

# HTML5 based Smart eService using Server Side JavaScript and JADE environment

Dr. Enas Hadi Salih  
Software Engineering Dept. / Al-Rafedain University Collage  
[dr.enashadi@yahoo.com](mailto:dr.enashadi@yahoo.com)

## Abstract

This paper introduces the concept of Smart-eService to be the kernel building block of the Smart-eGovernment. The presented Smart-eService has many privileges over the traditional eService such as fully cross-platform, social, liable, negotiable, autonomous and mobile.

Smart-eService is implemented in this paper as a deliverable mobile agent that complies with HTML5 as a frontend and Node.js modules to interface JADE platform at the backend. The presented Smart-eService is preemptive behavior rather than reactive or even proactive; preemptive interpreted as actions based upon hard domain intelligence.

HTML5 terminologies such as Real Multi-Threading and WebSockets have been exploited and deployed to efficiently increase the performance of Smart-eService and sustain its attributes.

**Keywords:** Smartphone, smart-eGovernment, eService, Mobile Agent, Smart-eService, Node.js, HTML5, JADE

## 1. Introduction

eGovernment is defined as The employment of the Internet and the world-wide-web for delivering governmental information and services to the citizens [1]. The kernel core of eGovernment is the eService which it is what citizens expect from the eGovernment to deliver (i.e., provide/publish). eServices designed to be consumed per requested citizen; this happened in two scenarios:

- 1- Deliver eService entirely to citizen side and consume it there; this approach imposes delivering eService as a standalone application; this has to consider the compatibility of the delivered eService module with the designation platform. Many serious challenges face this approach such as security risks, enough knowledge to install/uninstall eService, and experience to troubleshoot malfunctioning of installed eService; this is what 'Apps store' from Google and other providers are doing.
- 2- Decompose eService into backend component to be consumed on the server side, and frontend component to be delivered and consumed at the citizen side; this approach throws less burden on the citizen in term of starting the eService but it leaves the delivered eService with less functionalities and having the same issue of compatibility issue.

Both presented scenarios do not respond to the specification of Smart-eGovernment, which is the next generation of eGovernment.

In Smart-eGovernment, eServices hold more smart behavior in its interaction with citizens and the environment. Smart behavior can be viewed as the ability to acquire knowledge from external or internal resources; furthermore, smart behavior is able to build semantic relationships over the acquired knowledge to sustain the cognitive understanding of the environment.

In this paper, Smart-eService terminology is introduced and implemented using HTML5 technologies; this grants Smart-eService significant features such as fully independent of the underlying platform; where only web browser is needed to consume Smart-eService and it does not require knowledge to install/ uninstall or invoke the eService; this is beside the fact that web browser runs with limited privileges which reduces the risk of malfunctioning effect of the delivered Smart-eService while traditional eService which is installed within the device can ruin the entire system.

Two software technologies have been deployed in this paper: first one is the intelligent software agent and second technology is the HTML5 that promoted web browser to be an efficient execution environment, both technologies are introduced briefly in this paper.

## 2. eService Vs. smart-eService

eGovernment is a collection of eServices published for the public citizens as part of the commitments of a government. eService can be defined in a broad sense as the provisioning of services over electronic network [2]. The production of eService considers three main components to produce reliable eServices [1, 2]:

- **eService Implementation and Environment:** eService should be implemented to be cross-platform consumable software module; this is crucial in eGovernment due to the huge size of population. eService implementation should be adaptive to the environment holds be citizens to be consumed properly.
- **eService Publishing and Discovery:** eGovernment can be considered to be a large repository of eServices. UDDI ( Universal Description and Discovery Integration ) is a software
- **eService Delivery and Integration:** this issue draw main schemes of accessibilities to eServices published by the eGovernment where citizens can invoke and consume available eServices.

Smart-eService is an eService delivered from eGovernment repository to citizens' machine to fulfill their requests by autonomously contact information resources and retrieve knowledge. Smart-eService encompasses the ability to perceive the environment and cognitively adapt to the rapid changing within the deploying environment to fulfill citizen needs.

The ultimate objectivity of Smart-eService is to help citizens making decisions in certain problem domain by acquiring knowledge to make that decision more effective; Markman in [3] introduced the effective thinking in problem solving as *high quality knowledge* where problem solving is less about a flash of intuitive brilliance and more about the application of knowledge [3].

Smart actions are produced by software modules and applications installed in a device, for a software to be smart it should be able to assist clients making wise decisions by building semantic network over the domain and developing knowledge by traversing that network autonomously. Knowledge development is accomplished without the intervention of clients.

## 3. eService Delivery Scenarios

eService is a software module when consumed it produces valuable effects on hosting environment, traditionally there are two approaches to have eService delivered to designation system: first is the direct installation of the software service, here the service is written using platform dependent languages (e.g., C++, J2ME, Special Java for Android); this service is published in global repositories which client should login and download it. The second approach is accomplished by exploiting web browser where eService is written using web languages like XML and JavaScript; this way eService can be delivered anywhere to the destination system since there is web browser installed (i.e., some Web technologies like HTML5 provides new tags that need updated versions of internet explorer).

## 4. eService Liability

eService is a software entity that consumed over the clients' machine by exploiting available resources; this raises many alerts and risks of exposing personal and sensitive information, especially when considering Smartphone that holds very sensitive information about the client/citizen. In this paper we define eService feasibility as a set of *beliefs* initiated and developed at the run time:

$$S.L = \sum_{i=1}^N b_i$$

$b_i$ : is the  $i^{\text{th}}$  belief, and belief is a set of ontologies that represent initial restriction concepts (i.e., what are authorized actions and responses).

