TARTARUS: A Multi-Agent Platform for Bridging the Gap between Cyber and Physical Systems

(Demonstration)

Tushar Semwal[1], Nikhil S[2], Shashi Shekhar Jha[3],
Shivashankar B. Nair[4]
Department of Computer Science & Engineering
Indian Institute of Technology Guwahati
Guwahati-781039, India
{t.semwal[1], nikhils[2], j.shashi[3], sbnair[4]}@iitg.ernet.in

ABSTRACT

While there are many multi-agent platforms, one that can support the creation of a cyber system (comprising static and mobile soft agents), which in turn can control physical systems, seems grossly missing. In this demonstration, we introduce *Tartarus*, a Prolog based multi-agent platform that can cater to the creation, programming, cloning and termination of both static and mobile agents. *Tartarus* empowers developers to drive both the cyber and physical worlds thereby facilitating the creation and actual deployment of Cyber-Physical Systems (CPS) and Internet of Things (IoT) in the real world.

Keywords

Agent platforms, Multi-Agent Sytems, Cyber-Physical Systems (CPS), Robots

1. INTRODUCTION

For more than a decade, numerous Multi-Agent platforms and software tools have been developed and used for realizing Multi-Agent Systems (MAS) [14]. Except for a few such as Mobile-C [3, 4] which uses a Ch-interpreter to host C/C++ based agents onto embedded devices or JADEX [11] which combines XML and JAVA in order to create intelligent agents with easy binding to WSDL based web services, it is very rare that such platforms or tools are actually used in deploying multi-agent systems in the real world. By real world, we mean the 3-dimensional space where the cyber world intersects its physical counterpart using a communication channel, to culminate in a Cyber-Physical System (CPS) [12]. These systems consist of heterogeneous devices such as laptops, PCs, embedded boards, sensors, robots, cars, etc., connected together either via wired or wireless communication channels.

The distributed nature of a MAS adheres to the requirements of managing various processes within a CPS. An agent which forms the cyber component may sit on top of physical devices such as embedded hardware or robots and make them execute a set of tasks to achieve a goal. We introduce herein, *Tartarus* [13], a multi-agent platform, which, to the best of our knowledge, provides distinctive features that can make the cyber agents access and control the real physical devices with ease. Developed on top of SWI-Prolog [15], *Tartarus* not only provides for the development of static agents but also supports rapid development and deployment of agents with mobility viz. mobile agents. These mobile agents can migrate within a network of nodes and execute programs locally at the concerned node. In addition, *Tartarus* also supports multi-threading and thus caters to concurrent execution of tasks. It currently runs on both Windows and Linux operating systems and can be used in conjunction with Linux based embedded devices such as the Raspberry Pi, Intel® Galileo, etc.

Tartarus and its predecessor *Typhon* [10] have been used in a variety of applications ranging from an agent based Internet of Things (IoT) [6], synchronizing cooperative tasks among robots [7], creating partial green corridors for emergency services in a traffic [2], sharing and learning on-the-fly by a set of distributed nodes in a network [8], etc. These applications involve real time sensing of the environment and consequent decision making to deliver the relevant actuations to the physical side of the system.

2. FEATURES OF TARTARUS

Written in Prolog, *Tartarus*[†] inherently features all the plus points of this language. The platform, however has some salient features, few of which are listed below.

1. Multi-threading: Unlike its predecessor *Typhon*, *Tartarus* uses threaded execution of various processes that allows for concurrent deployment. Threads in *Tartarus* are inherently managed by its core engine, which in turn isolates the developers from the complexities related to mutual exclusion [9].

2. Heterogeneity: *Tartarus* supports a fair degree of heterogeneity in terms of the operating system (Windows/Linux) and the underlying hardware. Agents developed using *Tartarus* on Windows environment can thus communicate with similar agents running on a Linux environment thereby providing reasonable cross-platform communications.

[†]For the Demonstration Video visit:
`https://youtu.be/VeryfhtT5Tk`

Appears in: *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016), J. Thangarajah, K. Tuyls, C. Jonker, S. Marsella (eds.), May 9–13, 2016, Singapore.*

3. Hardware-in-the-loop: Unlike most of the available agent platforms, *Tartarus* can be ported on embedded boards that currently include the Raspberry Pi and Intel® Galileo development board. It also comes with an interface for the LEGO® MINDSTORMS® NXT robots. These special interfaces allow developers to access the control systems of the underlying hardware, thus making the job of developers easier.

