REPORT ON CIVA ICT SYSTEMS

Vladimir Machula

1 Introduction

The purpose of this document is to identify and describe different aspects of software, hardware and technical processes used within all CIVA operations. This covers different systems and applications during Championships organization, Judges education and examination, CIVA meetings and work of Subcommission/Working Groups. Software and hardware together with communication interfaces can be simply referred as ICT (Information and Communication Technology) systems.

Second part of the document explains ultimate question “why is a systematic approach necessary” and “why the hell do we need it”.

Third part gives basic insight into ICT system design models, software management, provisioning and lifecycle.

Fourth part tries to give possible solutions.

2 Essentials of ICT Systems

2.1 ICT complexity

Electronic computers and communication technologies are making our life easier and more effective. ICT is bringing the opportunities to do much more in no time. However, computers are sometimes capable of doing more harm than profit. ICT systems itself are nothing. They need users, operators and service consumers. They also need to be programmed and configured. As with many things, computers will provide benefits if used properly in suitable application.

Computer science is a complicated and very complex business. It is quite a long way to become system architect and many people simply do not have enough time/capacity/will to do it. People usually turn to delusions when working with something they do not understand. In the end they under- or overestimating capabilities of such system. Modern audiovisual life lived through fancy movies, TV and videos show super abilities of computer systems and are many times creating false expectations. Let’s face it. Industrial ICT systems can’t be controlled and managed by non-educated users. Management of those systems needs to be entrusted to well-educated specialist.

Management and interaction with ICT systems in operation is just ~~a~~ one phase of system life cycle. Life cycle and their phases will be described later, but it’s obvious that every system needs to be designed and created. The most crucial part of the whole lifecycle is a system design. Without clear and proper description of system behaviors, interfaces, management, verification and validation, it is impossible to create anything else than a small hobby non-critical system. Ignoring ~~of~~ this fact leads to faults with different scales of severity. Even if we talk about faults with impact on health, economic or social environment, we would like to avoid them generally. Entropy rises exponentially with system complexity or converted to simplified statement of third thermodynamics law “It is

impossible for any process, no matter how idealized, to reduce the entropy of a system to its absolute-zero value in a finite number of operations". In other words, a simple system is easy to maintain, verify, validate and modify. If more functions and modifications are added entropy is rising exponentially. Needs for exponential rise of resources spent on system design, documentation, verification and validation to achieve low error rate, sustainable test coverage and appropriate user experience are arising from these assumptions.

2.2 Modern Era Dark Side

Design and system/software management errors have led during history to many breakdowns, but there are two really remarkable ones.

First one is a story of a brand new Ariane 5 space vehicle designed to deliver 3 tons of payload into Earth's orbit. It took 10 years and 6 billion EUR to produce this vehicle. All it took to make huge and expensive fireworks forty seconds after lift-off, was a small software bug trying to fit 64-bit number to 16-bit memory space. One bug, one disaster [1].

Second one is a story of failure which cost life. On Feb 25, 1991, during the Gulf War (Dahran, Saudi Arabia), an US Patriot missile firing post failed to track and intercept incoming Scud missile. The Scud hit US Army barracks, killing 28 and injuring approx. 100 soldiers. It turned out that the cause was an inaccurate calculation of the time since boot due to computer arithmetic errors. The small chopping error when multiplied over time led to significant error. The Patriot battery had been up around 100 hours, and system prediction of next Scud position has been missed by few meters [2].

2.3 ICT in Today World of Aerobatics

Given examples are quite extreme, but both show how a lack of precise system design and software management can be painful. Aerobatics is in fact a multimillion business. Every single competitor spends hundreds of thousands of EUR for training and participation at FAI aerobatics events. With around 200 competitors participating at Championships every year, those expenses cannot be ignored.

