Copyright © 2015 IEEE

Paper presented at INDIN 2015 IEEE, 22-24 July, Cambridge, UK, 2015.

This material is posted here with the permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of ABB's products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to pubs-permission@ieee.org.

By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

Challenges and opportunities when introducing cloud computing into embedded systems

Daniel Hallmans
ABB AB / MDH
Ludvika/Västerås, Sweden
daniel.hallmans@se.abb.com

***Abstract* — Usage of cloud computing in a number of different business are growing fast. More and more of functions are moving into the cloud to take advantage of the advantages with cloud computing, for example scalability, resources on demand, cost based on usage etc. Embedded Systems has not yet taken the complete step into the cloud, except for on a higher level, for example data storage, user interface. In this paper we present a number of Challenges and Opportunities when introducing cloud computing into embedded systems. Today we see the possibility to move a complete soft real-time systems in to the cloud and with future development also hard real-time systems.**

Keywords—Embedded system, cloud computing, soft real-time, had real-time, challenges, opportunities

I. INTRODUCTION

Cloud computing are in use in more and more business around the world. There are a number of large well-known companies that are providing the services of computer resources in the cloud, for example Microsoft, Google and Amazon. They are providing users with a simple, flexible and cost effective way to get access to hardware, preinstalled operating systems, applications etc. Users gets the possibility to expand or decrees its need of different resources without having to install/purchase new hardware/software. The typical types of functions that are implemented in the cloud are software that depends on interchange data between users or other soft wares, storage of data, office applications, applications that needs to be accessed from anywhere in the world etc., Figure 1.

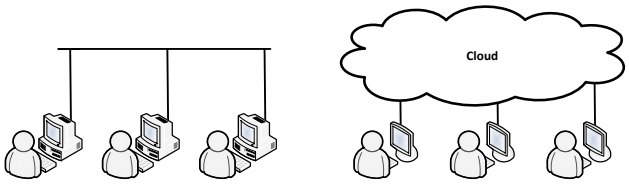

Figure 1: (left) Users with applications running on the PC, (right) users running applications in the cloud

Traditional embedded systems, like a control system for a power plant or a factory, has used different types of programmable logical controllers (PLC) or other embedded types of devices for a long time now. The advantages with cloud computing (sharing of resources, resources on demand, hardware as a service, cost based on usage etc) has not yet fully been introduced to this market except for on a high level to allow for data storage and user interface (UI).

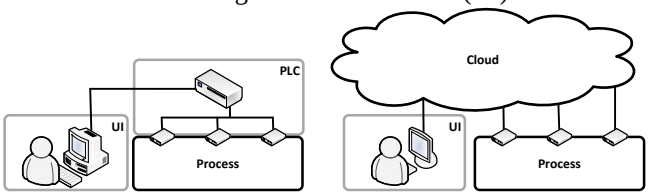

Figure 2: (left) Traditional PLC implementation, (right) all control loops has been moved to a cloud service

In this paper we have gathered different Opportunities and Challenges you will face when combining an embedded control system and a cloud, Figure 2, by moving the control loops to the cloud and only keep the sensors and actuators close to the process.

The outline of this paper is as follows: In section II we give a background to cloud computing. Section III we introduces the concept of embedded systems in a cloud. Section IV describes the Opportunities and Section V the Challenges. Section VI describes the related work and in section VII we present our Future work. In section VIII we summarize our conclusion.

II. CLOUD COMPUTING OPPORTUNITIES

Cloud computing in its simplest form is when a large amount of remote servers are connected together to allow for storage, networking, development and deploying platforms as well as business processes. National Institute of Standards and Technology defined cloud computing as "*a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction*". Today clouds comes in different deployment models dependent on the need by the user, from private clouds, only available for the user and typically located inside a organizations network, and public clouds open for

access for all users. Combinations of these, called hybrid clouds, can be used by organizations to create business models with internal, from organization, and external, customers/organization, access. Clouds are delivered in the following three different ways:

A. Infrastructure as a service (IaaS)

IaaS is a delivery model that gives the user access to hardware, storage and networks on request. If the user does not use the specific parts any longer they are simply de-provisioned and made available for other users instead. On top of the supplied hardware the user can add their own software to create the needed applications. One example of an IaaS is Amazon Web Services [1].

