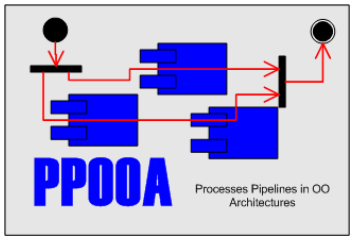

PPOOA, An Architectural Style for Real Time Systems

José Luis Fernández Sánchez
Industrial Engineering School
Universidad Politécnica de Madrid
e-mail: fernandezjl@acm.org
September 2004

Abstract

PPOOA, Process Pipelines in Object oriented Architectures, is a software architecture style for concurrent real-time systems. PPOOA proposes a vocabulary of components and coordination mechanisms to be used in software architecture design. UML metamodel was extended to include PPOOA building elements. PPOOA is implemented in CARTS, a prototype of architecting CASE tool and Microsoft Visio, a commercial CASE tool. PPOOA can be implemented in any CASE tool supporting UML extensibility mechanisms.

The PPOOA architecting process, named also PPOOA_AP should be included as a part of the general software development process. As a part of this general process, it takes some inputs from the analysis phase and produces some outputs to the detailed design phase. PPOOA architecting process may be applied in either a waterfall software development process or iterative development process such as the Unified Process.

The main purpose of an architecting process is to produce a rigorous description of the solution, allowing quality attributes evaluation by quantitative and qualitative methods. Responsiveness is a critical quality attribute of real-time systems. It is related to the response time to stimuli (events) either external or internal to the system. Rate Monotonic Analysis (RMA) is an analytical method that eases the assessment of system responsiveness. It does so through a collection of quantitative techniques and algorithms that let engineers understand, analyse and predict the timing behaviour of their designs, mainly in terms of response times.

Contents

1. Introduction

A real-time system is one in which correctness depends on meeting time constraints. In these systems, correctness arguments must reason about functional requirements and response time requirements.

Real-time systems often have to deal with multiple independent streams of input events and produce multiple outputs. These events have arrival rates often unpredictable, although they should be processed meeting timing constraints specified by software requirements.

Object oriented methodologies and early architectural decomposition efforts traditionally focus on domain analysis and functional cohesion to derive software components.

The experiences in defining time constrained architectures, let us to consider that timing concerns must play an important role in the choice of system components and objects. Concurrency modelling and its accompanying coordination (synchronization and/or communication) behaviour become a dominant concern surprisingly early in the architecture development process whenever response time is a critical factor.

Current CASE tools are supporting UML as the universal notation for object oriented software development.

This general applicability is the cause of some limitations that difficult UML use in modelling real-time component based software architectures.

The main limitations of UML regarding the real-time systems architecture modelling are:

- Not explicit support of the "design component" as a UML building element. The component concept in UML is considered from the implementation perspective.
- The Notation used for behaviour description, for example sequence and collaboration diagrams, is not suitable for responsiveness assessment techniques, such as Rate Monotonic Analysis (RMA). The reason is their emphasis in interactions or messaging instead of activities or actions, as they are called in RMA.
- Some building elements necessary in real-time architectures such as aperiodic processes or coordination (synchronization and/or communication) mechanisms are not considered as part of UML metamodel.

The author in his doctoral thesis report proposed a new style for concurrent object oriented architectures. The style can be used in object oriented systems when individual paths of execution are required to be concurrent and multiple processes may be positioned along the path to control the action (Figure 1). Therefore each process controls a segment of the path sequentially. So he called the style PPOOA, "Pipelines of Processes in Object Oriented Architectures".