4. Distributed Control: MASs are well suited in applications which require distributed computations and communications among heterogeneous entities. Agents within *Tartarus* can share information between nodes, migrate from one node to another (mobile agents) and execute the code at a destination node. Agents can even clone to exhibit a certain amount of parallelism within a network.

5. Payloads: Mobile agents can carry a set of programs or data as payload to be either executed on demand at the destination node or be downloaded for use by that node. *Tartarus* provides the necessary execution environment to the agents in order to execute a program based on their defined logic. Agents can also be programmed to shed (offload) or upload payloads during runtime. By modifying payloads, one may even attempt to alter the behaviour of the agent in runtime.

6. Security: Security in *Tartarus* is at the moment naïve and is implemented using a lock and key mechanism. Each *Tartarus* platform has a lock which could be an encrypted code or password. A mobile agent is allowed to enter a *Tartarus* node only if it has the associated key for the same. Better security aids could be developed to make the system secure.

7. Ease of Installation: Installing *Tartarus* is simple and does not require any intricate adjustments such as setting path variables as in the case of Java based platforms.

3. TARTARUS: SOME REAL WORLD APPLICATIONS

Tartarus and its predecessor, Typhon, have been used as a tool for devising various distributed mechanisms in a number of practical applications,[¶] some of which are:

1. A Mobile agent based framework for IoT[¶] [6]: An IoT is a network of heterogeneous devices such as control boards, sensors, actuator systems, etc., all of which work together to monitor, gather and analyse data and aid us make appropriate decisions. It constitutes a typical distributed system and hence provides a fitting environment for the use of agents in realising it. Mobile agents in *Typhon*, called Typhlets have been used to mitigate the problem of searching for remote resources, monitoring the temperature of an area, making the robot to follow a line, etc.

2. Synchronization of a Sequence of Tasks in a Network of Asynchronous Nodes[¶] [7]: A group of entities (such as a set of robots) may be required to perform a set of tasks in a synchronous manner. Synchronizing the movements of robots in a distributed networked system requires a considerable number of messages to be passed on among the participating entities. This results in a large consumption of bandwidth. It also entails problems during real implementation. In this work, a novel technique inspired by the self-organized synchronous behaviour of insect colonies is proposed. The technique uses mobile agents to achieve synchronization through stigmergic communication [1]. The mobile agents, each carrying programs for distinct tasks knit through the robots, communicate using stigmergy and make them perform the tasks in synchronism. They can also handle failures inherently and recover from them. A multi-robot synchronized dance scenario portrayed in the demonstration video[†] accentuates the use of *Tartarus* in controlling and synchronizing multiple entities.

3. Partial Green Corridors for Emergency Vehicles : A partial green corridor for emergency vehicles [2] is a far better proposition than a full one. Using agents, we demonstrated that the same is feasible and far more efficient. Mobile agents, carrying all information about the vehicle move ahead in a *Tartarus* based Digital Traffic Infrastructure Network (DTIN) to regulate the traffic signals so as to form a few kilometre-long corridor to enable the vehicle to rush through seamlessly. The forward moving corridor is constantly monitored and updated during the vehicle's journey to its destination. The application uses two kinds of mobile agents - a *Map agent* that updates the status of the traffic at each traffic signal and a *Path-finding agent* which creates and maintains the partial green corridor amidst the traffic. Whenever, an emergency vehicle such as an ambulance starts its traversal through the traffic, the partial green corridor created by the *Path-finding agent* aids the ambulance to reach the hospital via the least crowded route. The mechanism also ensures that the adjoining traffic is in no way adversely affected due to the creation of such partial green corridors.

4. *Tartarus* on Embedded Hardware: Currently *Tartarus* has been tested out to work on the Raspberry Pi and Intel® Galileo development boards. Running on Linux, *Tartarus* on such hardware can act as an active wireless node. Agents within these nodes can monitor and act based on the sensors and even send information to remote base stations as has been portrayed in the demonstration video[†].