B. Platform as a Service (PaaS)

The next type of delivery model is PaaS and it can simplified be explained as an IaaS with added base functionalities (operating systems, development tools, etc) to give the user a well-tested and integrated environment to deploy its applications in to the cloud. Typical examples of PaaS are Windows Azure [2] and Google App Engine.

C. Software as a Service (SaaS)

A SaaS sits on top of a PaaS and IaaS and gives the user a predefined set of software services. Examples of SaaS are Google Apps and Dropbox.

In common for all versions are that they are built on resource pooling to keep cost as low as possible but at the same time make it possible to rapidly on demand change the number of resources needed. They are based on a broad network access since everything is accessed through the network (local or internet). The flexibility of a cloud system allows for a cost model based on the measured amount of used resources.

III. EMBEDDED SYSTEMS AND CLOUDE COMPUTING

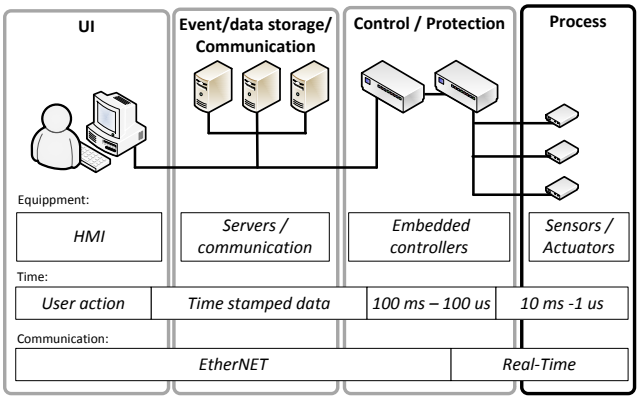

Figure 3: Embedded system with four different levels

Traditionally embedded systems consist of four different levels, Figure 3. Sensors and actuators are located close to the process, Next level contains control and protection devices containing all control and protection loops/functions based on hard or soft real-time requirements. The last to levels are combing Event handling (to store the history), data storage (store process data), high level communication (for example between control and remote control center) and user interface (allowing the user to supervise and control the process). Together with the process the functions in the four different levels will create a closed loop control with timing requirements changing from highly timing dependent close to the process (sensor/actuators) with higher requirements on low latency to more and more focus on throughput and time stamped data the closer it gets to the user level.

The definition of an embedded system range from a very simple systems, for example a control for a coffee machine, to a complex distributed systems used for controlling a power plant. Introducing cloud computing in to these varieties of systems can be done in many different ways from just using the cloud to store data and interact with users to move the complete implementation, except for needed physical parts like sensors and actuators, to the cloud.

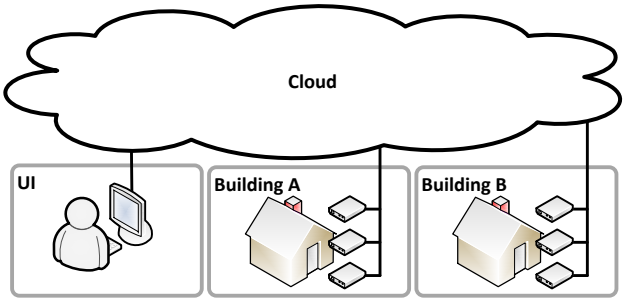

Figure 4: Example of a cloud based soft real-time system with sensors and actuators connected to a cloud

An example of a cloud based implementation can be to move the control logic for a building security system, Figure 4, to the cloud. Left in building would then be sensors (key-card readers, cameras, intrusion detection systems, fire alarms etc.) and actuators (door locks, alarms, motorized doors etc.). All of the sensors and actuators would communicate over different types of encrypted communication media (based on for example radio communication, Zigbee, Wifi, cabel etc) directly with the control logic located in the cloud. The requirements on the control system is this case is more soft real-time than hard real-time, a delay up to a second for a "round trip" from that a key card is used until a door is opened would be OK for a user. Moving all of the control to the cloud servers not only simplifies the installation on site, less hardware to install, it also allows for fast changes of functionality as long as no new sensors/actuators has to be installed. The supervision of the building can also be done in any part of the world, for example could the night watch be handle by people located on the other side of the earth, in a similar way as many European news departments have people located in Australia to be able to handle the news flow during night time in Europe. The price tag from the security company to the end customer can in this case depend on the usage of functions since the security company also will pay the cloud supplier dependent on the usage.