## Intelligent Software Agent and JADE

Intelligent Software Agent is a software entity that can perceive the environment through its sensors and acts on that perceiving autonomously. The perceiving is domain specific and a sophisticated process that needs

ontological representation for the concepts compasses the environment [4]. In Agent based system, the problem domain is represented using ontology which is defined as the following:

$$\text{Ontology} = \{C, O, R, P\}$$

Where

C: are concepts defined over the problem domain

O: Object within the problem domain

R: Relationships between concepts

P: properties of Concepts

JADE is a developing environment written in Java languages to assist developing Agent systems. In [4] JADE is presented in details, thus in this paper we are not going to introduce JADE and focus on building wrapper classes only.

## 5. Smartphone

There are many definitions for the meaning of Smartphone but the most convenient shared definition is: “Smartphone is a device with high level of computation power and large screen” [5].

Smartphones are devices combined both the functionalities of regular phone and other functionalities which were only implemented on different devices such as the PDA, PC or Laptop. Smartphones significance came from the operating system (e.g., android from Google and windows mobile from Microsoft); the operating system provides enough resources to lunch applications that can collect knowledge for the user of the Smartphone; this imposes the availability to connect information resources such as the internet or information repositories[5][6]. Smartphones are running the latest versions of web explorers such as Chrome, Firefox, Internet Explorer and others

## 6. HTML5 –The New Platform for Developing Software

HTML5 is a co-operation between the World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG). New technologies have been introduced by HTML5 like WebSocket and worker Multi-threading; these new technologies are a promising environment for developing more complex and safer software modules; this is due to the limited access right granted to the web browser which is the shell under which HTML5 based software modules are executed[7].

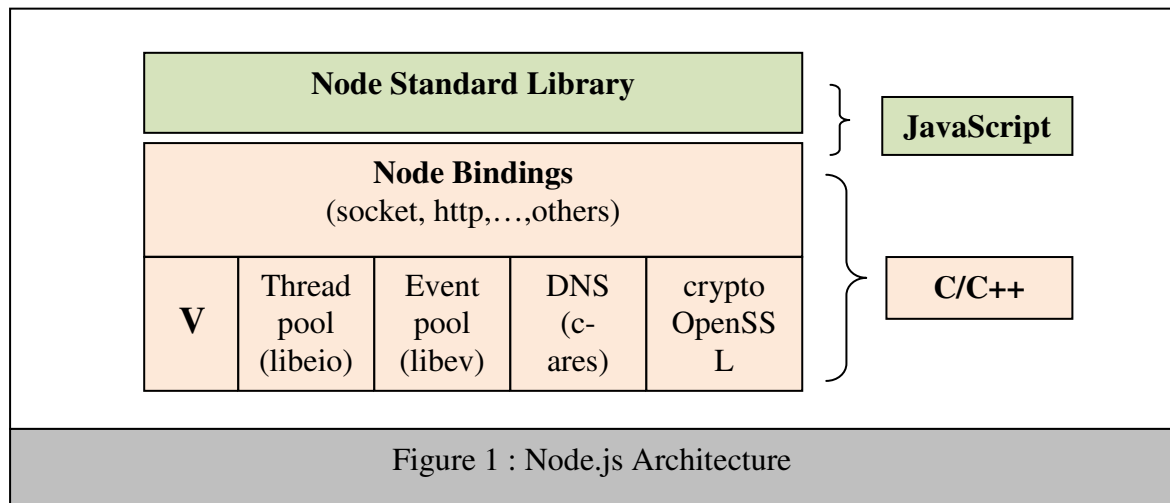
Enterprise web applications are tend to use web browser as Thin client; this is where all needed to execute web applications functionalities are available in the web browser, and what is missing can be added as plugins (e.g., flash player). Most of today’s smartphones have web browsers that support HTML5, CSS3 and JavaScript which represent the only requirement to develop more sophisticated software [7]

## 7. Google V8 engine and Server Side JavaScript (node.js)

V8 JavaScript engine is an open source JavaScript engine developed by Google in Denmark and shipping with the Google Chrome browser. V8 adds a virtual JavaScript machine and a special operating system (i.e., Chrome OS) which runs on x86 machines and Advanced RISC Machines (ARM) [8]. V8 increases performance by compiling JavaScript to native machine code before executing it, rather than to a bytecode or interpreting it. Further performance increases were achieved by employing optimization techniques such as inline caching [8,9].

*Node.js* is an open source toolkit for developing server side applications based on the V8 JavaScript engine. *Node.js* is written in C++ which is the same language used to write V8 engine; this brings a lot of correlations among programs written for server side and client side, anyway, *Node.js* extends JavaScript API to build web applications at higher level of scalability and extreme performance due to the non-stop feature [9].

V8 enables C++ applications to expose its own objects and functions to JavaScript where objects are wrapped in V8 templates. Two types of templates are available: Function template and Object template, and C++ object’s methods are accessed in an internal field stored within function template instances.[9]