4. CONCLUSIONS

Connecting the cyber world with the physical one is always an arduous task. The use of *Tartarus* alleviates this to quite an extent by empowering agents to actually sense, think and act in the real world. It is envisaged that by 2020, the number of connected devices across the world will reach 20.8 billion [5]. This will result in the creation of an ecosystem comprising different IoT/CPSs working either sequentially or concurrently. While one IoT may be used for weather forecasting and reporting, another could be working to ameliorate agriculture. With the hardware-in-the-loop feature, *Tartarus* will not only be able to handle issues in forecasting and agriculture but also use the derived information to monitor and drive physical devices such as sensors and actuators, used in these scenarios.

Acknowledgments

The authors acknowledge the work done by Manoj Bode and Vivek Singh in the building and testing phases of *Tartarus*.

[¶]For videos of some of the applications cited herein visit: `http://www.iitg.ernet.in/cse/robotics/?page_id=1936`

REFERENCES

[1] R. Beckers, O. E. Holland, and J.-L. Deneubourg. Fom local actions to global tasks: Stigmergy and collective robotics. In *Prerational Intelligence: Adaptive Behavior and Intelligent Systems Without Symbols and Logic, Volume 1, Volume 2 Prerational Intelligence: Interdisciplinary Perspectives on the Behavior of Natural and Artificial Systems, Volume 3*, pages 1008–1022. Springer, 2000.

[2] M. Bode, S. S. Jha, and S. B. Nair. A Mobile Agent based Autonomous Partial Green Corridor Discovery and Maintenance Mechanism for Emergency Services amidst Urban Traffic. In *The First International Conference on IoT in Urban Space*, pages 13–18, Rome, 2014. Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering.

[3] B. Chen, H. H. Cheng, and J. Palen. Mobile-C: A mobile agent platform for mobile C/C++ agents. *Software: Practice and Experience*, 36(15):1711–1733, 2006.

[4] B. Chen, H. H. Cheng, and J. Palen. Integrating mobile agent technology with multi-agent systems for distributed traffic detection and management systems. *Transportation Research Part C: Emerging Technologies*, 17(1):1 – 10, 2009.

[5] Gartner. Gartner says 6.4 billion connected "things" will be in use in 2016, up 30 percent from 2015, 2015. [Online; accessed 20-February-2016].

[6] W. Godfrey, S. S. Jha, and S. B. Nair. On a mobile agent framework for an internet of things. In *Communication Systems and Network Technologies (CSNT), 2013 International Conference on*, pages 345–350, April 2013.

[7] S. S. Jha, W. W. Godfrey, and S. B. Nair. Stigmergy-Based Synchronization of a Sequence of Tasks in a Network of Asynchronous Nodes. *Cybernetics and Systems*, 45(5):373–406, June 2014.

[8] S. S. Jha and S. B. Nair. On a Multi-agent Distributed Asynchronous Intelligence-Sharing and Learning Framework. *Transactions on Computational Collective Intelligence XVIII*, 9240:166–200, 2015.

[9] L. Lamport. The mutual exclusion problem: Part I: A theory of interprocess communication. *J. ACM*, 33(2):313–326, Apr. 1986.

[10] J. Matani and S. B. Nair. Typhon- A mobile agents framework for real world emulation in Prolog. In *Multi-disciplinary Trends in Artificial Intelligence*, pages 261–273. Springer, 2011.

[11] A. Pokahr, L. Braubach, and W. Lamersdorf. Jadex: A BDI reasoning engine. In *Multi-agent programming*, pages 149–174. Springer, 2005.

[12] R. R. Rajkumar, I. Lee, L. Sha, and J. Stankovic. Cyber-physical systems: The next computing revolution. In *Proceedings of the 47th Design Automation Conference*, DAC '10, pages 731–736, New York, NY, USA, 2010. ACM.

[13] T. Semwal, M. Bode, V. Singh, S. S. Jha, and S. B. Nair. Tartarus: A multi-agent platform for integrating cyber-physical systems and robots. In *Proceedings of the 2015 Conference on Advances In Robotics*, AIR '15, pages 20:1–20:6, New York, NY, USA, 2015. ACM.

[14] W. Van der Hoek and M. Wooldridge. Multi-agent systems. *Handbook of Knowledge Representation*, pages 887–928, 2008.

[15] J. Wielemaker, T. Schrijvers, M. Triska, and T. Lager. SWI-Prolog. *Theory and Practice of Logic Programming*, 12:67–96, 1 2012.