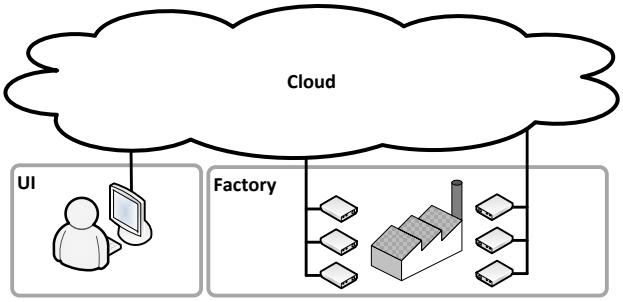

Figure 5: A hard real-time implementation of a cloud based control systm

A similar example as the cloud security implementation but based on a system with hard real-time requirements could be a cloud implementation of an embedded system from the process industry, Figure 5. In this case there will be a more firm requirement on timing and also strict requirements on safety certifications. Sensors and actuators would typically be placed in a Factory environment controlling a process that does not allow for much jitter or long delays. Waiting for one second until the process stops due to a fault is probably not OK.

IV. OPPORTUNITES

Implementing parts of the embedded system in the cloud gives a number of opportunities:

1) *Resource sharing:*
   - Resources can be shared between different systems dependent on the need for the moment, for example can different functions that are not connected to the embedded system but located in the same cloud share the same resources with not time critical control functions. An example of this could be that e-mail, document handling or economical calculations are using the resources during daytime and the same resources are used for trend analysis based on stored data from the process during nights when the office is closed.
   - Big data storage can be available for the system when it is needed, for example in case of a fault, but there is no to buy additional hardware.
   - If resources can be shared with other applications or rapidly be increased the control system could use a minimized amount of resources in "steady state" and if something unexpected (or unwanted) happens additional resources are used (running more advanced closed loop regulation functions during a fault stage, higher need of data storage during a short time period).
   - Resources in the cloud will be functional independent giving the possibility for having one system as redundant standby for more than one active system.

2) *User interaction:*
   - Interaction with users are scalable dependent on need and a cost model could be used based on how data is used.

3) *Data sharing*
   - With all information from the control system stored in a cloud it allows for sharing the information with other actors on the market. It would for example give an opportunity for third party companies to sell functionality based on the information.

4) *Software updates*
   - A company with system distributed on a larger site or worldwide can in a simpler way get access to all there systems for maintenance and updates.

5) *Life cycle*
   - The hardware independence of the installed virtual machines gives a possibility to handle end of life of hardware components in a simpler way as long as they can be supported by the virtual machines. If there is an end of life of the software functions you will have similar problems as today with for example operating systems.

6) *Provider Independence*
   - If the cloud in the future could be seen as a provider independent platform the user would have a possibility to freely choose between different suppliers of hardware platforms. This will change the business for embedded system developers from creating a function by designing hardware and software to only supply a function based on software.

V. CHALLENGES

Changing to the cloud for an embedded system also introduces a number of challenges that has to be solved.

1) *Governance and security*
   - Moving the complete control functionality for a power station to a cloud located on a server owned by a third party organization will put a number of constraints from a security (Physical security, Human resource security, Business continuity, disaster recovery, Identification and access management, Encryption etc.) and governance (can data be shared across borders, specific performance requirements, do I get access to resources when I need them? etc.) perspective. Form a cloud perspective this is not anything new, similar requirements can be found from other companies, for example banks, retail industry etc., since they also need to relay on a high degree of availability of the service not to lose any income. A difference could be that some of the

control systems, for example a transmission and distribution plant, are that they also are seen as critical infrastructures and has specific requirements from the Government on security, one of such example are NERC-CIP in North America [3], that will require extra caution when designing the system. An example of an cyber security incident that possible would have been easier to achieve if all control functions were located in a common cloud would be the Stuxnet [4] incident that was aiming to infected a control system located on a specific location and did not harm any other systems.