### 8. Smart-eService implementation as Mobile Agent (The Proposal)

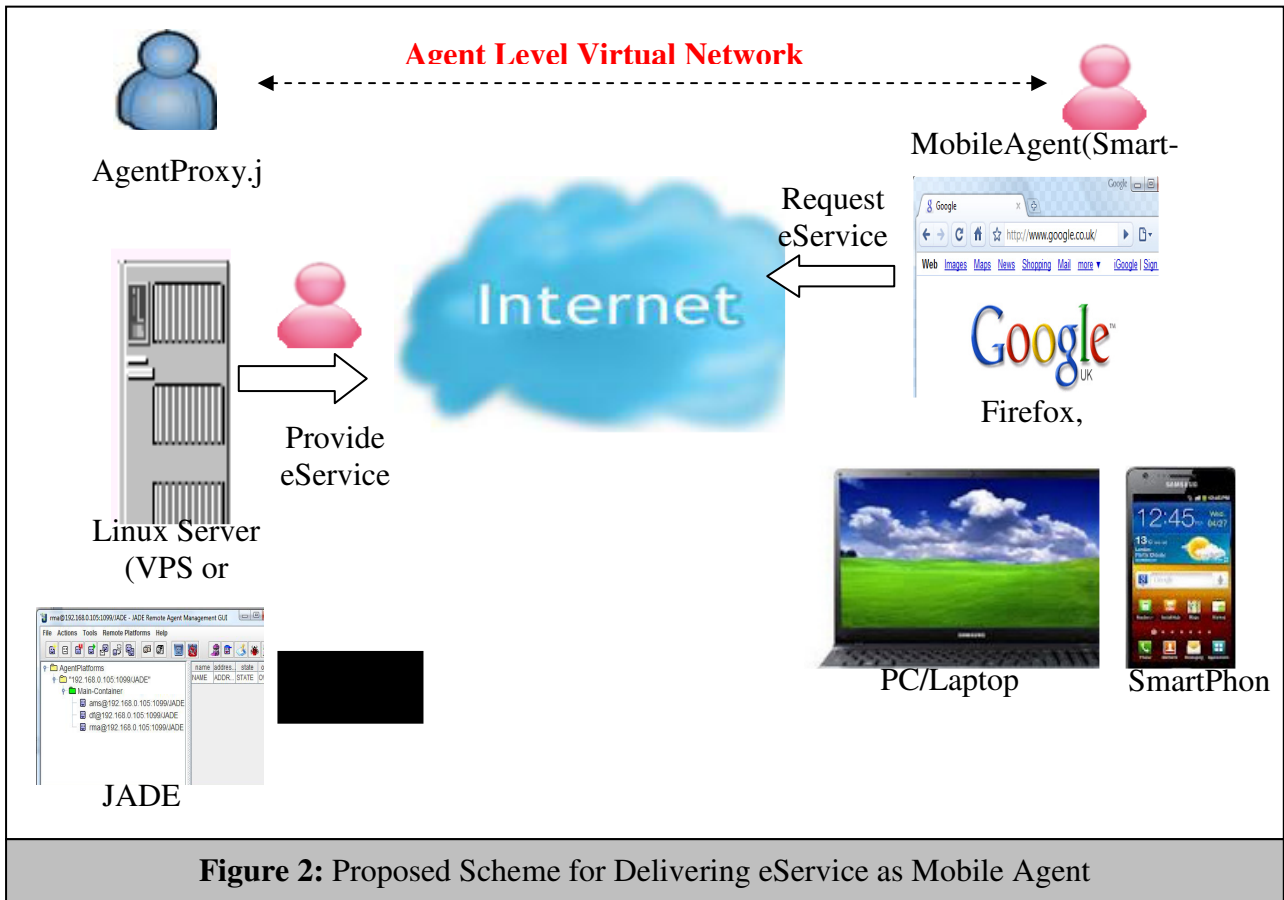
The proposed system is a collection of software technologies, methodologies and protocols to implement eService as mobile agent module; this has been a worthwhile effort to promote the deployment environment toward introducing smart and later on intelligent behaviors.

We have objectives to be fulfilled in the proposed system as the following:

- 1- Implementing eService as smart, social, mobile and autonomous software module.
- 2- Implementing eService as fully platform independent software module that works on Smartphones, laptops and other devices that have Chrome or Firefox internet explorer.
- 3- Implementing eService as on demand utility that can invoked without a need for installation
- 4- Implementing eService with attributes that grants eService the ability to exploit and develop domain intelligence

As it has been introduced in previous sections of this paper, web browser has been evolving to be a rich development environment for software applications; this is especially recognized with the emerging of HTML5 tags (i.e., Multi-threading, primitive data types); we focused on developing eService as JavaScript module which is transferred to requested client and launched in the internet explorer. Using JavaScript was due the promotion of this language to be client side as well server side (i.e., Node.js is a server side JavaScript environment).

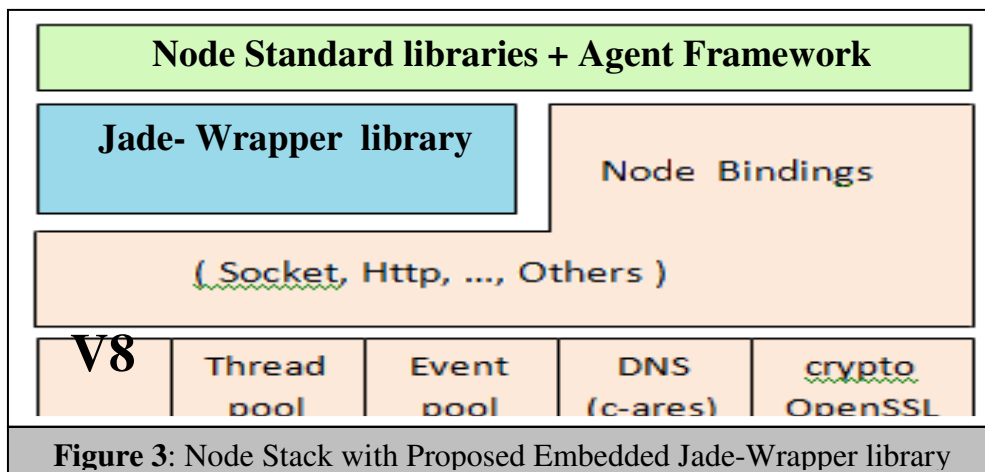
Implementing eService as JavaScript supports the fulfillment of objectives (2) and (3); this JavaScript eService is designed to be FIPA-Compliant or in other words, to be compatible with JADE platform as in figure (2), and this approach supports the fulfillment of objectives (1) and (4):



