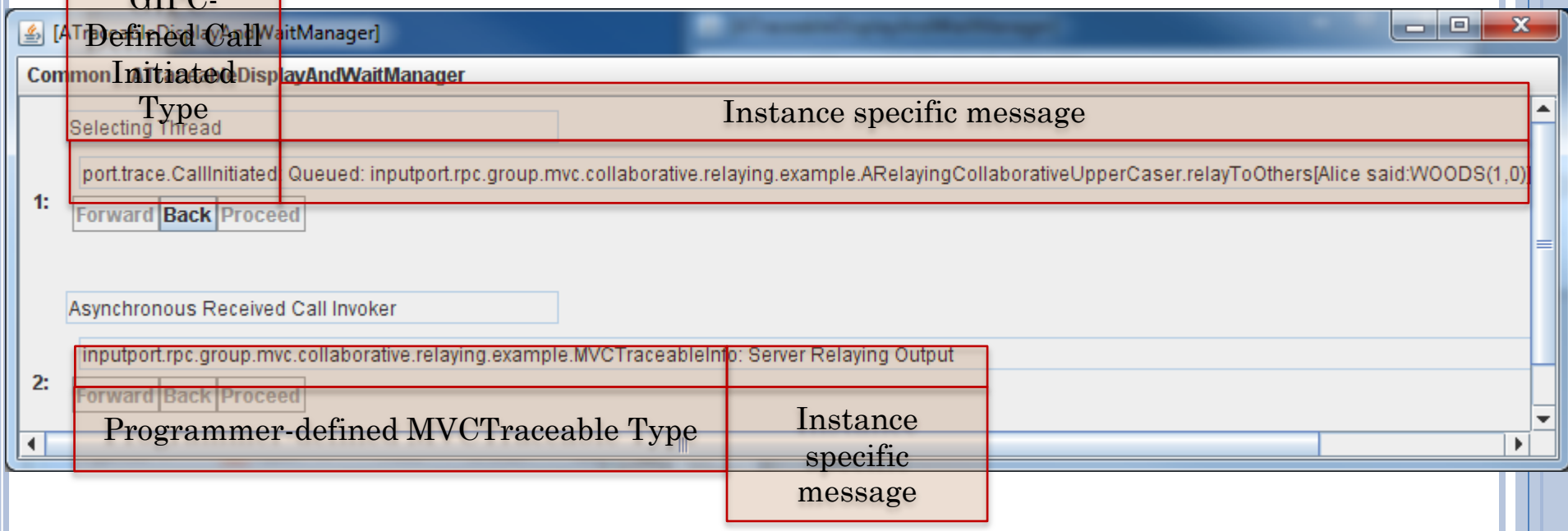
Set break point

BP Resume

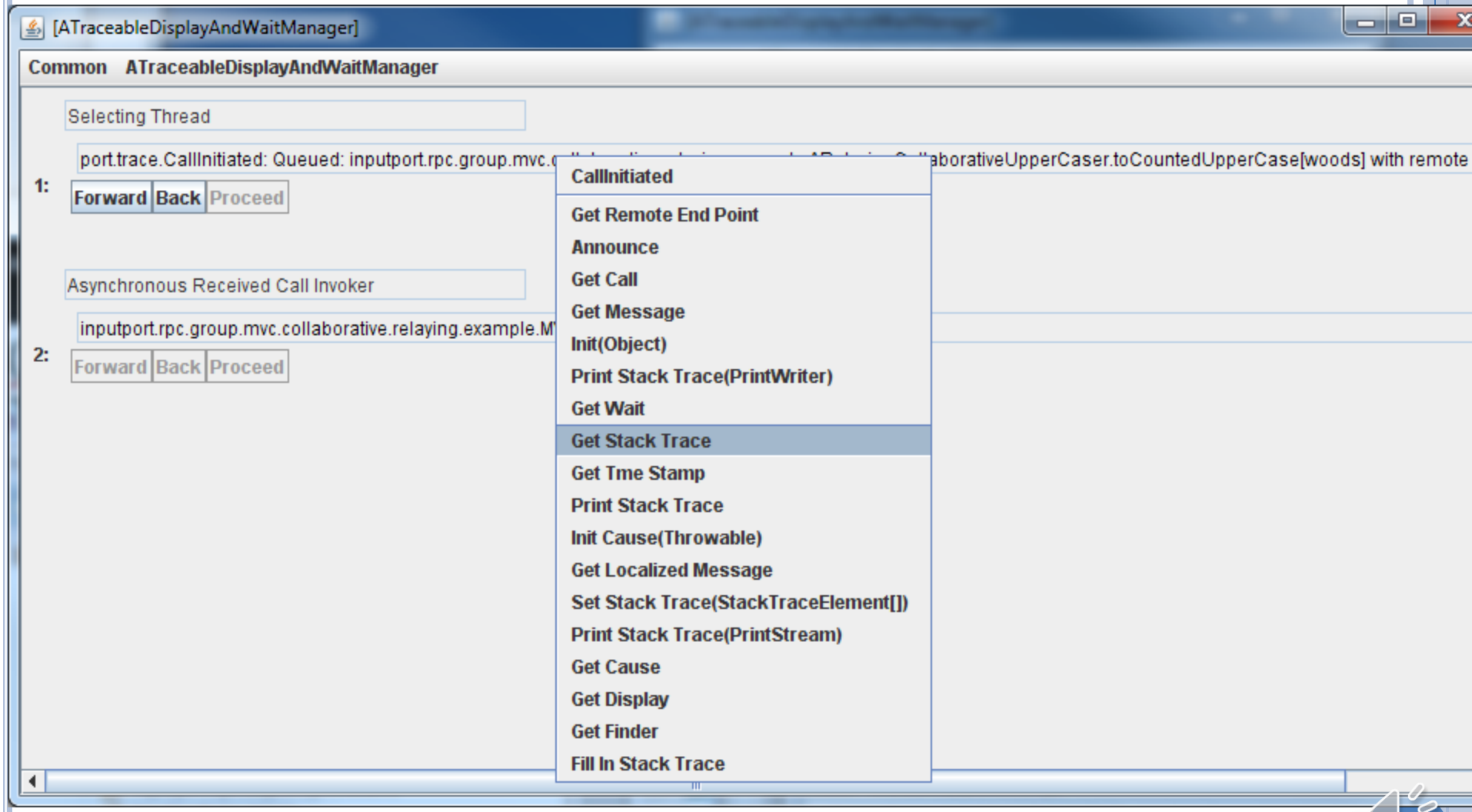
BP Console message

BP Variable Values

Console output Equivalent: Typed Tracing Message

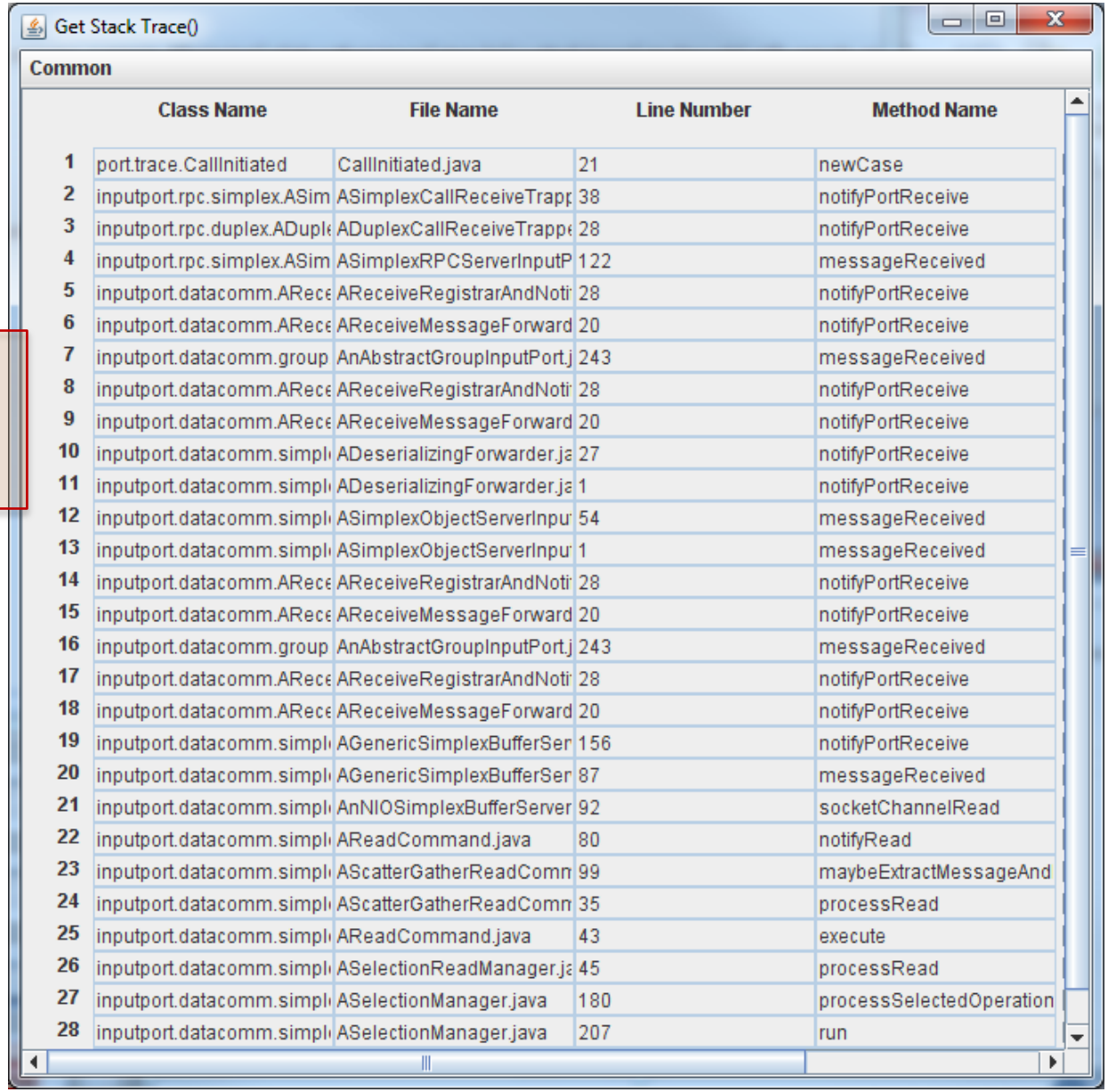


Stack Trace Equivalent: Stack Trace



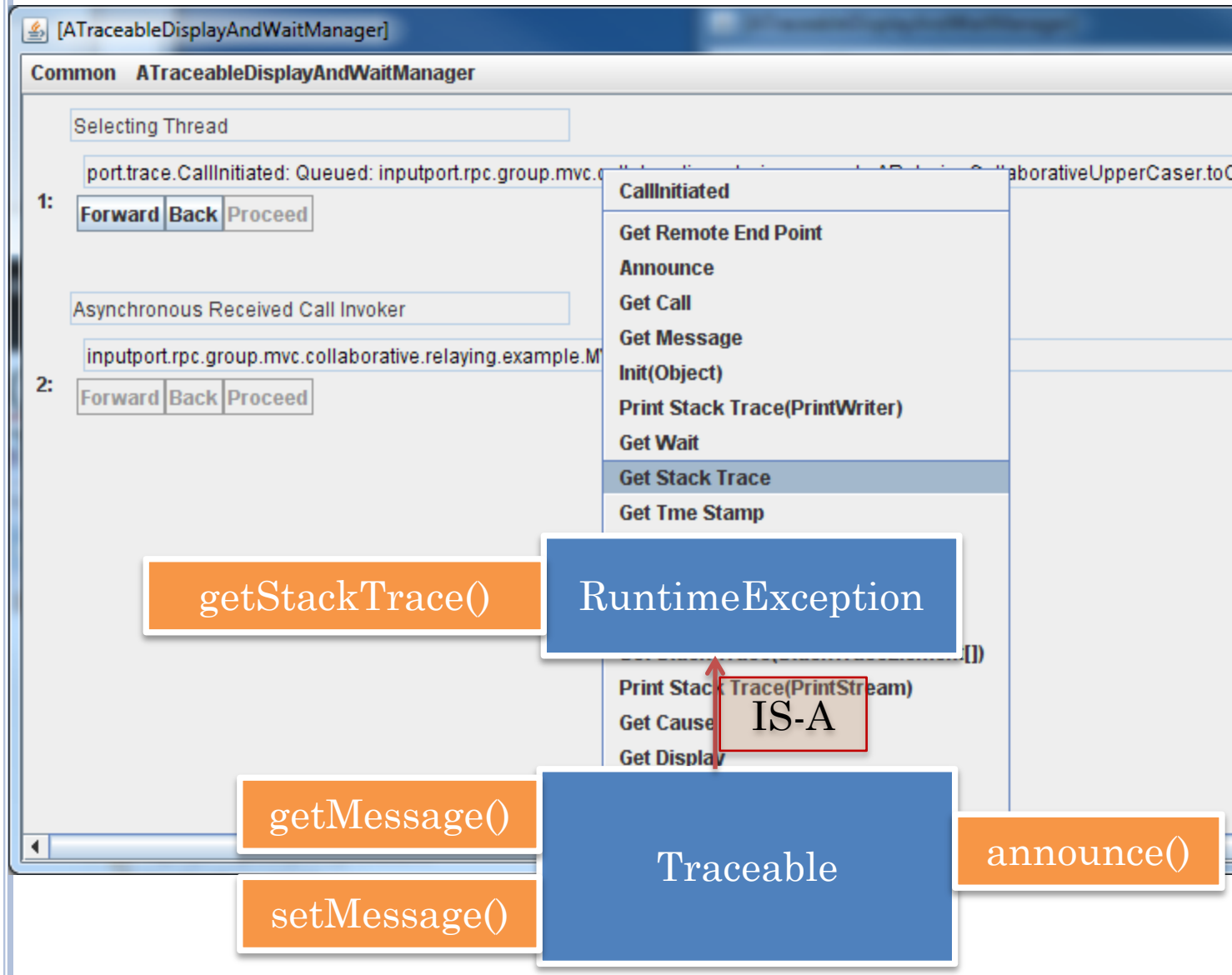
Stack Trace Display

Display of StackTrace Object

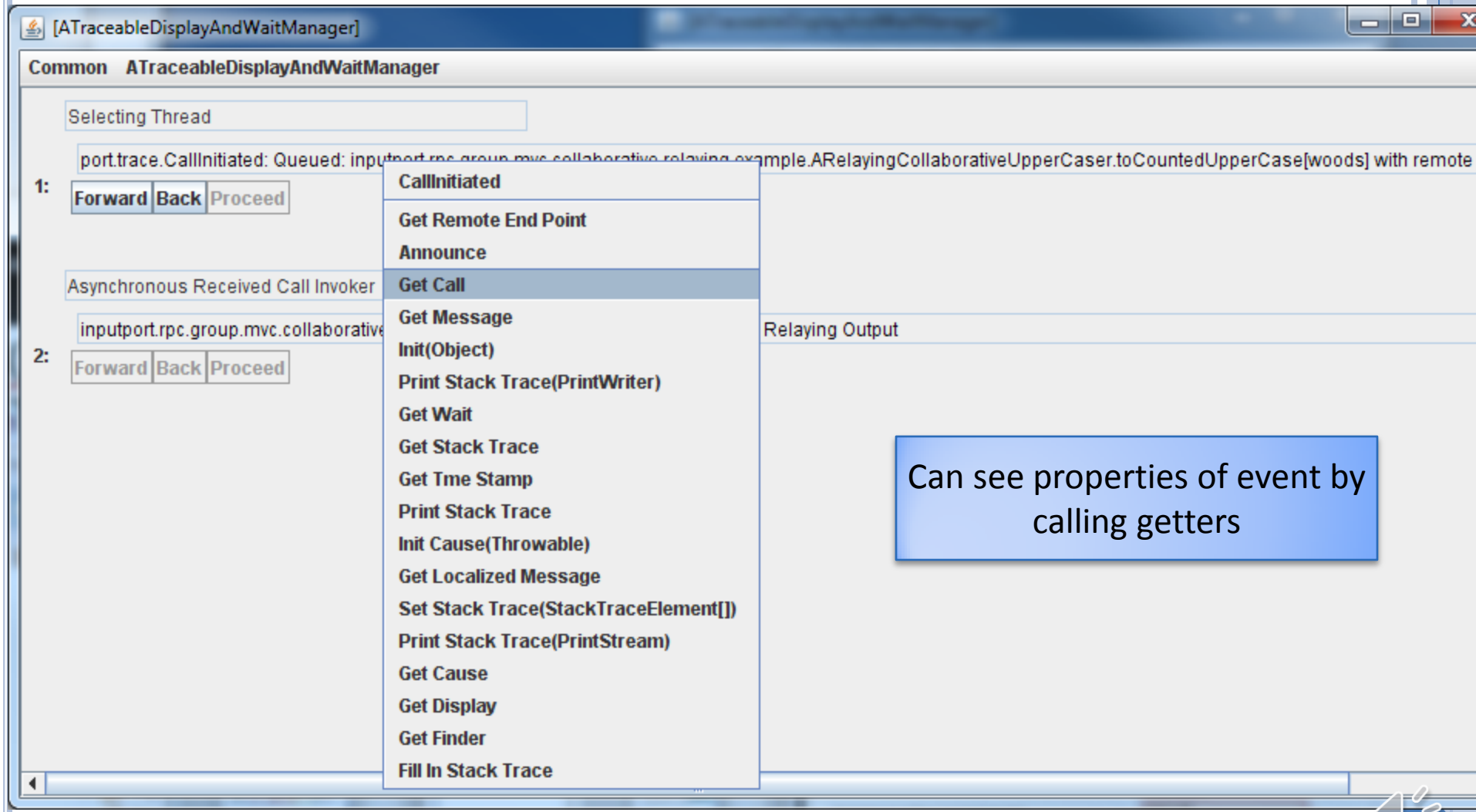


	Class Name	File Name	Line Number	Method Name