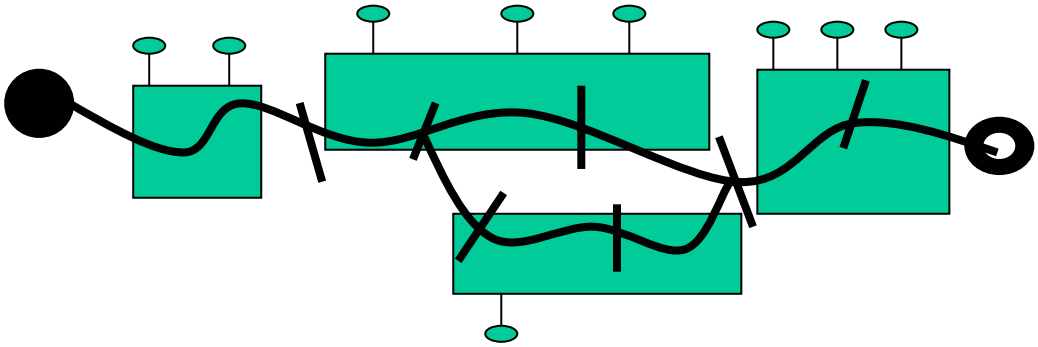

Figure 1. Pipelines of Processes

PPOOA promotes that architectural decisions related to concurrency in real-time systems should be made at the preliminary design or architecture phase of the development cycle.

Currently UML is the standard notation for object-oriented development. An important issue of UML is its capacity to be extended by using the profiling concept that allows the compatibility with other UML modelling issues. A UML profile is defined as a stereotype of package that groups model elements that have been customized for a specific domain. It allows one to develop variants of the UML suited for such specific domain.

The PPOOA profile is compatible with UML and improves its use in real-time architectures. The UML stereotypes are extended with the elements of the style (periodic and aperiodic processes, controller object, and coordination mechanisms). Activity diagrams are also adapted for PPOOA and RMA requirements, specifically modelling resources and considering scheduling points.

2. PPOOA Evolution

PPOOA is the result of 10 years of research

It began with a taxonomy of coordination mechanisms for real-time systems developed by the author at the SEI. · "A Taxonomy of Coordination Mechanisms Used in Real-Time Software Based on Domain Analysis". CMU/SEI-93-TR-34. Software Engineering Institute.Carnegie Mellon University. December 1993.

The author in his doctoral thesis report presented at Madrid Technical University (UPM) in 1997 proposed an architectural style for concurrent object oriented architectures.

The seminal paper about PPOOA was published in 1998. " An Architectural Style for Object-Oriented Real-Time Systems", 5th International Conference on Software Reuse.Victoria (Canada). IEEE 1998. This paper describes the style emphasizing the usage of coordination mechanisms and the architecting guidelines specified for the style.

PPOOA was developed before UML standard. So it did not use UML notation.

As UML popularity increased, the author realized the importance of using UML notation.

So, partially funded by the European Union CARTS project, we developed a UML profile for real-time systems based on PPOOA, and also and architecting process named PPOOA_AP. PPOOA and PPOOA_AP were validated in real-time systems developed by the industrial partners of CARTS.

Finally PPOOA has been implemented in Microsoft Visio. This is a general design tool that provides mechanisms for implementing diverse engineering methods. This implementation offers the benefits of commercial CASE tools that already support UML notation.

The results of this work are published in project reports, a book chapter and several public papers.

3. PPOOA Strengths and Weaknesses

PPOOA is a more specific architectural style to be applied to pipelined real-time systems. Typical applications are Air Traffic Control systems and Supervisory Control and Data Acquisition systems (Figure 2). PPOOA emphasizes the identification of coordination mechanisms because they are main contributor to blocking situations. PPOOA uses UML activity diagrams in a particular way to describe causal flow of activities (CFA), more or less an instance of a white box use case (Figure 3 at the end of the white paper).