**Figure 2:** Proposed Scheme for Delivering eService as Mobile Agent

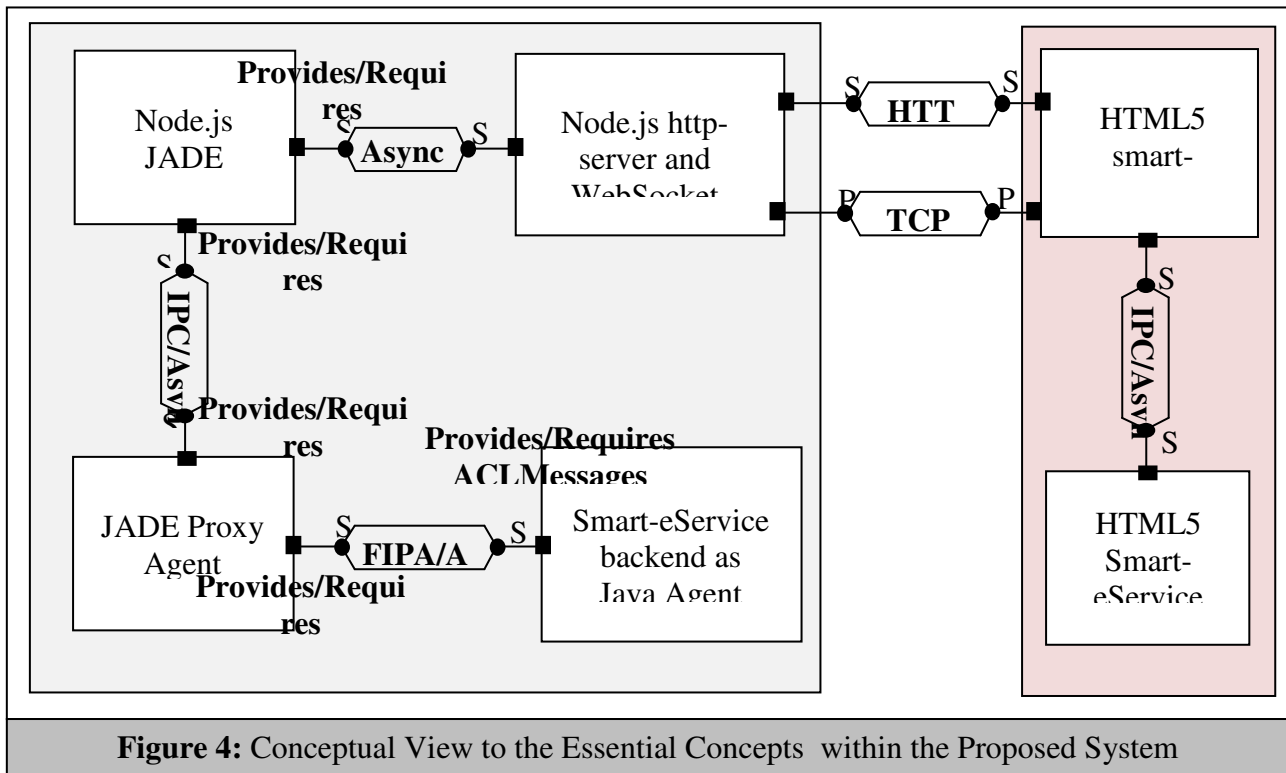
As figure (2) presents, eService is delivered to the requested client and behave as JADE agent; this has been a challenge due to the need to equip JavaScript eService with a lot of functionalities (i.e., JADE is a collection of classes used by Agent to accomplish its tasks); these are so hard to be implemented all using only JavaScript, thus we have implemented it as two components: first one is a light client side JavaScript module which is to be delivered over the internet to the client and Second one is heavy server side JavaScript module which is designed and implemented as Node.js module called JADE-Wrapper library, as it is presented in figure (3)

JADE-Wrapper library also contains many interfaces to JADE utilities (i.e., RMA agent, Sniffing agent and others)



**Figure 3:** Node Stack with Proposed Embedded Jade-Wrapper library

Figure (4) presents the conceptual view to the essential components that compose the proposed system.



**Figure 4:** Conceptual View to the Essential Concepts within the Proposed System

Figure (5) present server side Node.js module with stub code for JADE wrapper:

Figure (5) presents the implementation partial Node.js code JADE wrapper module with the conceptual view.

