Applying a Process-Oriented Algorithm for development of Automatic Number Plate Recognition System

Atanas Atanassov

Abstract: *The paper presents the application of an algorithm [1] based on process oriented algebra of Communicating Sequential Processes (CSP) for the development of a flexible software architecture intended to an Automatic Number Plate Recognition (ANPR) System. The capabilities of the available on the market ANPR systems were analyzed and the requirements to the new generation of ANPR system were formulated. The appropriate software components corresponding to these requirements were identified. All ANPR components were decomposed to the CSP processes exchanging specific messages via channels. Finally, these processes were mapped to the operating system's processes and threads communicating in parallel. As a result an high-performance reconfigurable ANPR System was provided to the customers.*

Key words: *ANPR system, WEB based UI, traffic control software, CSP*

INTRODUCTION

The automatic number plate recognition (ANPR) systems become of great importance for on-line traffic control. There are different types of ANPR systems deployed in special scopes: traffic control, traffic violation, control of parking lots, highway lots, control of the access to cities centers, etc. Those systems, in general, are based on the CCD infrared or color cameras, radars (optional), some digital inputs and outputs, OCR software for number plate recognition, Data Bases (DB) and Graphical User Interface (GUI) for interaction to the operators. Usually, the process (Fig. 1) of number plate recognition includes:

- Getting image stream from the camera and sending it to the operator PC,
- Converting the image stream to the sequence of images in appropriate image format (TIF, GIF, JPG, BMP) used by the OCR software. It is done by SW for image handling (IH),
- Recognition or not of the number plate via OCR [4],
- Verification of the found plate number using plates data in DB;
- Visualizing the found number (and car image, image stream) to the operator.
- In some cases sending some data or signals to the digital outputs or reading additional data from the inputs.
-

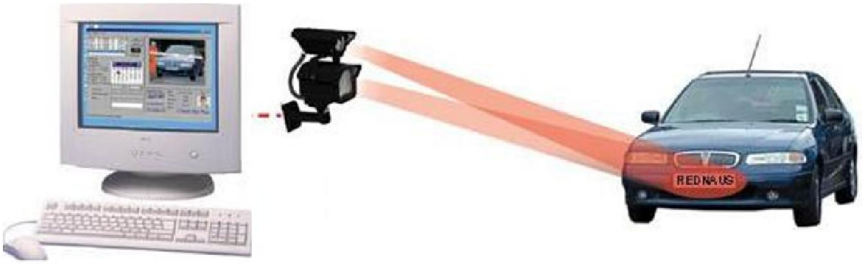

Fig. 1. Hardware and software components of ANPR system

Most of the existing ANPR systems [5] are with specific purposes corresponding to one or more of mentioned above scopes. In most of them the camera hardware and the computer(s) running the software (OCR, GUI and DB) are placed on different places and the connection between them is based on coaxial or optical cables. A great majority of systems are still using this architecture. This is not a robust architecture because if the PC controlling the camera fails the whole ANPR system fails. It requires complex installation and start-up. It is necessary to install video and control wire for each one of the cameras, furthermore, it is necessary to provide power supply cable to the cameras. If the distance between the lanes is too long, the signal of the cameras will not arrive with clearness to

CompSysTech'13 Local Proceedings, ISSN 1314-9687, pp. 22 - 27, 28-29 June 2013, Ruse, Bulgaria

the PC. This architecture is very expensive because of the wiring and the time of installation is multiplied, as well the maintenance of the system is multiplied.

Other ANPR systems are based on cameras with software that implements some IH and OCR procedures. Last generation of ANPR systems (Fig. 2) are embedded systems [3] that incorporate in one body the camera (infrared, color or both), the illumination, the PC hardware with external interfaces, power supply, the operating system, IH, OCR and communication SW. They are known as all-in-one ANPR Systems. These systems are capable immediately to inform other external systems or interested organization for vehicles data.

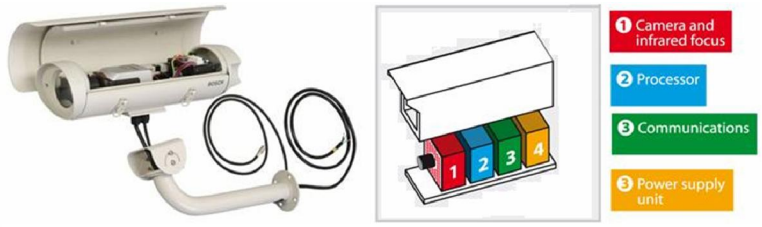

Fig. 2. All-in-one ANPR Systems

The advantages of this architecture are that it is simpler [4] and all the necessary elements for the ANPR system are integrated in the same housing. The equipment may be connected by Internet or GPRS to the client application. It is a modular architecture and the failure of one ANPR camera does not affect the operation of others.