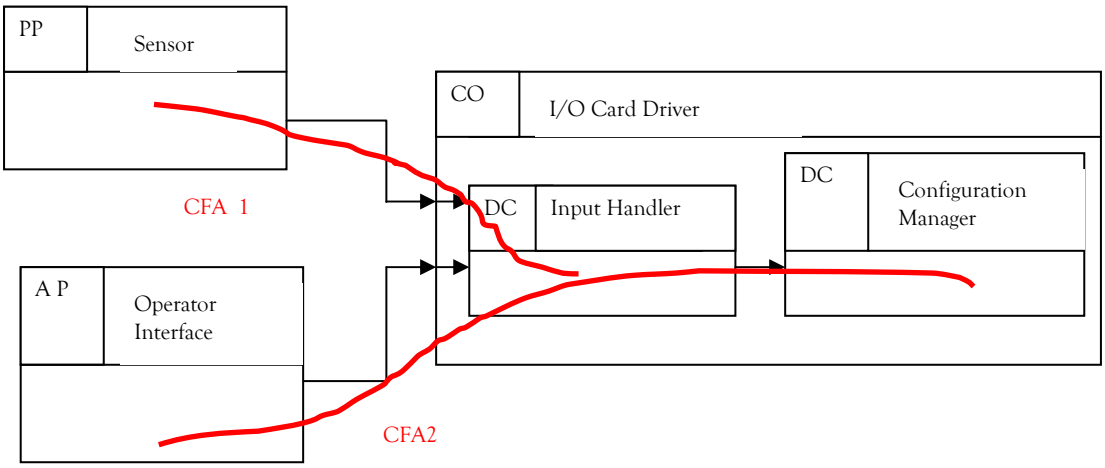

Figure 2. SCADA Example, (Intentionally incomplete)

This approach allows seamless application of RMA time responsiveness assessment techniques (table 1). PPOOA is complemented with architecting guidelines and an architecting process called PPOOA_AP. So, PPOOA applicability is one of its main strengths.

The strengths of the combined use of PPOOA architectural modelling and RMA are:

- Accurately dimensioning the system architecture at preliminary design time
- Eases the activities allocation to the components of the architecture
- Facilitates trade-offs and the evaluation of design alternatives
- Obtaining all required timing information during RMA assessment
- Saving testing time and avoiding typical real-time problems (priority inversion, unbounded blocking, deadlock situations)

Table 1 describes the combined application of PPOOA modelling and RMA during the software development lifecycle.

Table 1. Assessment techniques (RMA) and the Software Life Cycle

Lifecycle Phase	*Requirements Analysis*	*Architecture or Preliminary Design*	*Detailed Design*	*Implementation*
Models	Conceptual Model	Architectural Model (PPOOA)	Resources,	
Assessment techniques application		Responsiveness of CFAs evaluated	Network delays Detailed evaluation of blocking conditions	Actual Measurements for activities times

PPOOA weaknesses are related to its goal to deal with pipelined software architectures. Other real time architectural styles as timeline (cyclic executive) are not well modelled using PPOOA building elements.

PPOOA is best suited to model intranode architectures. In order to model extranode architectures. PPOOA metamodel should be extended to include network resources and middleware. This is possible, but it is out of scope the allready-achieved research. The same problem is found with current version of UML.

4. PPOOA implementation in Microsoft Visio[1]

Commercial CASE tools already support UML notation and offer general functionality that is expected of visual languages. Visual elements can be created and deleted; manipulated; connected; copied and pasted; and saved and loaded.

[1] Visio is a trademark of Microsoft Corporation

Developers are already familiar with commercial CASE tools. Thus a particular software development method as PPOOA can be crafted on the top of the existing general functionality of a commercial visual CASE tool.

We illustrate the approach with the implementation of PPOOA on the top of Microsoft Visio, considering the extension mechanisms of the tool, and the PPOOA metamodel previously created as an extension of UML metamodel. The PPOOA profile is compatible with UML and improves its use in real-time architectures. The UML stereotypes are extended with the elements of the style (periodic and aperiodic processes, controller object, and coordination mechanisms). UML Activity diagrams are also adapted for PPOOA requirements, specifically modeling resources and considering scheduling points.