2) Timing

- A hard real time system is dependent on fulfilling a number of very strict timing requirements, for example round trip delays ranging from a few microseconds to a number of milliseconds dependent on the application. Deploying the same application on a cloud server would put a number of similar constraints on the cloud system to fulfill. If resources at the same time are share with other applications it could be almost impossible to handle the shortest time intervals, when also including communication delays, risk of other applications running in the same server to affect the performance (usage of memory bandwidth, cache usage, Ethernet communication etc). For systems with not that extreme timing considerations it could still be doable.

3) Communication

- More and more of sensors and actuators are using an Ethernet protocol for communication and this opens up for moving applications to a cloud server and still be able to communicate with sensors/actuators. The main problem arises when control system need to use different real-time network communication protocols, for example different industrial Ethernet networks, [5][6] or other types of communication protocols that does not directly allows to be routed over a network. Instead data has to be embedded in routable packages via a gateway without lowering the performance.
- Adding more levels of communication, initially only between sensor and control located in the same physical location to a cloud system located somewhere on the internet, will decrease the mean time between failures if not a redundancy concept is used. Loss of communication to the cloud has to be handled in the same way as if a communication bus goes down in a system today. A similar “alive”/”heart beat” as implemented in systems today has to be reused for the actuators to be able to act if control is lost.

4) Redundancy/control-/protection-system

- A large part of the systems used in industry today are designed based on some kind of a redundancy concept to get a higher availability, for example a two system solution with one active system and one standby system. A cloud system has to be able to give the same type of availability and also a fast switching between the redundant systems. Other requirements could be to have different physical instances for the control loops compared to the functions implemented to protect the system if anything goes wrong. In this case the protection will supervise the control actions taken in a similar way as for the redundancy concept.

5) Safety certification

- Is it possible to certify a safety application that is running in a cloud? Today industry and research are struggling with certifying multicore and reuse of software. In the case of cloud computing we have to certify a code that is running on an unknown hardware together with other unknown software, will it be possible? There are operating systems today that are certified for running on a single hardware can that type of certify to run on an unknown hardware? Will there be limitations on what type of hardware and communication that can be used for a certified process?

VI. RELATED WORK

There are a lot of research ongoing around cloud computing and how it could be used in different situations. Only a little part of this research are focusing on using cloud computing for real-time applications, one example is around designing soft PLC placed in the cloud, [7], where they also tested the performance of an soft PLC in a case study. [8] gives an comprehensive overview of cloud computing in industrial systems. [9] describes a three level approach (Device level, communication level and cloud service level) to introduce intelligent vehicle service with cloud computing technics into cars. A similar layered system intendant for use in a carpool service is presented in [10]. [11][12] brings up some of the issues and solutions on real-time services in cloud computing. Security and cloud are brought up in a number of different research papers, understandable since most clouds are connected to the internet, examples that also focusing on control and real-time are [13] [14]. Reliability and fault tolerance is also brought up as a possible issue with different proposed solutions, [15][16][17].

VII. FUTURE WORK

We see a large possibility with moving different types of embedded systems to a cloud environment, already today a soft real-time system could be moved without too much problems and in the future also hard real-time functions, as process industry control systems. We are continuing our research by proposing a solution for hard real-time systems based on a hybrid between a cloud system and a more classic embedded system.