1	port.trace.CallInitiated	CallInitiated.java	21	newCase
2	inputport.rpc.simplex.ASim	ASimplexCallReceiveTrapp	38	notifyPortReceive
3	inputport.rpc.duplex.ADuple	ADuplexCallReceiveTrappe	28	notifyPortReceive
4	inputport.rpc.simplex.ASim	ASimplexRPCServerInputP	122	messageReceived
5	inputport.datacomm.ARece	AReceiveRegistrarAndNoti	28	notifyPortReceive
6	inputport.datacomm.ARece	AReceiveMessageForward	20	notifyPortReceive
7	inputport.datacomm.group	AnAbstractGroupInputPort.j	243	messageReceived
8	inputport.datacomm.ARece	AReceiveRegistrarAndNoti	28	notifyPortReceive
9	inputport.datacomm.ARece	AReceiveMessageForward	20	notifyPortReceive
10	inputport.datacomm.simpl	ADeserializingForwarder.ja	27	notifyPortReceive
11	inputport.datacomm.simpl	ADeserializingForwarder.ja	1	notifyPortReceive
12	inputport.datacomm.simpl	ASimplexObjectServerInpu	54	messageReceived
13	inputport.datacomm.simpl	ASimplexObjectServerInpu	1	messageReceived
14	inputport.datacomm.ARece	AReceiveRegistrarAndNoti	28	notifyPortReceive
15	inputport.datacomm.ARece	AReceiveMessageForward	20	notifyPortReceive
16	inputport.datacomm.group	AnAbstractGroupInputPort.j	243	messageReceived
17	inputport.datacomm.ARece	AReceiveRegistrarAndNoti	28	notifyPortReceive
18	inputport.datacomm.ARece	AReceiveMessageForward	20	notifyPortReceive
19	inputport.datacomm.simpl	AGenericSimplexBufferSer	156	notifyPortReceive
20	inputport.datacomm.simpl	AGenericSimplexBufferSer	87	messageReceived
21	inputport.datacomm.simpl	AnNIOSimplexBufferServer	92	socketChannelRead
22	inputport.datacomm.simpl	AReadCommand.java	80	notifyRead
23	inputport.datacomm.simpl	AScatterGatherReadComm	99	maybeExtractMessageAnd
24	inputport.datacomm.simpl	AScatterGatherReadComm	35	processRead
25	inputport.datacomm.simpl	AReadCommand.java	43	execute
26	inputport.datacomm.simpl	ASelectionReadManager.ja	45	processRead
27	inputport.datacomm.simpl	ASelectionManager.java	180	processSelectedOperation
28	inputport.datacomm.simpl	ASelectionManager.java	207	run