We decided to choose Microsoft Visio because it is highly customizable and offers a robust user interface to build upon. Furthermore, it has a large user base and is commonly found in industry. Visio can be easily customized for different domains as nicely illustrated by the applications that Visio already offers: UML and other software methods diagrams, network architectures diagrams, ER diagrams to model databases, electrical engineering diagrams, pipes and instruments diagrams for industrial processes etc.

A Visio customized application such as PPOOA, offers

- Customized stencils that contain the visual elements of PPOOA,
- additional toolbars, accelerators and menus,
- additional menu entries in a visual element's context menu,
- real-time properties for PPOOA visual elements,
- windows that contain hierarchical views ("drawing explorer window"),
- PPOOA specific error messages (violation of PPOOA building rules),
- and PPOOA help features as an extension to the Visio help systems.

The Visio object model represents the objects, properties, methods and events that the Visio engine exposes through automation. Most objects in the model correspond to items, the user can see and select in the Visio user interface. Typical objects in the Visio model are: application/global object, document object, page object, master object, selection object, shape object and window object. For example, a shape object can represent anything on a Visio drawing page that the user can select with a pointer or an object from another application that is linked, embedded, or imported into a Visio drawing.

PPOOA-Visio has the following capabilities:

- PPOOA Navigator. This window (Figure 4) is a view of PPOOA repository for the system developed. It contains the references to architecture diagrams and dynamic views (CFAs) of the developed system, and the instances of the PPOOA building elements used in this particular software architecture.

- Handling Visio events. In Visio, events can result from user actions such as opening or closing documents, dropping or deleting shapes on the drawing page, editing the text of shapes and altering shape attributes.

- Implementing PPOOA document structure. That is, pages representing architecture diagrams, and pages representing the dynamic view or responses model represented by the CFAs.

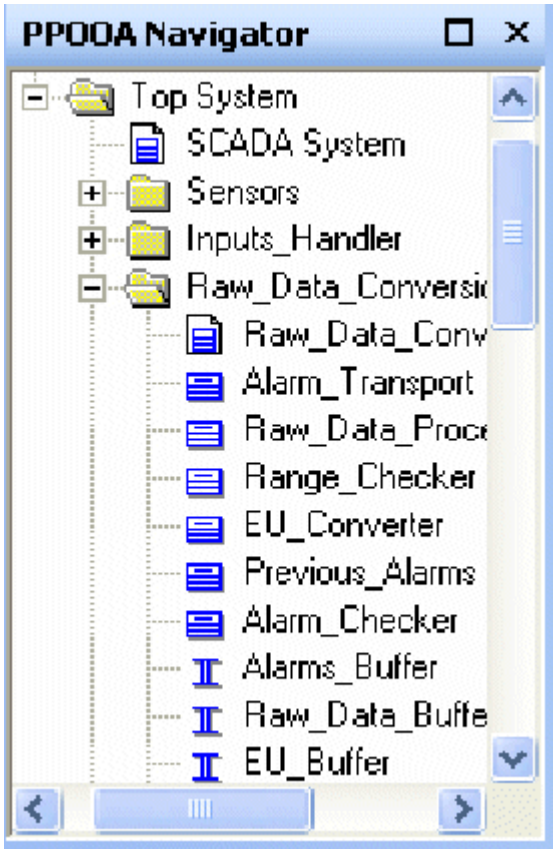

Figure 4. PPOOA Visio Navigator

An example of a PPOOA-Visio window is shown in the figure 5 below. It represents the architecture diagram for the" raw_data_processing" subsystem architected using PPOOA-UML building elements. For PPOOA we could reuse several shapes (masters in Visio terminology) that were already part of Visio UML template. Other shapes as coordination mechanisms (semaphore, bounded buffer, transporter and rendezvous) are specific of PPOOA building elements. In the subsystem architecture diagram, components and coordination mechanisms (buffers and transporters) are independently tagged, and their attributes defined and stored in the tool database.