A. *Prototyped cloud implementation*

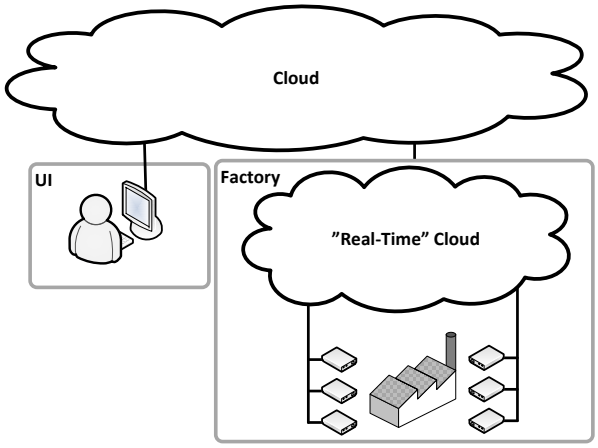

Figure 6: A Three level cloud control implementation

To be able to take advantage of the possibilities of cloud computing in a timing critical system/ hard real-time system we propose a solution with a system divided in to three levels, Figure 6 and Figure 7:

1. One "classic" specialized embedded system close to the process, based on FPGAs, DSPs etc. technology allowing for the fast time critical timing loops.
2. A cloud based "Real-time system" based on a hard-real-time hypervisor. A future alternative open source alternative could be RT-Xen [18] that today is more soft real-time. Scheduling could be done using for example Giotto or Ptides [19][20] that will give a control based on time stamped data. Between the level one and the "real-time cloud" we could create a "rubber band" to be able to handle temporarily delays and jitter in the system. This would allow for the cloud system to catch up with the process if it falls behind, due to for example a temporary network over load. New concepts has to be designed for redundancy, separation of functions (control and protection loops), safety and catch up functionality (for example by fast reallocation of cloud resources). An idea could be to use a more timing correct server implementation based on for example a PRET architecture, [21].
3. And finally a more "classical" cloud for data storage, event handling, user interfaces and high level communication with other systems.

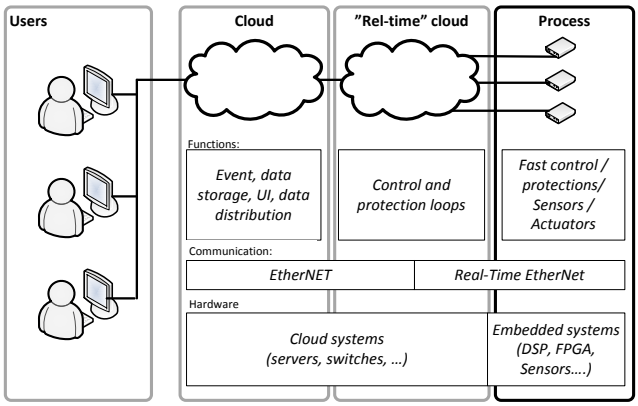

Figure 7: Proposed "leveled" design

VIII. CONCLUSION

All types of embedded systems can today without a too high effort be interfaced with cloud systems on a high abstraction level, to allow for interchanging of data, storage of data, user interface etc, typical non time critical functions. A soft real-time system could also place some or all of its control loops in a cloud environment. From a user and developer perspective this opens up a number of possibilities with resource sharing, simple expansion, hardware independence, easy access, etc. but also risks from a security and governance perspective. Moving the complete set of functions for a hard real-time system to the cloud puts a number of more constraints on timing (for example real-time communication, closed loop period time and jitter etc.) that could be hard to achieve today if not a combination of traditional embedded systems (for the fastest control loops) and cloud based technologies (based on time stamped information) are used in a heterogeneous combination.

IX. ACKNOWLEDGMENT

The research presented in this paper is supported by the Knowledge Foundation (KKS) through ITS-EASY, an industrial Research School in Embedded Software and Systems, affiliated with the school of Innovation, Design and Engineering (IDT) at Mälardalen University (MDH), Sweden.

REFERENCES

[1] "Amazon Web Services (AWS) - Cloud Computing Services." [Online]. Available: http://aws.amazon.com/. [Accessed: 30-Nov-2014].

[2] "Azure: Microsoft's Cloud Platform." [Online]. Available: http://azure.microsoft.com/en-us/. [Accessed: 30-Nov-2014].