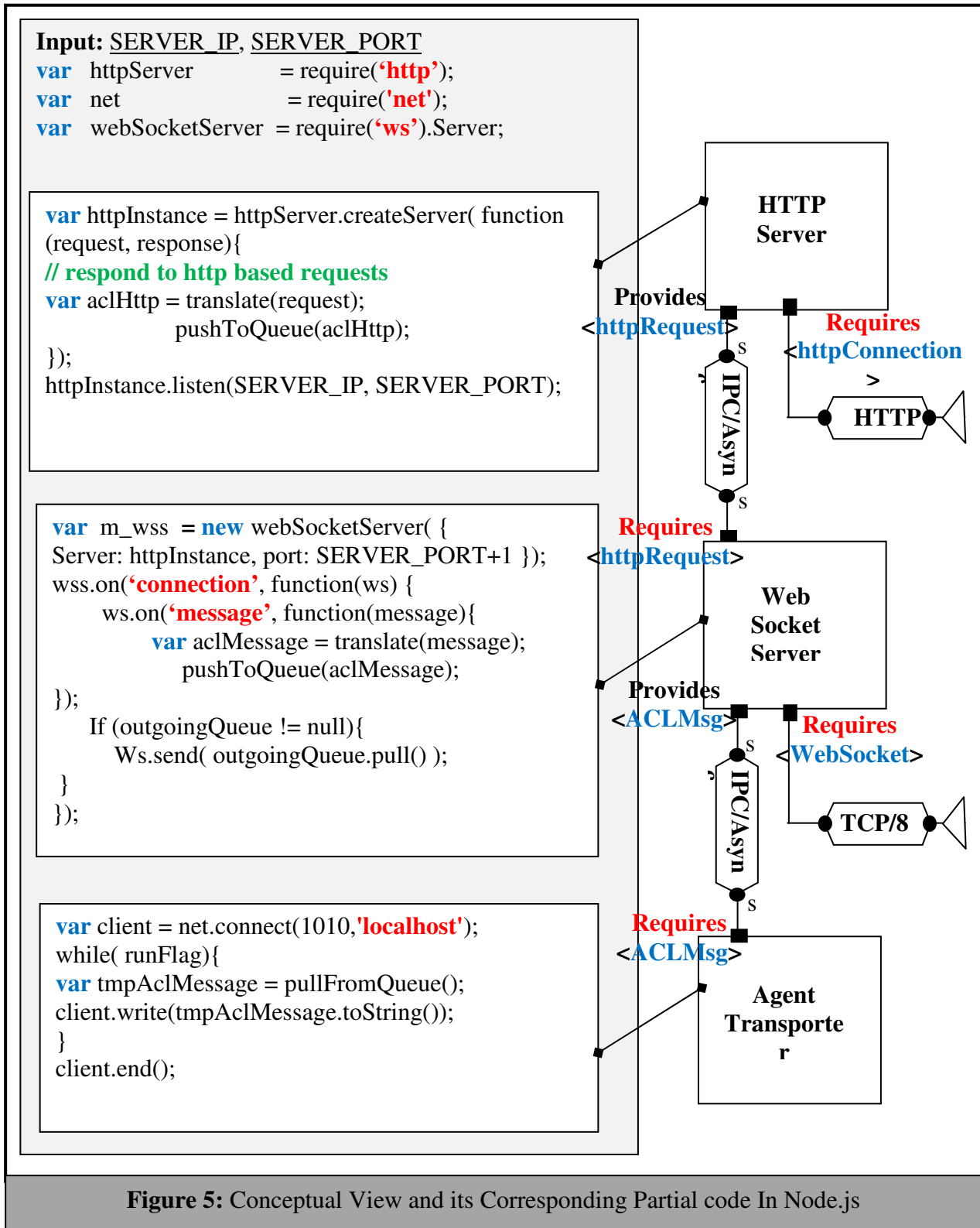


Figure (5) presents partial HTML5 code to load and start Smart-eService

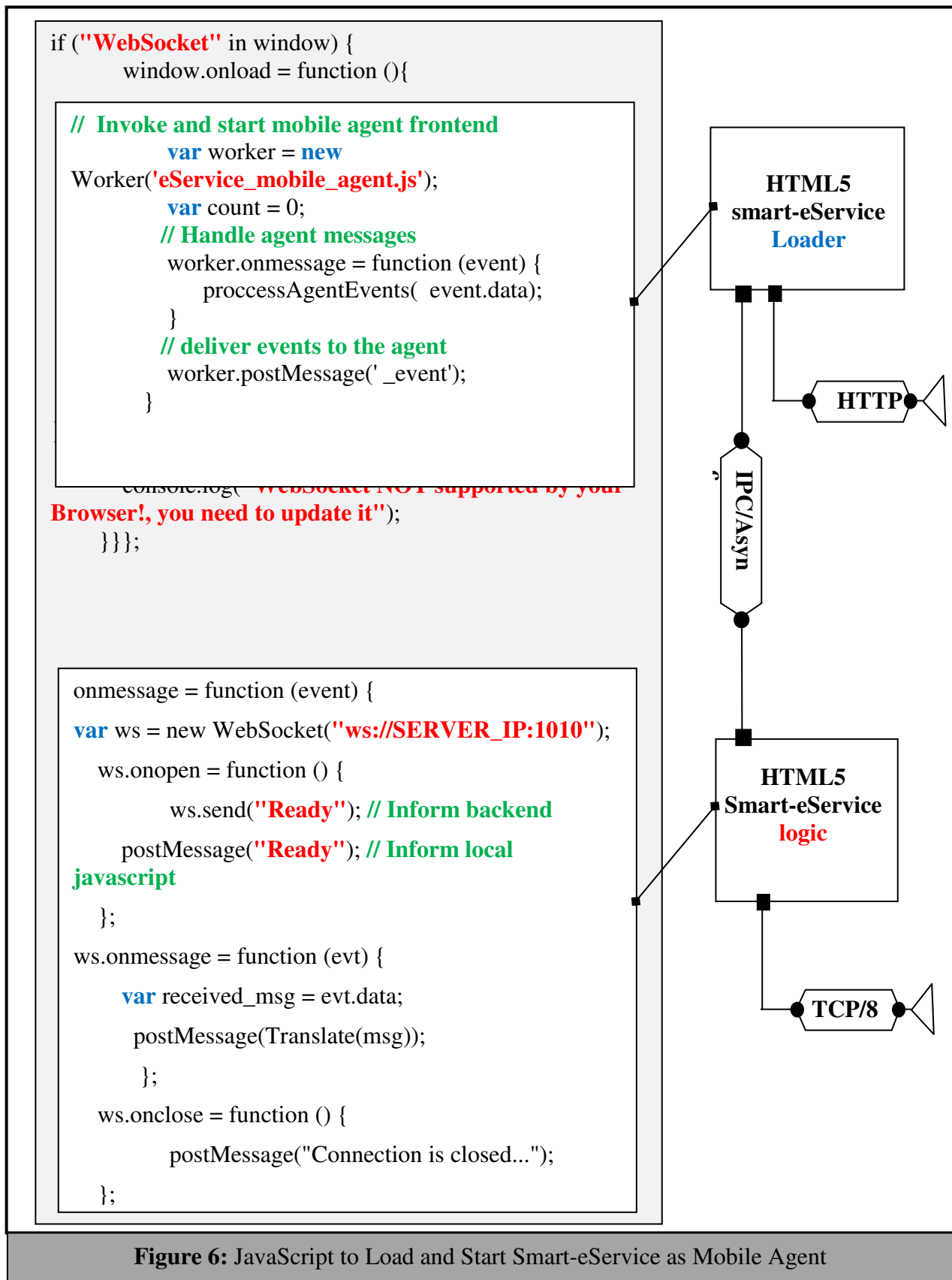


Figure 6: JavaScript to Load and Start Smart-eService as Mobile Agent



Algorithm (1) is the backend Java Agent which is designed and implemented using JADE environment