ICT tools like ACRO, OLAN, VISIO ARESTI DRAWING, OPENAERO and HMD solutions were used as supporting tools to make Championships organization easier. However, they are turning from tool to necessity. No one can imagine aerobatic championships without ACRO producing start lists, results and judging analysis. Going back to non-computer era is not even possible due to FAI Section 6 requirements. There is also no one producing sequence drawings by hand. We came to an age where more systems will make aerobatics championships more cost effective and easier to organize. Those systems will all be somehow involved in overall results of championships. We will benefit from their use, but they can ruin a whole event or leave that bitter taste with all participants. None of that is desired by anyone.

2.4 ICT Failures in Modern Aerobatics History

Since ICT applications are not used for long time, there have not luckily been many failures causing remarkable problems during FAI/CIVA championships or they have been solved and covered by the organizer and thus not known publically. However there are few minor issues (in matter of caused damage) causing delays and overnight work of very dedicated superfixers (mostly producers of those ICT tools). In past four years, following issues have been observed:

- (ACRO) Errors in starting order creation or even a fall of the whole application and then time consuming need for bug fix and reconfiguration.
- (ALL DRAWING APPS) Sequence checking errors within drawing tools.

- (ACRO) Unhandled user input causing application crash and contest database inconsistency.
- (PHMD) Errors in calibration, mechanical issues and communication failures.
- (WINDSOND) Loss of wind sonds due to misconfiguration and communication problems.
- (SMS INFO SYSTEMS) Problems with delivery to specific mobile operators, unavailability of the service and expensive operations of the SMS information system.
- (RADIO COMM) Problems with connection quality, not suitable equipment, interference with other public users and cost of the service.

Mentioned failures and drawbacks needs to be understood as an example of what has already happened and haven't caused greater problems. Producers of mentioned tools should still be emphasized as those who spent months with putting their valuable knowledge and expensive time into well-being of the whole aerobatic community without expectations of any royalty.

3 Aerobatics ICT System Management

3.1 Overview

Every ICT system should have some kind of management of its lifecycle. Lifecycle is a set of processes which assure the quality level of the final product, drives its development and allow scalability. There is a plenty of different Management systems, but combination of ISO 12207 and ECSS seems to be most usable. First of all they are completely publically available. ECSS is very robust set of standards since it is created for space industry, however it can be easily tailored to the needs of Aerobatics ICT Systems management.

3.2 Limitations

This standardization prescribes minimal needs for system design, requirements specification, documentation, validation, verification and operation. This Standard does not detail system architecture, used technologies and methods or procedures required to meet the requirements and outcomes of a product.

3.3 System Life Cycle

The life of a system or a software product can be modelled by a life cycle model consisting of stages. Models may be used to represent the entire life from concept to disposal or to represent the portion of the life corresponding to the current project. The life cycle model is comprised of a sequence of stages that may overlap and/or iterate, as appropriate for the project's scope, magnitude, complexity, changing needs and opportunities. Each stage is described with a statement of purpose and outcomes. It should be usually composed by following processes:

- Phase 0 – Idea/initial definition of system features or needs identification.
    - Identifying needs.
    - Proposing system concepts.
- Feasibility
    - Detailed needs identification/verification.
    - Solutions proposal including identification of criticalities and risks.
- Preliminary definition
    - Preliminary system requirements definition / specification in a formalized way.
    - Planning and demonstration of a development schedule, budget, target cost and organizational requirements.
- Detailed definition

    - Detailed system requirements definition, verification & validation plan, development of schedule definition, software requirements definition, set of use cases to describe system behavior, etc.
- Qualification and production
    - System development.
    - Validation – Component and system testing according to the planned test coverage. Making sure that product meets system requirements specification.
    - Verification – Acceptance and Quality reviews. Making sure that the system, documentation, functionality and validation meet the overall requirements/product specification.
- Operations / utilization
    - System deployment and configuration.
    - Product usage.
    - System user support and product anomaly investigation and resolution.
- Disposal
    - Archiving of collected data.
    - Transition to next-gen systems.
    - Saying goodbye.

