Replication of IEC 61499 Applications

(PhD Thesis Proposal for the MAPI program)

Thematic Areas: Industrial Informatics, Distributed Systems, Real-Time Systems

Supervisor: Prof. Mario de Sousa (University of Porto, INESC TEC)

External Researcher: Prof. Luis Miguel Pinho (ISEP)

Summary

PLCs (Programmable Logical Controllers) are embedded computers built specifically for the industrial environment, and used for the automation of industrial processes. These systems are typically programmed using programming languages defined in the IEC 61131-3 standard (this includes 2 textual and 3 graphical programming languages). IEC 61131-3 however only considers PLCs working in isolation. Consequently some vendors are starting to support the IEC 61499 standard which defines a framework for developing distributed control applications.

The use of distributed applications means the application may fail in new ways which were not previously possible with IEC 61131-3. The IEC 61499 standard does not however explicitly define how timing requirements may be specified and enforced, neither does it provide mechanisms for the development of reliable control applications. This thesis will study how IEC 61499 may take advantage of its distributed nature to increase application dependability, while simultaneously providing real-time guarantees which are fundamental for the typical applications of the industrial automation domain.

The IEC 61499 Standard

The IEC 61499 defines a framework for distributed applications that span multiple networked devices. Applications are composed by a network of function blocks (FB), where information is transferred through events and data passing. Basic FBs are defined by the data and algorithms they contain (and not a FB network), as well as an execution state chart (ECC) that defines when each algorithm will be executed. The FBs are executed by a 'resource' (e.g. processor/CPU) which, according to the standard, may use any scheduling algorithm. The IEC 61499 standard does not (A) provide mechanisms for the development of reliable control applications, neither does it explicitly define how (B) the application timing requirements may be specified and enforced. This in spite of the fact that in the industrial automation domain both high reliability and real-time constraints are of fundamental importance.

High Dependability for IEC 61499 Applications

To achieve high dependability (essential for safety-critical systems) the IEC 61508 standard is usually adopted. This standard suggests which methodologies should be applied at each phase of software development cycle in order to obtain reliable software. At the requirements analysis and design phases the methodologies suggested for SIL 3 and 4 (SIL – Safety Integrity Level) are mostly formal and semi-formal methods. How several of the suggested methods may be applied within the framework defined by IEC 61499 have already been the object of research by several authors.

However, current formal method approaches to model IEC 61499 applications are severely handicapped by the fact that the IEC 61499 standard itself does not fully specify an execution model, allowing 7 distinct execution models for a single resource [1]. This means that each method is assuming a different execution model, and that, given a physical implementation of IEC 61499, not all formal modeling techniques may be used. Additionally, the formal modeling tries to address the issue of correctness of the IEC 6499 application, but not its availability (i.e. is the application running at all), which is also an important facet of achieving dependability.

In this thesis the intention will not be to use design time formal modeling techniques to achieve software reliability, but rather to cover an area in which very little work has been done. More precisely, the thesis will focus on applying to the IEC 61499 framework methodologies used during the design and implementation phases of software development – in particular those which allow the increase of application availability.

As with the formal modeling, the methodologies themselves have been extensively researched in the context of computer science, and for this thesis we will be exploring the use of fault tolerance methods. The closest work done in this regard was by [2]. Here, a method of allowing a running IEC 61499 application to be dynamically re-configured is proposed. The intention is to reduce down time by using cold or warm standby spares, which is not always acceptable for hard real-time applications with low latency requirements.

For this thesis an approach of tolerating faults through the use of software component replication will be studied. Due to the real-time requirements, even short down-times are not acceptable, which is why the focus will be on the active replication approach. However, instead of replicating all hardware, we propose that the replication be done at the software level; some software modules are replicated in order to achieve the desired reliability, while other less critical modules execute a single copy. Replicated modules may be placed on any hardware device the system designer chooses. This approach has better resource efficiency, but leads to a system configuration where hardware devices execute dissimilar sets of modules.

The main difficulty arises when the replicated modules contain internal state, in which case all replicas must be guaranteed to maintain a synchronized internal state (i.e. replica determinism), while simultaneously providing hard-real time guarantees. To achieve replica determinism, all replicas must be deterministic, and process all input events and data in the same causal and total order. Traditionally this is done by either (i) defining (off-line) a predetermined execution schedule to which all replicas must adhere to, or (ii) have all replicas agree (during execution) on every scheduling decision using distributed agreement protocols.

The first approach implies a static schedule which does not map favorably to the IEC 61499 framework of event passing, while the second relies on the use of reliable total order broadcast protocols that introduce considerable overhead. The exact protocol to use will depend on the failure models and failure semantics of the processors executing the application. Additionally, the data and events coming out of the several replicas of a replicated FB need to be voted upon and become a single data/event point. The voting algorithm to use will once again depend on the exact execution semantics, as well as the failure semantics being considered.

One goal of this thesis is to study alternatives that meet the desired requirements. One possibility might be the timed messages [3] approach. However, even timed messages may have limitations that might have to be worked around in order to work in an IEC 61499 context.

It is too much to ask of an application developer to be aware of all these details. For this reason, the second goal will be to define a framework that will enable a control application developer to design and implement his application with little to no regard to the replication requirements of the application. Only during the installation phase will the developer define which modules of the control application need to be replicated, and the framework and services will handle the details (such as choosing the best communication protocol) almost seamlessly.

Supporting Analysis Methods

Methods to formaly analyse the above framework will need to be developed, in order to provide both (C1) timing guarantees as well as to (C2) quantify the dependability parameters of the resulting applications.

To achieve (C2), one approach that may be explored is to use the TNCES/SNS (Time Net Condition/Event Systems – Signal/Net Systems) approach, i.e. by extending the previous models developed in [4][5][6]. Two new areas will need to be modelled: the communcation protocols used by the replication framework, and the stub FBs used to isolate the application designer from the fact that the application has been replicated.

To achieve (C1), we will extend the formal real-time calculus method of timing analysis so as to apply it to the framework. This method of timing analysis has never been used for IEC 61499 applications, although it seems to be a good fit, as it was developed with distributed real-time applications in mind.

References

[1] L. Ferrarini and C. Veber, “Implementation approaches for the execution model of IEC 61499 applications”, in Proc. 2nd Intl. Conf. on Industrial Informatics (INDIN’04), Berlin, Germany, 2004, pp 612–617.

[2] “Towards Zero-downtime Evolution of Distributed Control Applications via Evolution Control based on IEC 61499”, (ETFA 2006)

[3] S. Poledna, A. Burns, A. Wellings, P. Barret, Replica Determinism and Flexible Scheduling in Hard Real-Time Dependable Systems, IEEE Transactions on Computers, 49(2):100-111, 2000

[4] V. Vyatkin and H.-M. Hanisch, “A Modeling approach for Verification of IEC 61499 Function Blocks using Net Condition/Event Systems,” in Proc. IEEE Conf. On Emerging Technologies in Factory Automation (ETFA’99), Barcelona, Spain, 1999, pp. 261–270.

[5] V. Vyatkin and H.-M. Hanisch, “Modeling of IEC 61499 function blocks – a clue to their verification,” in Proc. XI Workshop on Supervising and Diagnostics of Machining Systems, Wroclaw, 2000.

[6] V. Vyatkin and H.-M. Hanisch, “Software Environment for Automated Verification of Distributed Industrial Controllers Following IEC611499,” in Proc. XII. Workshop on Supervising and Diagnostics of Machining Systems, Karpacz, Poland, 2001, pp. 62–72.