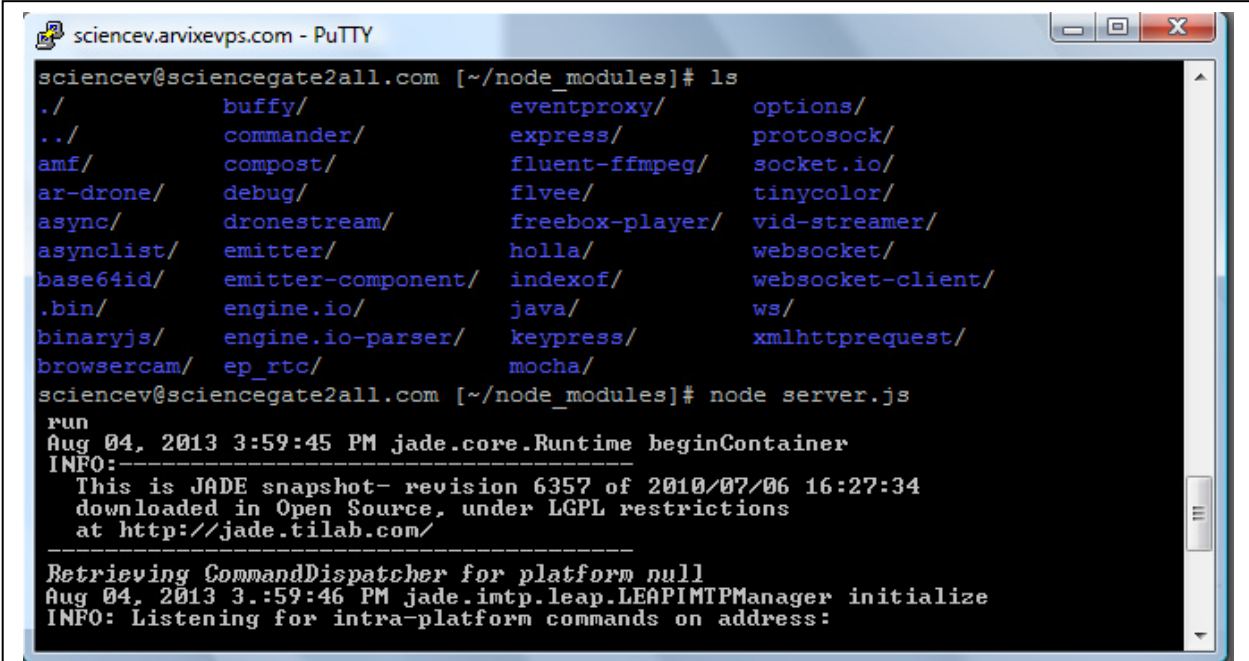
### Algorithm 1: Backend Java Agent Interaction to JavaScript Mobile Agent

```
public class AgentProxy extends Agent {
Begin
  Declare RequestQueue as private Queue<String>
  public void setup()
    Begin
      Instantiate RequestQueue as ConcurrentLinkedQueue<String>();
      Instantiate ResponseQueue as ConcurrentLinkedQueue<String>();
      Instantiate TempQueue as ConcurrentLinkedQueue<ACLMessage>();

      Object [] receivedArguments = getPassedArguments();
      RequestQueue = ( Queue<String>) receivedArguments[0];
      addBehaviour(new CyclicBehaviour()
        Begin
          public void action()
            Begin
              if(reqQueue.size() > 0)
                while(reqQueue.size() >0)
                  Begin
                    Var mobileAgentMsg = RequestQueue.poll();
                    ACLMessage msg = (ACLMessage)
Translate(mobileAgentMsg);
                    TempQueue.push(msg);
                  End
                End
              End
            addBehaviour(newCyclicBehaviour()
              Begin
                Var msg2 = (ACLMessage) TempQueue.poll();
                String str = GetResponseAsString(msg2);
                ResponseQueue.push(str);
              End);
          End );
    End

```

Figure (7) presents a screenshot of starting Node.js module in a Virtual Private Server (VPS) , till this research Node.js does not have JADE wrapper despite the fact that JADE is a widespread Agent development environment.