[3] Chee-Wooi Ten, M. Govindarasu, and Chen-Ching Liu, “Cybersecurity for electric power control and automation systems,” in *2007 IEEE International Conference on Systems, Man and Cybernetics*, 2007, pp. 29–34.

[4] R. Langner, “Stuxnet: Dissecting a Cyberwarfare Weapon,” *Secur. Priv.*, vol. 9, no. 3, pp. 49–51, 2011.

[5] D. Jansen H|Buttner, “Real-time Ethernet: the EtherCAT solution,” *Comput. Control Eng. J.*, vol. 15, no. 1, p. 16, 2004.

[6] J. Feld, “PROFINET - scalable factory communication for all applications,” *Factory Communication Systems, 2004. Proceedings. 2004 IEEE International Workshop on*. pp. 33–38, 2004.

[7] O. Givehchi, J. Imtiaz, H. Trsek, and J. Jasperneite, “Control-as-a-service from the cloud: A case study for using virtualized PLCs,” in *2014 10th IEEE Workshop on Factory Communication Systems (WFCS 2014)*, 2014, pp. 1–4.

[8] O. Givehchi, H. Trsek, and J. Jasperneite, “Cloud computing for industrial automation systems — A comprehensive overview,” in *2013 IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA)*, 2013, pp. 1–4.

[9] J. Wang, J. Cho, S. Lee, and T. Ma, “Real time services for future cloud computing enabled vehicle networks,” in *2011 International Conference on Wireless Communications and Signal Processing (WCSP)*, 2011, pp. 1–5.

[10] Chih-Hsiang Lin, Ming-Kai Jiau, and Shih-Chia Huang, “A cloud computing framework for real-time carpooling services.” pp. 266–271, 2012.

[11] S. Liu, G. Quan, and S. Ren, “On-Line Scheduling of Real-Time Services for Cloud Computing,” in *2010 6th World Congress on Services*, 2010, pp. 459–464.

[12] W.-T. Tsai, Q. Shao, X. Sun, and J. Elston, “Real-Time Service-Oriented Cloud Computing,” in *2010 6th World Congress on Services*, 2010, pp. 473–478.

[13] A. O. Joseph and G. J. W. Kathrine, “Security of real time cloud service providers: A survey,” in *2014 International Conference on Electronics and Communication Systems (ICECS)*, 2014, pp. 1–5.

[14] C. Huo, T.-C. Chien, and P. H. Chou, “Middleware for IoT-Cloud Integration Across Application Domains,” *IEEE Des. Test*, vol. 31, no. 3, pp. 21–31, Jun. 2014.

[15] K. An, “Strategies for Reliable, Cloud-Based Distributed Real-Time and Embedded Systems,” in *2012 IEEE 31st Symposium on Reliable Distributed Systems*, 2012, pp. 483–484.

[16] A. Ganesh, M. Sandhya, and S. Shankar, “A study on fault tolerance methods in Cloud Computing,” in *2014 IEEE International Advance Computing Conference (IACC)*, 2014, pp. 844–849.

[17] S. Malik and F. Huet, “Adaptive Fault Tolerance in Real Time Cloud Computing,” in *2011 IEEE World Congress on Services*, 2011, pp. 280–287.

[18] Sisu Xi, J. Wilson, Chenyang Lu, and C. Gill, “RT-Xen: Towards real-time hypervisor scheduling in Xen.” pp. 39–48, 2011.

[19] Y. Zhao, J. Liu, and E. A. Lee, “A Programming Model for Time-Synchronized Distributed Real-Time Systems,” in *13th IEEE Real Time and Embedded Technology and Applications Symposium (RTAS’07)*, 2007, pp. 259–268.

[20] T. A. Henzinger, B. Horowitz, and C. M. Kirsch, “Giotto: a time-triggered language for embedded programming,” *Proc. IEEE*, vol. 91, no. 1, pp. 84–99, Jan. 2003.

[21] I. Liu, J. Reineke, and E. A. Lee, “A PRET architecture supporting concurrent programs with composable timing properties,” in *2010 Conference Record of the Forty Fourth Asilomar Conference on Signals, Systems and Computers*, 2010, pp. 2111–2115.