Technologies evaluation

# Overview

This document aims to synthetic all advantages and disadvantages from the technologies that were studied.

# Web page

**Advantages:**

* **[Thirft, ICE, TAO]:** Web page dedicated to the product. It focuses the user’s mind in the product.
* **[ICE]:** ICE web page is intuitive, simple but with a design that makes think in a professional product.
* **[Thrift, ICE] :** Design in vertical format with a top menu.
* **[Thrift, ICE]:** The home page tries to capture the user. It has a brief resume of the product, instead of the typical corporation image. In this resume is pointed out all programming language supported, the different licenses that are offered and other highlights of the product.
* **[Thrift]:** The home page has a direct download link of the product and a link to a *Getting Started* example.

**Disadvantages:**

* **[OpenSplice, JavaRMI]:** Web page dedicated to a lot of products. It is difficult for user to find that it needs.
* **[Thrift]:** Thrift web page is too simple. It makes think in a non-professional product.

# Documentation

**Advantages:**

* **[TAO, ICE]:** Good online documentation in their web pages.
* **[TAO, JavaRMI]:** They have a lot of online documentation from several sources, not only from their web pages.

**Disadvantages:**

* **[TAO]:** User manual is not free.
* **[ICE, Thrift, JavaRMI]:** Their packages don’t contain documentation.
* **[TAO, Thrift]:** They don’t have good installation documentation.

# Software

## Paradigm

All products are RMI except Thrift. Thrift is RPC.

## API

**Advantages:**

* **[Thrift, ICE, TAO, OpenSplice]:** Usage of share pointers to avoid memory leaks from user.
* **[Thrift, ICE, OpenSplice, JavaRMI]:** Easy API from all objects that user has to use.
* **[Thrift]:** It is modular and it makes a good usage of interfaces. With this approach Thrift has achieved to implement a lot of functionality, as different transports, different protocols, etc...
* **[ICE]:** Connections are configured by strings.
* **[ICE]:** Configuration can be stored in files.
* **[ICE]:** Offers asynchronous calls for all procedure calls. This asynchronous calls accepts a callback.

**Disadvantages:**

* **[Thrift]:** The user has to manage too object because there isn’t dynamic mechanism to configure Thrift.
* **[TAO]:** TAO is a powerful middleware but it is complicate to develop using tools that leave from a simple usage. Example: create a server with thread pool.

## Behavior

**Advantages:**

* **[Thrift]:** It offers to user a lot of different transport, protocols, type of connections (like TServerSocket).
* **[ICE]:** It is a good simplification of CORBA. It is focused to make a easy middleware to achieve a RMI implementation.
* **[JavaRMI, Thrift, ICE, TAO, OpenSplice]:** Their implementation offers asynchronous invocations.
* **[Thrift, ICE]:** They permit to change the serialization/deseralization mechanism.

**Disadvantages:**

* **[TAO]:** As CORBA implementation, it can be defined like a monster because it has a lot of functionalities that are defined in CORBA standard.
* **[OpenSplice] :** It doesn’t work in a WLAN scenario. OpenSplice RMI is focused in use DDS with RMI paradigm, instead of create a RMI product.

## Supported programming languages

**Advantages:**

* **[Thrift, ICE, OpenSplice]** They support several programming languages.
* **[ICE]:** It has support from mobile devices.

**Disadvantages:**

* **[TAO, JavaRMI]:** Only support one programming language.
* **[Thrift]:** For each programming language that it support, there is a different implementation.

## Platforms

**Advantages:**

* **[ICE, TAO, JavaRMI, Thrift , OpenSplice]:** Support Windows and Linux.

**Disadvantages:**

* **[TAO, Thrift]:** User has to compile their libraries.

## Supported types

**Advantages:**

* **[Thrift]:** Support several containers as maps, dictionaries.
* **[ICE, TAO]:** Supports inherence.
* **[ICE] :** Supports classes.
* **[JavaRMI]:** Supports all Java types.
* **[ICE, Thrift, TAO, JavaRMI]:** Support dynamic types.

**Disadvantages:**

* **[OpenSplice]:** Doesn’t support unions.

## Programming model

**Advantages:**

* **[TAO, ICE, Thrift]:** User has to make four steps:  
  1. Write a IDL file.  
  2.Generate code from IDL.  
  3. Implements the client application.  
  4.Implement the server application.
* **[TAO]:** its IDL compiler generates little code.

**Disadvantages:**

* **[JavaRMI]:** It doesn’t use a IDL file to define the interfaces. User has to create the class and inherits it from a based class. Based class specifies that the user class will be use by JavaRMI.
* **[OpenSplice]:** User needs to compile two IDL files.

## Installation

**Advantages:**

* **[OpenSplice, ICE]:** They offer an easy installer for Windows and packages for Linux.

**Disadvantages:**

* **[TAO, Thrift]:** User has to compile their libraries.
* **[JavaRMI]:** It is installed with Java packages

## External libraries

**Advantages:**

* **[TAO, OpenSplice, JavaRMI]:** Doesn’t need external libraries.

**Disadvantages:**

* **[Thrift, ICE]:** Needs some external libraries.

## Making an example

**Disadvantages:**

* **[Thrift]:** Both have compilation errors. In Linux the source code doesn’t have all includes. In Windows Jaime has to apply patches.
* **[OpenSplice]:** License issues.
* **[ICE, JavaRMI, TAO]:** Doesn’t create project files.

# Technical details

## Memory usage

**Advantages:**

* **[TAO, ICE, Thrift]:** Use smart pointers.

## Threads

**Advantage:**

* **[Thrift]:** It uses BOOST threads.

**Disadvantages:**

* **[TAO, ICE]:** they use their own threads.

## Performance

**Advantages:**

* **[Thrift**

**] :** It has good performance. Only protocol buffers is better.