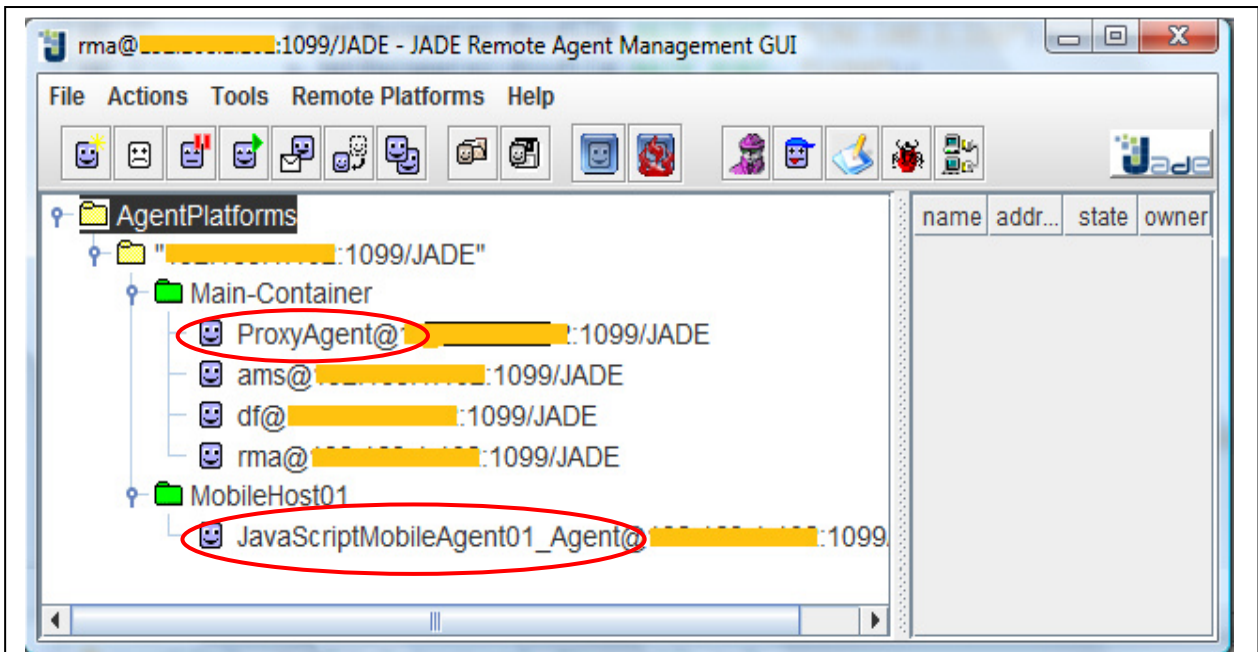


```
scienceev@sciencegate2all.com [~/node_modules]# ls
./          buffy/      eventproxy/ options/
../         commander/ express/    protosock/
amf/       compost/   fluent-ffmpeg/ socket.io/
ar-drone/  debug/     flvee/     tinycolor/
async/     dronestream/ freebox-player/ vid-streamer/
asynclist/ emitter/    holla/     websocket/
base64id/  emitter-component/ indexof/   websocket-client/
.bin/      engine.io/  java/      ws/
binaryjs/  engine.io-parser/ keypress/  xmlhttprequest/
browsercam/ ep_rtc/     mocha/

scienceev@sciencegate2all.com [~/node_modules]# node server.js
run
Aug 04, 2013 3:59:45 PM jade.core.Runtime beginContainer
INFO:-----
This is JADE snapshot- revision 6357 of 2010/07/06 16:27:34
downloaded in Open Source, under LGPL restrictions
at http://jade.tilab.com/
-----
Retrieving CommandDispatcher for platform null
Aug 04, 2013 3.:59:46 PM jade.imtp.leap.LEAPIMTPManager initialize
INFO: Listening for intra-platform commands on address:
```

Figure 7: screenshot of console output when calling Server.js

Figure (8) presents JADE GUI has been started with ProxyAgent and a Mobile Agent instance



**Figure 8:** JavaScript Mobile Agent Started in Container MobileHost01

in figure(8) JavaScriptMobileAgent01\_Agent is launched due a call from the client side over http connection or WebSocket connection. Figure (9) presents sniffing utility been deployed to capture ACLMessages exchanged between ProxyAgent at the backend and the Mobile Agent eService at the frontend.

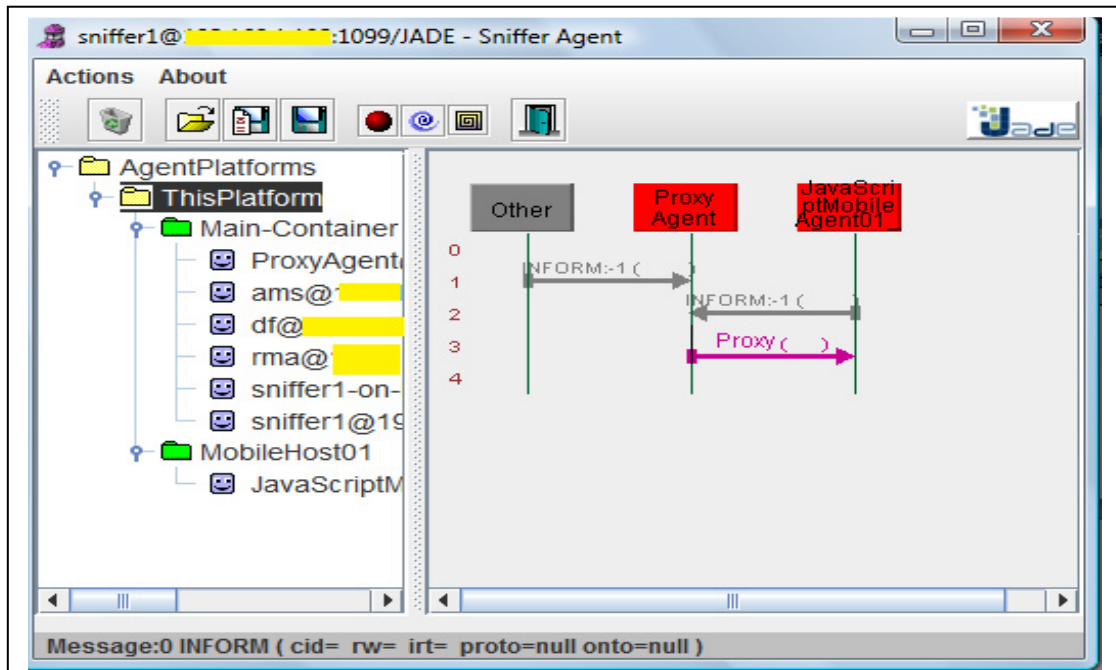


Figure 9:

Figure (10) presents ACLMessages exchanged between Smart-eService and Java Agent running at the server side.