Inheritance Hierarchy

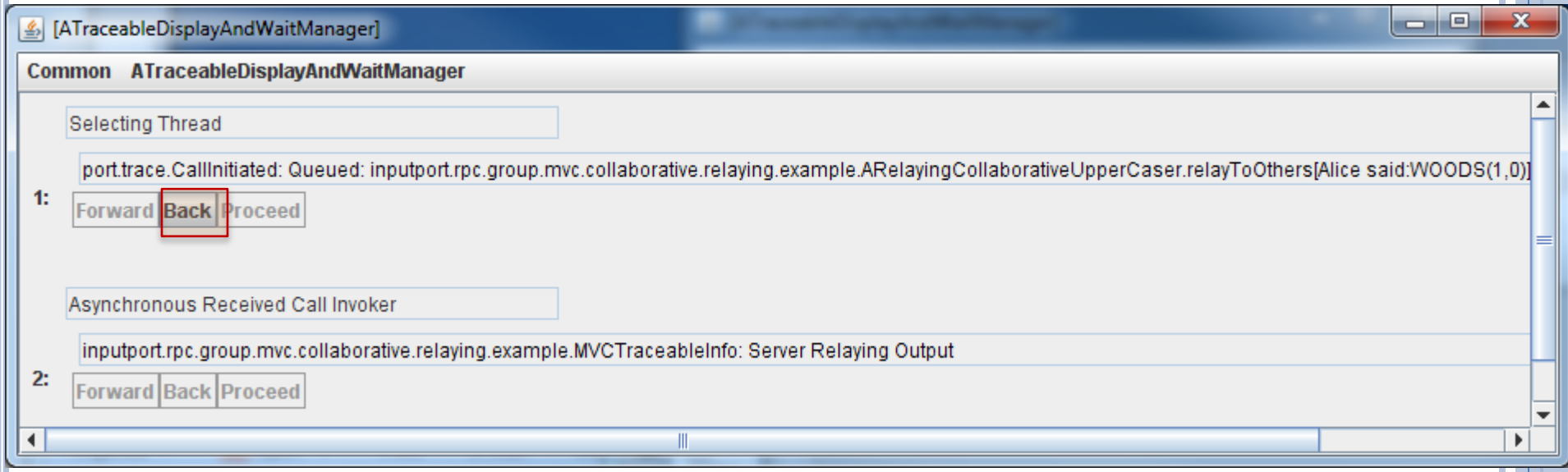


Exploring Variables: Event Properties

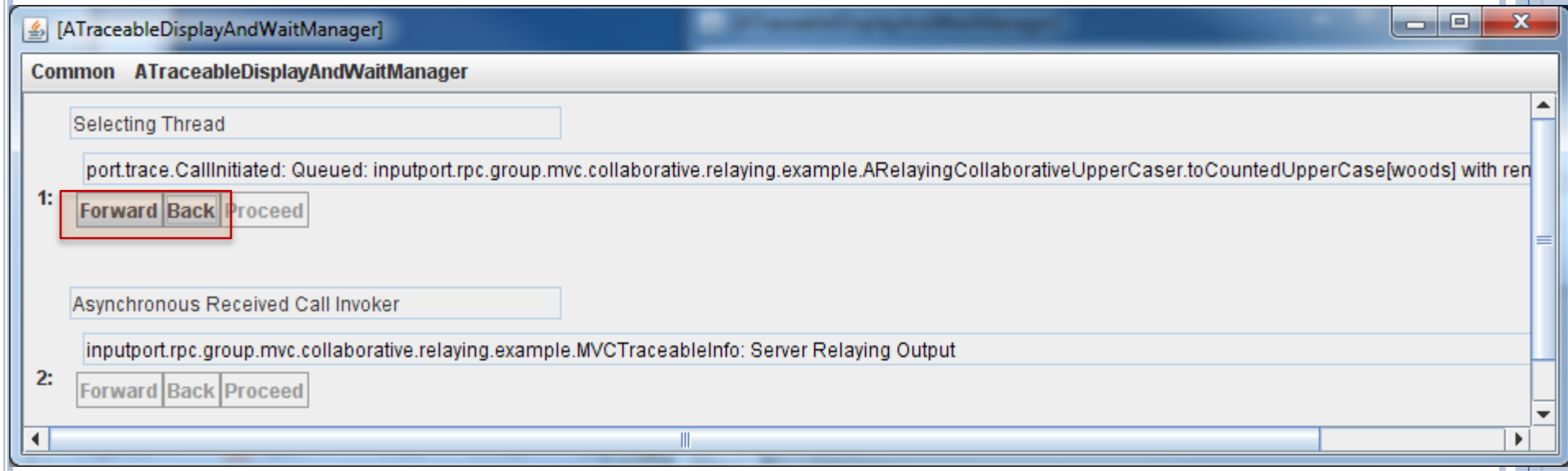


Can see properties of event by calling getters

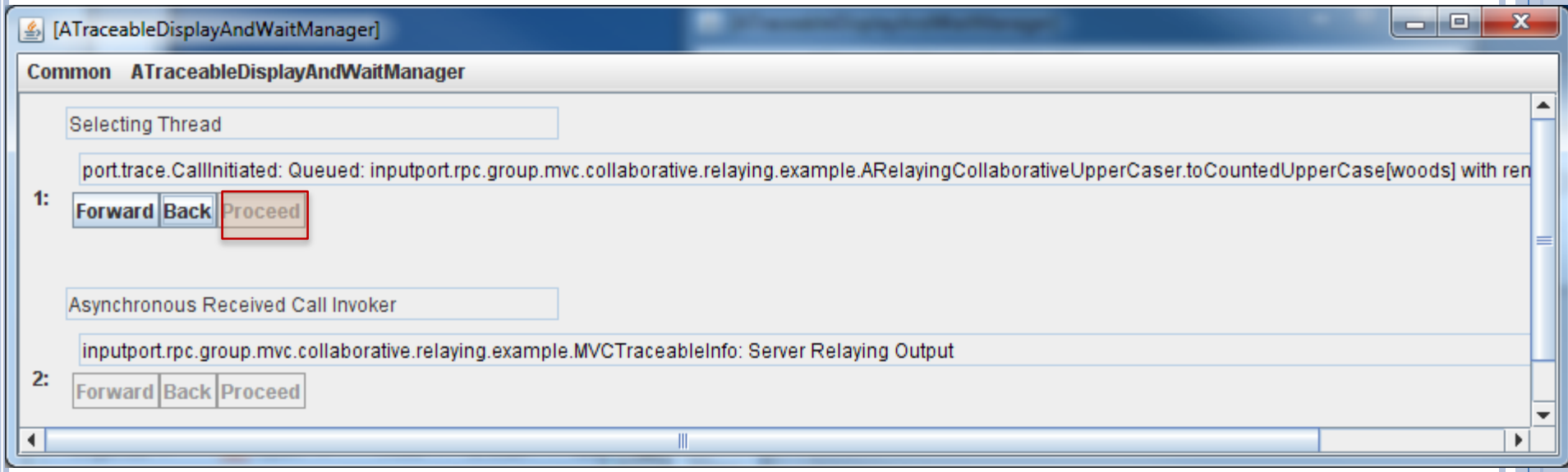
Thread Has History of Typed Trace Info



Browsing: Last to Last Call



Resume: Proceed Button



Set Break Point: Pause/Display all Announced Messages?

May want to pause only some of the announcements

May want to display only some of the announcements

How to specify a set of related announcements that should be displayed or paused?