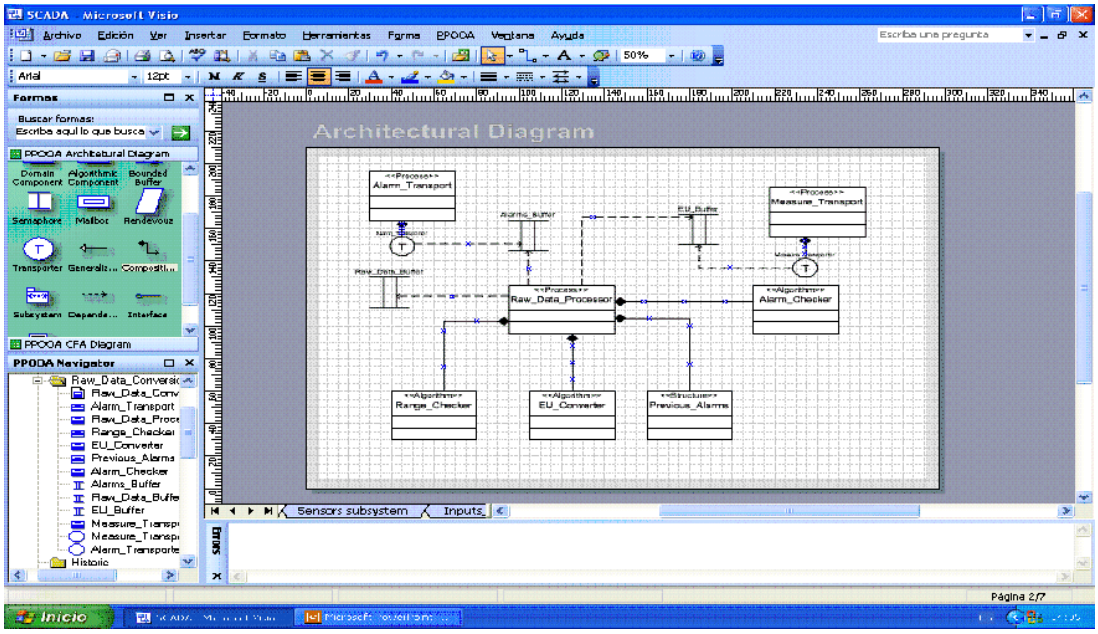

Figure 5. Example of a Subsystem Architecture Diagram

For PPOOA dynamic view, we propose the UML activity diagram notation extended to implement CFA building elements and PPOOA constraints. Figure 6 is an example describing the response that models the activities of the "processing of raw data" to convert it to engineering units and check its alarm condition. Different instances of the components represented in the architecture diagram can participate in the CFA. In a complex and large system we need several CFAs to model the complete system behavior