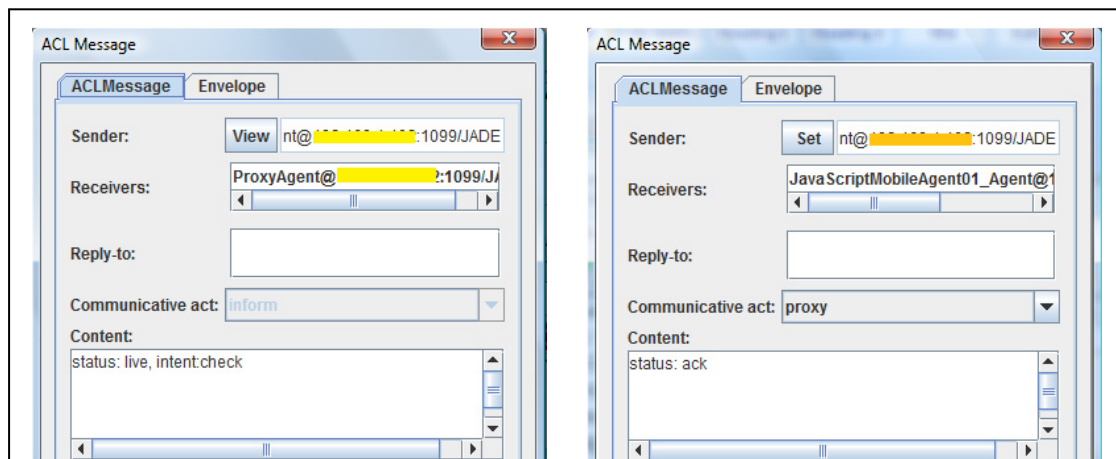


Figure 10: ACL Messages exchange between JavaScript Smart-eService and Java

## 9. Conclusion

- 1- JavaScript running under V8 engine introduces efficient open source development environment in developing large applications. Along the implementation of the Mobile Agent using JavaScript language, many JavaScript engines have been tested (i.e., Firefox, Internet Explorer, Safari, Opera), and V8 Chrome shown recognized performance level.
- 2- Implementing Smart-eService as JavaScript mobile agent has added significant attributes to the eService like autonomous, social and other features mentioned in above sections. Embedding intelligent software terminologies within enterprise web application is expanding the opportunities of having smarter enterprise software modules.
- 3- JavaScript is an open source language; this introduces new challenge in implementing eService using JavaScript where client, with little experience in JavaScript, can hack eService manually or automatically, in the proposed system we divided the processing into frontend and backend; this methodology hardens the penetration into the eService.
- 4- Smart-eService mobility is valid only during the web session, when requester closes the session Smart-eService shuts down all its behaviors (e.g., interacting the environment, socializing other agents and mobility); this issue has cons and pros where from a side it increase code safe execution; this due to the limited authorization of web browser and from another side it affects the availability of the Agent.
- 5- eService Mobility is guaranteed when implementing Smart-eService using JavaScript language; this is due to the standard terminologies adopted by all smart devices ( Smartphone, Smart PDA, ..., and others) that have updated web browser installed. Smart-eService can be viewed as a web service that runs on Node.js, and maintain all invocation and deploying standards such as XML (Extensible Markup Language), SOAP ( Simple Object Access Protocol ), HTTP ( Hyper Text Transfer Protocol) and WebSocket.
- 6- Implementing Smart-eService using JavaScript increases the potential of Web application to hold Intelligent Infrastructure of the next generation of eService, which is the Intelligent-eService

## 10. References

- 1- Hossein Bidgoli, "Handbook of Information Security, Key Concepts, Infrastructure, Standards, and Protocols", John Wiley & Sons, Inc., USA, 2006, ISBN-13: 978-0-471-64830-7.
- 2- Roland T. Rust, and P.K. Kannan, "E-Service: New Directions in Theory and Practice", M.E., Sharpe, Inc., USA, 2002, ISBN: 0-7656-0806-5.
- 3- Art Markman, " Smart Thinking: Three Essential Keys to Solve Problems, Innovate, and Get Things Done", Penguin Group, USA, 2012.
- 4- Fabio Bellifemine, Giovanni Caire, and Dominic Greenwood," Developing Multi-Agent Systems with JADE", John Wiley & Sons Ltd, USA, 2007, ISBN-13: 978-0-470-05747-6
- 5- Michael Juntao Yuan, " Nokia Smartphone Hacks", O'Reilly Media, Inc., USA, 2005, ISBN: 0-596-00961-5.
- 6- John W. Rittinghouse and James F.Ransome, "Cloud Computing: Implementation, Management, and Security", Taylor and Francis Group, LLC, USA, 2010, ISBN: 978-1-4398-0680-7
- 7- Jason Lengstorf and Phil Leggetter, " Realtime Web Apps: with HTML5 WebSocket, PHP, and jQuery",
- 8- Oswald Campesato and Kevin Nilson, "Web 2.0 Fundamentals: With AJAX, Development Tools, and Mobile Platforms", Jones and Bartlett, USA, 2011, ISBN-13: 978-0-7637-7973-3.
- 9- Pedro Teixeira, " Professional Node.js: Building Javascript Based Scalable Software", John Wiley & Sons, Inc, Indians, USA, 2013, ISBN: 978-1-118-18546-9

The IISTE is a pioneer in the Open-Access hosting service and academic event management. The aim of the firm is Accelerating Global Knowledge Sharing.

More information about the firm can be found on the homepage:  
<http://www.iiste.org>

## CALL FOR JOURNAL PAPERS

There are more than 30 peer-reviewed academic journals hosted under the hosting platform.

**Prospective authors of journals can find the submission instruction on the following page:** <http://www.iiste.org/journals/> All the journals articles are available online to the readers all over the world without financial, legal, or technical barriers other than those inseparable from gaining access to the internet itself. Paper version of the journals is also available upon request of readers and authors.

## MORE RESOURCES

Book publication information: <http://www.iiste.org/book/>

## IISTE Knowledge Sharing Partners

EBSCO, Index Copernicus, Ulrich's Periodicals Directory, JournalTOCS, PKP Open Archives Harvester, Bielefeld Academic Search Engine, Elektronische Zeitschriftenbibliothek EZB, Open J-Gate, OCLC WorldCat, Universe Digital Library, NewJour, Google Scholar