Using Announcer Object Attributes

static setKeywordDisplayStatus (Object announcer, boolean status)

static setImplicitDisplayKeywordKind (ImplicitKeywordKind val)

static setImplicitWaitKeywordKind (ImplicitKeywordKind val)

Tracer

default

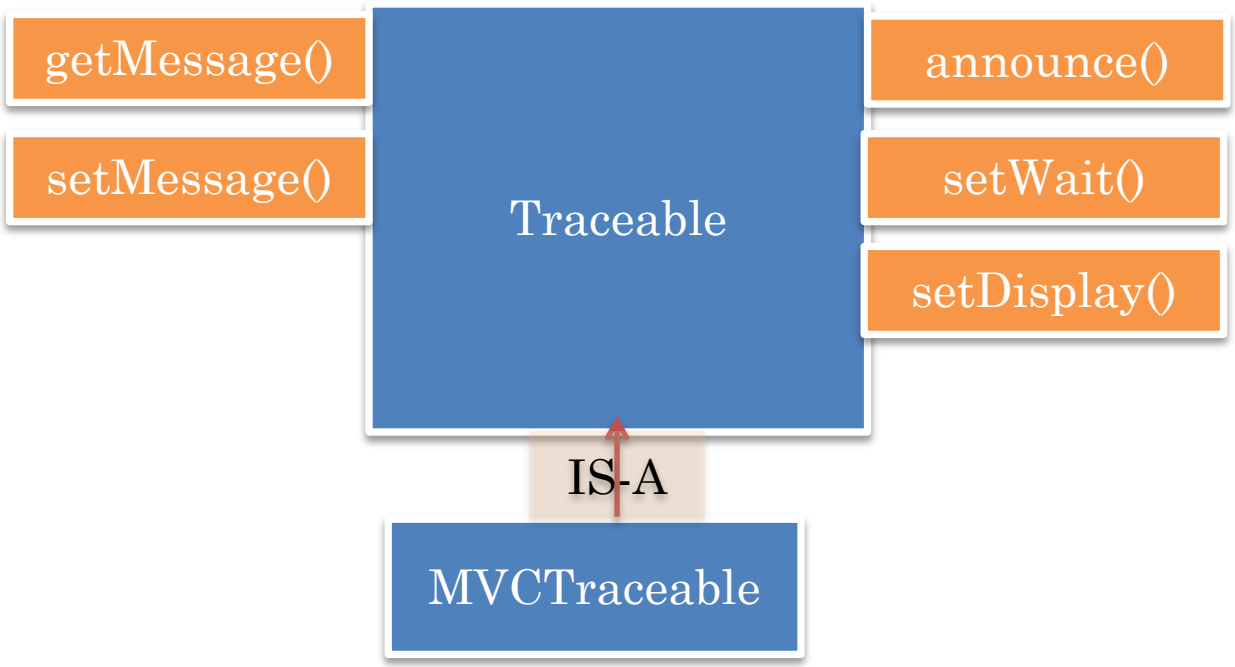
```
public enum ImplicitKeywordKind {
        OBJECT_TO_STRING,
        OBJECT_CLASS_NAME,
        OBJECT_PACKAGE_NAME }
```

setKeywordDisplay(Wait)Status(announcer, true(false)) says that if an event is announced by an object whose toString()/class/package attribute is that of the announcer then it should be displayed(wait).

Print, Display and Wait are three different things you can do with traced information

setImplicitDisplay(Wait)KeywordKind determines if toString(), class or package attribute is used for all events

Using Event Object



Debugger Issues Resolved

- Debugger makes it difficult to test race conditions
- All threads and processes mapped to a single code window
- Cannot see the history of actions taken by a thread
- Break points do not transfer to another computer
- Cannot use a mechanism to set multiple related debug points

Summary

- A Message bus allows communication of events among a set of related objects such as debugger and editor.
  - An event announced by an object are sent to all objects listening to it.
- A Message bus can use a semi-wait free algorithm to avoid conflicting conecurrent announcements.

Summary

- A Traceable bus is a message bus that allows one to simultaneously see the values of Traceable objects announced by different threads of a process.
- One of the listeners of these objects is a single ATraceable DisplayAndWaitManager
- The notification callback called in this object in response to the announcement of a Traceable:
  - Displays information about the Tracable if its Display property is true.
  - Blocks the thread until the user unblocks it if the Wait property of the Traceable is true.
- Primitives are provided to set the display and wait property of all Traceables announced by
  - all code in a class,
  - all coce in a package
- A Message bus can use a semi-wait free algorithm to avoid conflicting conecurrent announcements.