Current paper presents the new flexible process-oriented architecture of an embedded ANPR system. The system is built by set of OS processes related to the required ANPR functionalities. The proposed architecture is developed on the base of theory of the Communicating Sequential Processes (CSP).

IDENTIFYING THE SOFTWARE REQUIREMENTS TO ANPR SYSTEMS

After intensive investigations and analysis of the leading ANPR systems vendors Tattile, Vitronic [9], Leutron Vision [7], Hi-Tech Solutions [8], etc. the general and specific functional requirements valid to all contemporary ANPR systems were defined.

General requirements:

- Online capturing of the images of the vehicles' number plates.
- Online optical character recognition (OCR) of the captured vehicles' number plates.
- Providing an operator control of the observed by the ANPR system object, including the supervising the ANPR live video-stream of the controlled road, lane, etc. and visualization of the found number plates
- Support of databases with information of the vehicle plate numbers (Black and White lists) and providing the search in DB in order to find if there is a hit (match) of the found plate number in the White/Black list.
- Collecting and sending of plain or encrypted information (evidential records (ER)) of each plate number to the specific database servers via internet or GPRS.
- Configuration of the ANPR camera parameters, as brightness, gain, frames per second, shutter, position, angles, etc. Tuning of OCR parameters.
- Restart of the ANPR application or Reboot of the operating system.
- Authentication of logging ANPR users.

- Reporting the status of entire ANPR systems (its specific hardware devices or software components) to the operator.

Specific requirements:

- Configuration of the interfaces to the external systems (DB servers, GPRS, etc.)
- Adjustment of serial lines (COM ports) in order to control parking or highway lots
- Download and upload of configuration files or software updates.
- Support of configurable rules and actions related to found plate numbers, meaning how and where the ERs should be saved or which external system should be triggered.
- Collecting of statistical data (recognized/not recognized numbers per hour, day).
- Collecting of logging and tracing data.
- Support of recording of test decks with live-video images used for further adjustment of the OCR algorithm.

CSP THEORY AND PROPOSES PROCESS-ORIENTED ALGORITHM

The theory of CSP, developed by A.S. Hoare [6] is a mathematical formalism - the process algebra describing the behavior and interactions between components of a wide range of systems. The basis of the theory are the processes exchanging sequences of messages (events) via input and output channels. The process is a component, encapsulating some data structures and algorithms that are inaccessible to other processes. They are private in the terms of object-oriented programming. Each process executes its own algorithm in one or more threads, scheduled by the operating system /OS/. In control systems the process can be related to the controlled object, to the controller or to various filters, adders, nonlinear elements, etc. From more abstract point of view the process can be regarded as the finite machine, driven by events (or data) received from other processes via input channels.

Steps of the Process-Oriented Algorithm:

1. Determination of overall control process or overall software system.
2. Decomposition of this process to sub-processes on the base of the functional requirements and constraints (time, hardware, software, etc.). In this case the interdisciplinary knowledge in control systems, parallel programming, OS and real-time OS is required.
3. Definition and determination of the channels and the messages exchanged over these channels (alphabet of the processes).
4. Determination of processes' priorities (from the control point of view)
5. Refinement of algorithms of individual processes.
6. Selection of a software programming architecture and mapping the CSP-processes to OS processes and threads.
7. Distribution of the OS processes over hardware (processors and/or cores) .
   For process allocation the following formula can be applied:

$$Ttotal = Tos + \sum_{i=1}^{N} (Tproc_i + Tswitch_i) < Td$$

where:

Ttotal – is a total software execution time;
Tos – is time for OS;
$Tproc_i$ – is the execution time of the process i ;
$Tswitch_i$ – context switch time for process i;
Td – is sampling time of the control system or critical time of the software system.

