Technical Evaluation of the Pilots

The Architecture Analysis for the VNF Application

Authors	Mohamed Boukhebouze	30/10/2015
Verification	Philippe Drugmand	02/11/2015

CETIC ASBL – Centre d'Excellence en Technologies de l'Information et de la Communication
Aéropole – Rue des Frères Wright n°29/3 - 6041 Charleroi (Belgique) – Tél. : +32-71-490 700 – Fax : +32-71-490 799
e-mail :info@cetic.be- TVA BE 0474 549 932

Document Control

Creation Date	04/06/2015
Last Modification Date	02/11/2015
Version	2.0

Document Versions History

Version	Date	Description	Authors
0.1	04/06/2015	First Draft of the document	Mohamed Boukhebouze
0.2	18/06/2015	Second Draft of the document	Mohamed Boukhebouze
1.0	24/06/2015	Review the document	Philippe Drugmand
1.1	30/10/2015	Minor reviews	Mohamed Boukhebouze
2.0	02/11/2015	Review the document	Philippe Drugmand

References

[1]	ISO/IEC 25000, System and Software Quality Requirements and Evaluation (SQuaRE). 2014.
[2]	"OMG Unified Modeling Language (OMG UML), Superstructure. Version 2.4.1". Object Management Group. 2014.
[3]	Kruchten, Philippe (1995, November). Architectural Blueprints — The "4+1" View Model of Software Architecture. IEEE Software 12 (6), pp. 42-50.

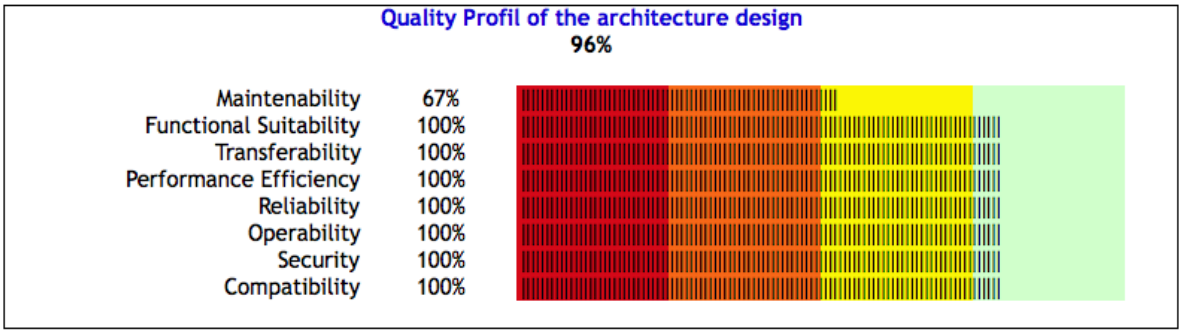
Confidentiality Level

Public	
Restricted to a specific audience (Numericanal Project Members)	**x**
Confidential (for CETIC members only)	

Contents

Executive Summary

VNF Application Architecture Maturity



Caption: Risques **Very High** **High** **Medium** **Low**

Introduction

This document presents the results of a quality evaluation (called "Archicheck") of the VNF application architecture. This evaluation is based on a set of questions asked (interview) to one or more designer and developers of the application. Archicheck helps to quickly provide a quality evaluation report by relying on the software quality standard: ISO25000 [1]. The result of this evaluation allows identifying the strengths and weaknesses of the application architecture. In addition, the evaluation report helps to identify recommendations to improve the architecture as well as further analysis tracks. Archicheck considers on the following criteria:

- **Functional Suitability:** The capability of the software product to provide functions, which meet stated and implied needs when the software is used under specified conditions (what the software does to fulfil needs).
- **Reliability:** The capability of the software product to maintain its level of performance under stated conditions for a stated period of time.
- **Performance Efficiency:** The capability of the software product to provide appropriate performance, relative to the amount of resources used, under stated conditions.
- **Usability:** The capability of the software product to be understood learned, used and attractive to the user, when used under specified conditions (the effort needed for use).
- **Security:** The capability of the software product to protect system components from accidental or malicious use: access, modification, destruction or disclosure.
- **Compatibility:** The capacity of two or more software components to exchange information and / or perform their functions by sharing the same hardware or software environment.
- **Maintainability:** The capability of the software product to be modified. Modifications may include corrections, improvements or adaptations of the software to changes in the environment and in the requirements and functional specifications (the effort needed to be modified).
- **Transferability:** The capability of the software product to be transferred from one environment to another. The environment may include organizational, hardware or software environment.