3.4 System Design and Documentation

System design documentation is the most important part of the whole product. It usually consists of documents produced in the following order:

1. Preliminary System Requirements Definitions.
2. Budget/Schedule/Validation definitions and plans.
3. Set of System Requirements Definitions.
4. Set of System Requirements Detailed Definitions / Software Requirements Definitions.
5. Models / use cases (for system as well as for models/software functions etc.).
6. Test plans and test outputs.
7. User documentation.

3.5 Verification & Validation

Every system needs to go through testing process. With more detailed documentation and with more complex test plan, product outcome and user experience rises significantly.

Validation stands for processes to confirm that the requirements baseline functions and performances are correctly and completely implemented in the final product.

For a new development flow or verification flow, validation procedures may involve modeling either flow and using simulations to predict faults or gaps that might lead to invalid or incomplete verification or development of a product, service, or system (or portion thereof, or set thereof). A set of validation requirements (as defined by the user), specifications, and regulations may then be used as a basis for qualifying a development flow or verification flow for a product, service, or system.

Verification is a set of processes to confirm that adequate specifications and inputs exist for any activity, and that the outputs of the activities are correct and consistent with the specifications and input. Verification can be in development, scale-up, or production. This is often an internal (developer) process.

Verification is usually done by programmed automatic sets of component and system tests. If a product contains some kind of user interaction, than user interface tests are also crucial. Verification plan should have defined minimal test coverage ratio estimated by system criticality and risk management.

It is sometimes said that validation can be expressed by "Are you building the right thing?" and verification by "Are you building it right?".

3.6 Reviews

Review is a process or meeting during which a product is examined by a project personnel, managers, users, customers, or other interested parties for comment or approval. In this context, the term product means any technical document or partial document, produced as a deliverable of a product development activity. In case of Aerobatics, a product is reviewed by review/project board established by proposed Technical Support Subcommission of CIVA.

A number of reviews should be conducted during the process of product development, they ~~are~~ should be in following order:

1. SRR – system requirements review
2. PDR – preliminary design review
3. DDR – detailed design review
4. TRR – test readiness review
5. CDR – critical design review
6. QR – qualification review
7. AR – acceptance review

Some of the listed reviews can be skipped or merged except SRR and AR. Those two must receive maximum attention since their output is an insurance of overall product quality.

4 Conclusion and Proposal

During the past few years we have been facing failures which have been solved in last minute and therefore it didn’t luckily ruined running Championships. With higher penetration of such technologies, this danger is more and more imminent.

CIVA also need to take care of long term sustainability of such systems. If there is a perfectly working system for crucial tasks carried out during championships, made by enthusiastic programmer, without a single piece of technical documentation, who can guarantee that this system will be updated and functional in few years? To reduce these drawbacks it is necessary to implement ICT project management and reliable technical documentation according to modern standards.

CIVA need to take care of creditability aspect of such systems as well. It can be very easy to compromise the output of such application, especially in case of intended manipulation. Used systems need to have a way to verify its outputs in form of hash codes, verified logs, etc.

This report proposes to establish **ICT Sub-Commission** with following aims and tasks:

- Validate, verify and certify ICT systems and applications for the use within CIVA operations.
- Support systems and applications developers to tailor and implement development management.
- Manage life cycle model of such systems and applications.

- Find new opportunities for implementation of modern ICT solutions.
- Support championships organizers during the use of mentioned systems.
- Supervise the use of mentioned systems during FAI/CIVA championships.

5 Referenced documents

[1] Ariane 5 – Flight 501 Failure Report

http://www.esrin.esa.it/htdocs/tidc/Press/Press96/ariane5rep.html

[2] GAO Report: Patriot Missile Defense – Software Problem Led to System Failure at Dahran

http://fas.org/spp/starwars/gao/im92026.htm

[3] ISO/IEC 12207 Systems and software engineering — Software life cycle processes

http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=43447

[4] ECSS

http://www.ecss.nl