Rules for mapping and allocation of the CSP processes to Os threads and processes and on the processors:

- IF *Ttotal* < *Td*, THEN the processes can be executed to one processor or core.
- IF *Ttotal* > *Td,* the processes can be executed to N processors or cores, or to a faster processor.
- $N = Ttotal/Td$ (N is the bigger integer number)
- IF processes $Proc_i$ and $Proc_j$ are working with shared data THEN they can be formed as threads in one OS process.
- IF processes $Proc_i$ and $Proc_j$ are not working with shared data THEN they can be formed as separate OS processes.

The execution time of certain process $Tproc_i$ includes a communication time $Tcomm_{ij}$ to other processes and threads, where j varies from 0 to N-1.

In order to determine the above mentioned times two programming classes- Statistics and Logger were developed. They serve to record the statistics associated with threads and processes of the developed software systems. Statistics class registerss the IDs of OS threads and processes, their priorities, their names (defined by the programmer), their time for communication with external processes, their time to work on common data and more. Class Logger records statistical information in log files, allowing recording of events / information with different priority (up to 8 different levels of importance). Offline analysis of the log files information is used in the decision to map a process on a thread or how to allocate the processes on the processors or cores.

IDENTIFYING CSP PROCESSES, CHANELS AND MESSAGES

After the analysis of the given above requirements, and taking into account the results achieved in [2] the ANPR system can be presented as one general CSP process composed by the following (Fig 3) processes:

- Image Handler (IH) Process is responsible for getting raw live images from the IR or color cameras and transforming them into GIF, TIF or JPEG formats used by the OCR process. The IH also redirects non-transformed images to ANPR Manager Process.

- The OCR process produces the results: – found plate numbers of the observed vehicles. These results are directed to ANPR Manager Process for further handling.

- The ANPR Manager Process (AM) transforms the live images to the image formats used by the Web Server Process and sends them to it. It gets the found number plates data and produces the ER images which are sent to the Web Server and/or to the remote DB Server. AM is also responsible for checking whether the found number plate matches the ones given into the Black or White verification lists, or if the number matches the rules and actions formed via Web UI. It also sends the status data from Status Manager Process to the Web Server and DB Server. AM is responsible for I/O ports control.

- The Status Manager Process collects hardware info (temperatures, humidity, voltages, etc.) regarding ANPR cameras and PC. It also gets the status of all software components. The Status Manager Process sends mentioned data via AM to the Web Server or logs it locally.

The WEB Server Process sends the live stream images and the results to the Web UI browser (operator). It is also responsible for online adjustment of the ANPR system's parameters. These parameters are exchanged between Web UI browser and AM via WEB Server process.

As can be seen the AM process is very complicated. That's way it was additionally decomposed to set of sub-processes – each of them related to the specific activity or communication (Fig 4).

- The Cameras, I/O Ports and DB Server can be interpreted as external hardware CSP processes which are connected to the ANPR System process via mentioned channels.

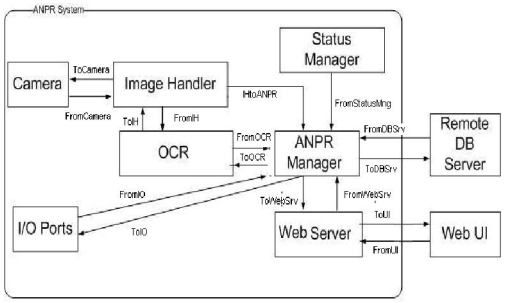

Fig. 3. CSP processes of ANPR System

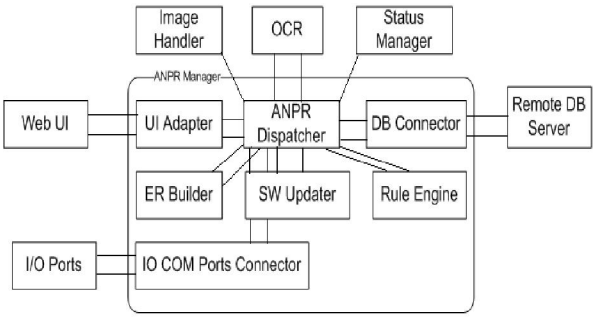

Fig 4. ANPR Manager sub processes

Identifying CSP channels and messages

The arrows on Fig 3 represent the point-to-point CSP channels. Over these channels the processes are exchanging messages. Channels pointing from AM to the surrounding processes are named ToXXX (where XXX corresponds to the targeted process) and channels named FromXXX are targeted to AM, for example FromOCR or ToOCR. Similar is the notation between other software and hardware processes. In general, the messages are of two types – messages with binary image data (in different formats and compressions) and text messages (in plain CSV or XML format). The length of the messages is variable and depends on the size of the images or number of the message parameters.

