Computation over Worlds with Dynamic Structure

Eloi Pereira[1,3], Pedro Marques da Silva[3], Clemens Krainer[2], Raja Sengupta[1] and Christoph M. Kirsch[2]

[1] Department of Civil and Environmental Engineering, University of California at Berkeley, USA
eloi@berkeley.edu, sengupta@ce.berkeley.edu
[2] Department of Computer Sciences, University of Salzburg, Austria
ck@cs.uni-salzburg.at
[3] Research Center, Portuguese Air Force Academy, Portugal
posilva@academiafa.edu.pt

The Von Neumann machine [1] defines a "bridging" model between the world of software and the world of computing systems hardware [2], i.e. the "cyber" and the "physical" worlds. Many of today's cyber-physical computing systems are *interactive*, *mobile*, *networked*, and *ubiquitous*. Computation may be distributed in a myriad of heterogeneous hardware such as smartphones, tablets, unmanned vehicles, robots, embedded computers, or even in the cloud where the hardware is abstracted. These cyber-physical computing systems contrast with the pioneer computers such as the IBM mainframes or first PCs where *interaction* and *mobility* was very sparse. Does this trend towards *interaction* and *mobility* requires new bridging models and software architectures?

According to Von Neumann, computing systems are devices that "carry out instructions to perform calculations of a considerable order of complexity — e.g. to solve a non-linear partial differential equation". Solving a PDE is not a representative example of the kind of computation being executed on smartphones, autonomous vehicles, and wireless sensor networks. More than just performing calculations, the programs in these CPS compute upon information received from their environment (e.g. reacting to some sensory data) or from other programs. They can also influence the environment by means of actuation (e.g. steering a vehicle, grasping an object, etc.). If the smartphones, vehicles, and sensors are the "machines" to these CPS programs, then programs change their "machines" and the" machines" constrain the execution of their programs. The Von Neumann machine does not change structure as its program is executing. A smartphone network supporting a distributed program can change both phones and links.

We introduce the BigActor model [3] as a bridge between concurrent distributed programs and a physical computing environment with dynamic structure. The BigActor model combines the Actor model [4] for specifying concurrent programs with Robin Milner's Bigraphical Model [5] for the structure of the computing environment in terms of the location and connectivity of the computing machines. In contrast, to the Von Neumann model, the structure of the distributed machine is a first-class concept in the BigActor model.

BigActors are hosted by computing machines (e.g. a smartphone, an Unmanned Aerial Vehicle, etc.). They are capable of performing regular actor operations (compute, asynchronously send messages to other actors, and spawn new actors). They are also able interact with the world upon which they are being executed. They are able to observe the structure of the world (e.g. a BigActor hosted in a vehicle may observe the location of other vehicles or what is the network topology that connects them), request control actions to change it (e.g. a BigActor hosted in a vehicle may request the vehicles to move to another location), and migrate to other machines (e.g. a BigActor hosted in a vehicle that is short in fuel may migrate to another vehicle and proceed with the primary task).

Our first BigActor programming language is a Scala Domain Specific Language (DSL). The DSL is built upon the Scala Actors library[1]. The language can be tested in a simulation

[1]We are currently migrating to Akka Actors library since the Scala Actors library is being deprecated and Akka is becoming the *de facto* actors library for Scala.

environment where the dynamics of the structure (i.e. bigraph reactive system) is simulated using the bigraph model checker BigMC [6].

We are applying the BigActor model to the management and control of heterogeneous mobile robotic systems comprised of Unmanned Aerial Vehicles (UAVs), Autonomous Underwater Vehicles (AUVs), and Autonomous Surface Vehicles (ASV's), and other kind of sensors such as drifters [7]. This sub-field of robotics has a diversity of communication links and interesting structural dynamics in the computing machines. For example, an AUV has an excellent communication link with a UAV when it is located on the surface, but no such link when located under the water. When underwater the AUV would do better to route first to an ASV and via it to the UAV. The BigActor model is able to reflect the inter-dependent dynamics of location and connectivity to the computer programs managing the UAV, AUV, ASV coordination in the system. We believe that the BigActor model is a first step towards bridging programs running on computing systems with dynamic structure.

To address these kind of robotic systems we developed a BigActor distributed architecture that runs upon the Robot Operating System (ROS) middleware. Each vehicle runs ROS providing an API to fetch telemetry from the vehicles and to request control actions to be executed. Each vehicle broadcasts over the internet its own information about the structure of the world (i.e. location and connectivity of entities) and its knowledge about the mission state (i.e. the set of current tasks and their state). Each vehicle also uses the information received from other vehicles to update their own knowledge. This provides means for the information to "flood" over the network. The software architecture includes software abstractions that facilitate the integration of other kind of systems rather than autonomous vehicles. For instance, an Android smartphone can be easily integrated over the internet using ROSJava.

We tested the overall system's architecture this summer in the Portuguese Navy exercise"Rapid Environment Picture 2013" at Portimão, Portugal. The demonstration consisted of an environmental monitoring exercise using the *Alfa* Unmanned Aerial Vehicle (UAV) from the Research Center of the Portuguese Air Force Academy, two UAV Ground Control Stations, the *NRP Cassiopeia* Navy vessel and several GPS drifters which broadcast their position using an Automatic Identification System (AIS) transmitters. The Alfa UAV is a fixed-wing platform with 3m of wingspan and it is equipped with an autopilot, a PC-104 computing platform, a gimbaled optical camera, and a Automatic Identification System (AIS) receiver. The mission consisted of monitoring an hypothetical oil spill (represented using 100 kg of popcorn) and was motivated by the "Prestige" oil spill disaster in the coast of Spain. The overall mission was specified using the BigActor language. The Alfa UAV took-off from an airfield at Portimão under the control authority of a ground station at the airfield. The control authority was transitioned to another ground station located at the shore (handover manoeuvre). At this point the environmental monitoring mission started. The UAV moved to the location of the oil spill. The video of the optical camera was streamed to the ground station where a payload operator received the video and controlled the gimbal. After identifying the oil spill the ground station sent a message to the vessel in order to deploy the drifters over the spill (with the pupose of forecasting the behaviour of the oil spill). Using AIS information, the UAV received the locations of the drifters and used this information to plan its behaviour.

References

[1] J. Von Neumann, "First draft of a report on the edvac," *Annals of the History of Computing, IEEE*, vol. 15, no. 4, pp. 27–75, 1993.

[2] L. Valiant, "A bridging model for parallel computation," *Communications of the ACM*, vol. 33, no. 8, pp. 103–111, 1990.

[3] E. Pereira, C. Kirsch, R. Sengupta, and J. Borges de Sousa, "Bigactors - a model for structure-aware computation," in *4th International Conference on Cyber-Physical Systems*. ACM/IEEE, April 2013.

[4] G. Agha, *Actors: a model of concurrent computation in distributed systems*. Cambridge, MA, USA: MIT Press, 1986.

[5] R. Milner, *The Space and Motion of Communicating Agents*. Cambridge University Press, 2009.

[6] G. Perrone, S. Debois, and T. Hildebrandt, "A model checker for bigraphs," in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*. ACM, 2012, pp. 1320–1325.

[7] E. Pereira, C. Potiron, C. Kirsch, and R. Sengupta, "Modelling and controlling the structure of heterogeneous mobile robotic systems: A BigActor approach," in *to appear at the 2013 IEEE International Systems Conference (SysCon 2013) Proceedings*, Orlando, USA, Apr. 2013.