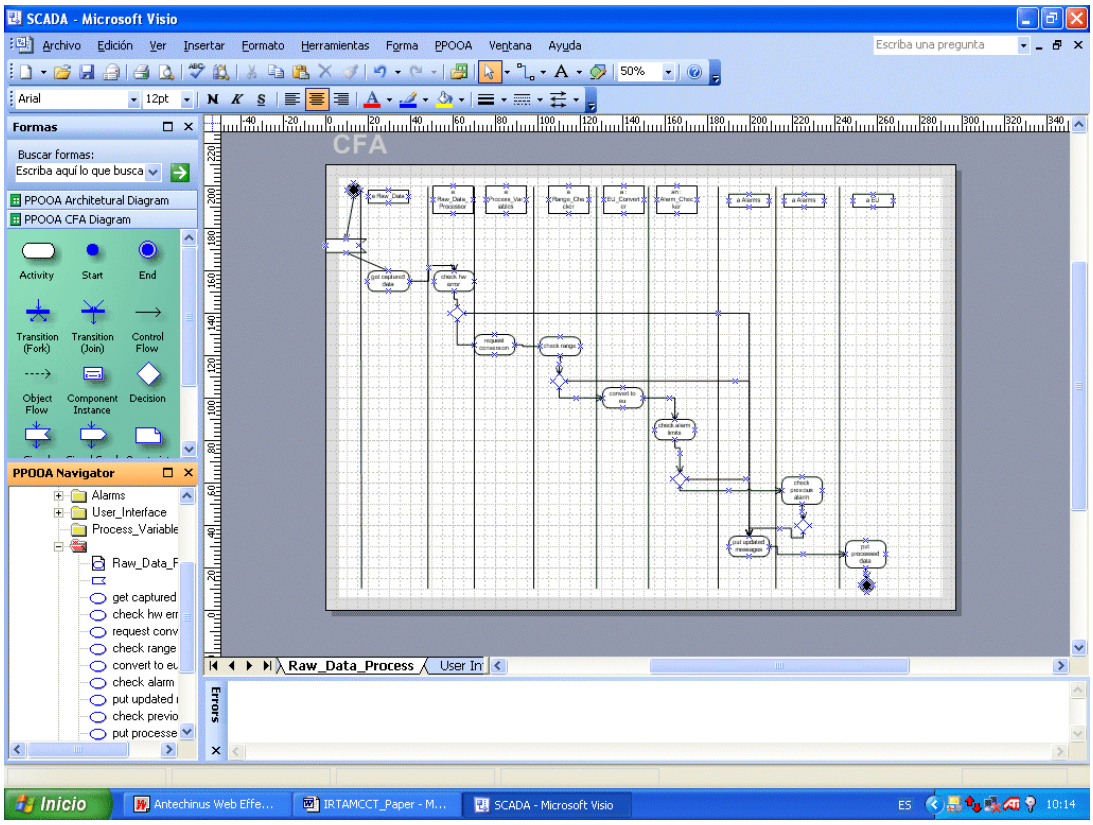

Figure 6. Example of CFA

Other important PPOOA Visio tool feature is the automatic generation of Architecture documentation. The PPOOA Visio Tool generates automatically the documentation of the system architecture developed

- Description of each building element, its domain attributes, interface and real time attributes
- Generated in HTML format (easy to convert to Microsoft Word and Adobe Acrobat pdf)

PPOOA-Visio add-on is a free trial license, so the user only needs to acquire Microsoft Visio 2003 and request the free PPOOA add-on.

5.References dealing with PPOOA and PPOOA_AP

1. "A Taxonomy of Coordination Mechanisms Used in Real-Time Software Based on Domain Analysis". CMU/SEI-93-TR-34. Software Engineering Institute. Carnegie Mellon University. December 1993.

2. ”A Taxonomy of Coordination Mechanisms Used by Real-Time Processes”. ACM Ada Letters. March 1997.

3. ”An Architectural Style for Object-Oriented Real-Time Systems”, 5th International Conference on Software Reuse.Victoria (Canada). IEEE 1998.

4. "Architecture Modeling at Preliminary Design of Real-Time Systems". European Reuse Workshop. Madrid. European Software Institute, November 1998.

5. “Extending UML for Real-Time Component Based Architectures”, 14th International Conference on Software and Systems Engineering (ICSSEA), Paris, December 2001.

6. Chapter 12 of the book: "Business Component-Based Software Engineering", Kluwer Academic Publishers, Boston, October 2002.

7. “A Process for Architecting Real-Time Systems”, 15th International Conference on Software and Systems Engineering (ICSSEA), Paris, December 2002.

8. “Implementing a Real-Time Architecting Method in a Commercial CASE Tool”. 16th International Conference on Software and Systems Engineering (ICSSEA), Paris, December 2003.