Mapping the CSP processes to OS processes and threads

After the decomposition the ANPR system was represented of 11 CSP processes. The AM process itself was split to seven sub-processes (Fig. 4) controlled by one of them taking dispatching role. These processes are mapped to OS-processes and threads according to their importance. So OCR, Image Handler, ANPR Dispatcher processes and Evidential Record Builder must be executed with highest priority. Other processes forming AM have middle priority. The Web Server and the Status Manager are the processes with low priority. Next CSP record represents the prioritised parallel composition of all ANPR system processes.

$$ANPR=(ImageHandler() \,||\, OCR() \,||\, ERBuilder () \,||\, ANPRDispatcher ()) \overset{<-}{||} (RuleEngine() \,||\, DBConnector() \,||\, UIAdapter() \,||\, SWUpdater() \,||\, IOCOMPort Conn()) \overset{<-}{||} (WEBServer() \,||\, StatusManager())$$

The sign || describes a parallel execution of two processes or group of processes, and the sign $\overset{<-}{||}$ a parallel execution of two processes with higher priority than that of the first (left-one) process.

Based on previous author's experience in the development of similar SW architectures, intended to automated parcel sorting systems [3], etc, the ANPR System was mapped to five OS processes grouped in two priorities: high for IH, OCR and AM processes and, low for WEB Server and Status Manager Processes. Inside the AM process there are, also, two levels of priorities: highest for AM Dispatcher and ER Builder processes and middle for the rest processes.

The decision to implement IH and OCR as separated OS processes derives from the fact that in most ANPR systems they are third party software (executables with known API) that can not be modified or integrated inside the existing processes.

The Web Server was implemented as a multithreaded process and described in details in separate paper [2]. It is running inside the Apache Server installed on ANPR system PC.

The developed ANPR system is ordered by Siemens and is intended to work on Windows and Open Embedded OS. All threads' and processes' communications are based on the binary or text messages exchanged over TCP/IP sockets.

Fig 5. Result view from ANPR System in Web UI

CONCLUSIONS AND FUTURE WORK

Presented in the paper process-oriented algorithm [1] for software decomposition ensures high performance to the developed ANPR system. The algorithm provides efficient way of mapping the CSP processes to corresponding OS tasks and threads. Developed on the base of this approach ANPR system is flexible and can be easily extended or reduced for different goals (speed control, traffic control (Fig. 5), highway tolling, lots control, etc.) by adding or discarding some processes or threads, using simple configuration files. A Number of ANPR system prototypes are in real-time operation in some EU countries. Next steps to ANPR improvement are related to adding modules (OS process) for color camera support, data encryption and on-line SW update. That can be easily obtained on the base of proposed in the paper sophisticated ANPR architecture.

REFERENCES:

[1] Atanassov, A., Algorithm For Synthesis Of Process-Oriented Software And Its Distribution On A Single And Multiprocessor Systems, Journal of UCTM, Sofia 2/2012

[2] Atanassov, A., Tomova. F., Web-Based Subsystem For Tuning Of An Automatic Plate Number Recognition System, Sixth International Conference, Challenges In Higher Education And Research In 21st Century, 2008, (pp 399-402), Sozopol, Bulgaria, 2008

[3] Atanassov, A. Interface and Control Manager for Parcels Sorting, International Conference on Computer Systems and Technology CompSysTech'2004, pp. IIIA.18-1 - IIIA.18-6, Rousse, June 2004

[4] Bar-Hen Ron, A Real-time vehicle License Plate Recognition (LPR) System, VISL, Technion 2002, Paper at http://visl.technion.ac.il/projects/2003w24/

[5] Gordon, A., License plate recognition technology: innovation in law enforcement use, Thomson Gale Press, 2007

[6] Hoare, C.A.R., Communicating Sequential Processes, Prentice Hall, 1985, extended and updated 2004.

[7] http://www.leutron.com/ Leutron Vision

[8] http://www.htsol.com/

[9] http://www.vitronic.de/en/traffic-technology/alpr/

ABOUT THE AUTHOR

Assoc.Prof. Atanas Atanassov, PhD, Department of Computer Systems, University of Chemical Technology and Metallurgy - Sofia, Phone: +359 28163484, E-mail: naso@uctm.edu