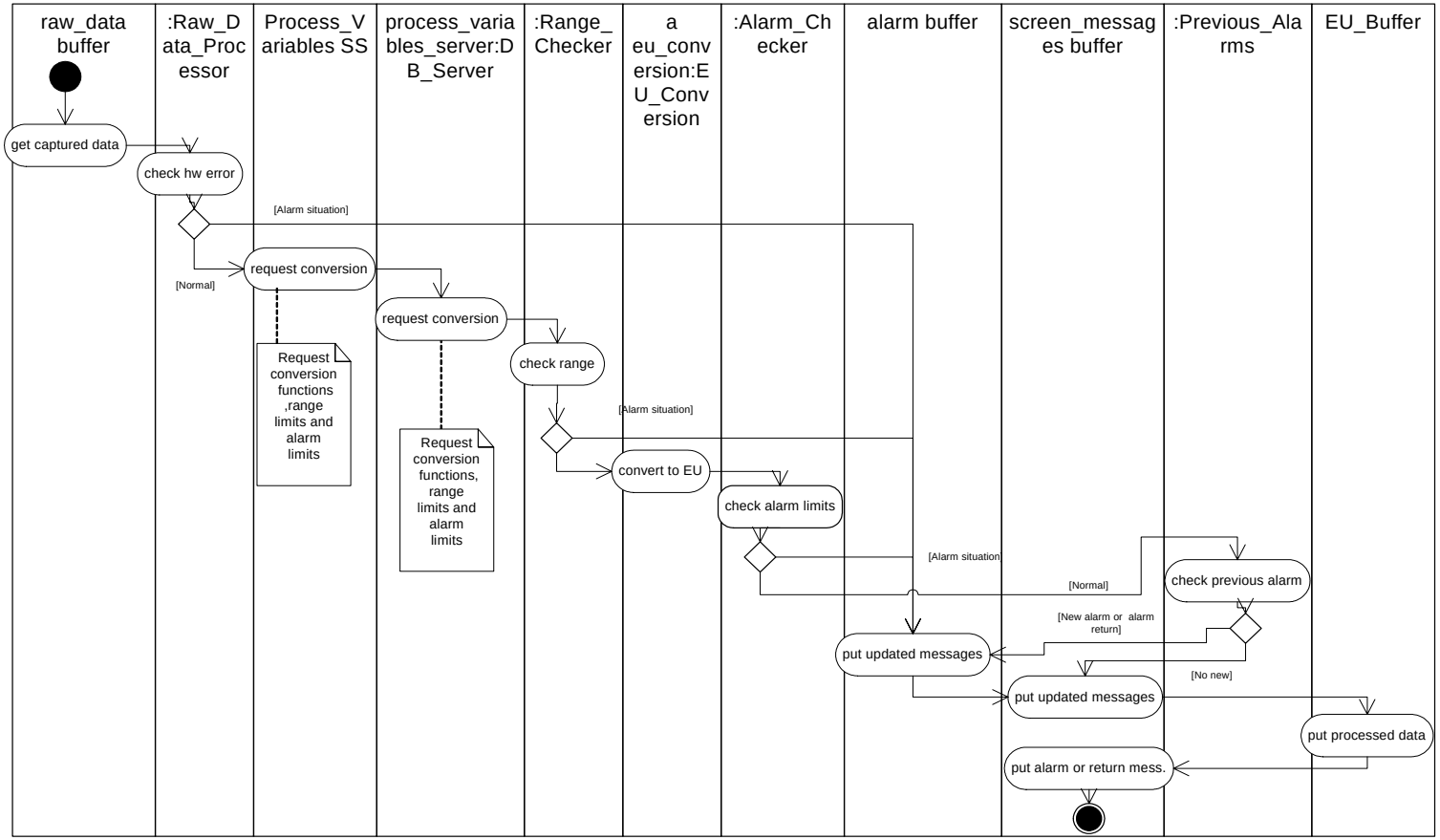

Figure 3. Raw Data Processing CFA