Context

The evaluation described in the report concerns the VNF application architecture (PoGo).
Evaluation data: 04/06/2015.
People interviewed: Pierre Alexandre BORTOLIN (CGI) and Ali (CGI)
Participants: Alaric BLAKEWAY (VNF) and Mohamed BOUKHEBOUZE (CETIC).

Observations

- In general, the results are excellent for an overall score of 96%.
  - All criteria are well supported.
  - The maintainability criteria can be improved (see Section 7).
- All best practices are followed to ensure the usability of the application (see Section 4):
  - A help section is available in the application.
  - The user interface is modelled.
- The functional suitability criteria of the application are well covered since a data dictionary is available in order to describe all information about data objects or items in the database. In addition, the maturity of the developed components is determined (see Section 1).
- The reliability of the application is ensured since an exception-handling component is implemented. Moreover, a restore mechanism that reduces downtime of the service is implemented (see Section 2).
- The performance efficiency criteria of the application are well covered since the resources consumption and the workload variation are controlled (see Section 3).
- The application favours a good usability since it relies on standard user interface widgets, which makes the applications more intuitive (see Section 4).
- The application covers the security criteria: The security components are implemented. In addition, the application log traceability is taken into account, which helps identifying the cause of issues and application problems (see Section 5).
- All best practices are followed to ensure the compatibility since the list of alternatives third-party components is identified (see Section 6).
- The transferability of the architecture is ensured since a physical view of the system is defined. Moreover, The detailed network image of the architecture is available (see Section 8).

Recommendations

- Improve the documentation of the architecture by describing the 4+1 views of the architecture [3]:
  - **Logical view** is concerned with the functionality that the system provides to end-users.
  - **Development view** illustrates a system from a programmer's perspective and is concerned with software management.
  - **Process view** deals with the dynamic aspects of the system, explains the system processes and how they communicate, and focuses on the runtime behavior of the system.
  - **Physical view** depicts the system from a system engineer's point of view. It is concerned with the topology of software components on the physical layer, as well as the physical connections between these components.
  - **Scenarios** describe sequences of interactions between objects, and between processes.
- Improve the maintainability by documenting the different aspect of the architecture (Section 7). This documentation should rely on standard description language (UML [2]). So that, handover of the architecture to new developers becomes easier.

1. Functional Suitability

The capability of the software product to provide functions, which meet stated and implied needs when the software is used under specified conditions (what the software does to fulfil needs).

Strengths

The architecture documentation is available.

The functional and technical documentation is available.

The software architecture documentation gathers all the details of the software structure: the architectural views, the software components, properties of those components, and the relationships between them, etc. Documenting software architecture facilitates communication between stakeholders, captures early decisions about the high-level design, and allows reuse of design components between projects.

A data dictionary is available.

A conceptual schema of the database is documented. Moreover, the physical schema od the database is also documented. This documentation includes the description of the tables and the attributes of the database.

A data dictionary allows describing all information about data objects or items in a data model: meaning, relationships to other data, origin, usage, and format. A data dictionary can be used to understand where a data item fits in the structure, what values it may contain, and basically what the data item means in real-world terms.

The maturity of the developed components was determined.

The maturity of the developed components was determined. Unit testing is done for each component. In addition, source code is reviewed regularly. Finally, the test scenarios are done in order to make a functional validation.

The identification of the maturity level (beta, stable, etc.) of each implemented component helps to indicate the expected quality the concerned component. This indication can be used to define the development roadmap and putting into production. An example of the level of maturity establishment is the analysis of the comments "TODO" or "FIXME" in the source code.

Recommendations

None.

2. Reliability

The capability of the software product to maintain its level of performance under stated conditions for a stated period of time.

Strengths

An exception-handling component is available.

Several exception-handling mechanisms are implemented to ensure the reliability of the system. For example, try-catch is used to control the correctness of the user inputs.

A component that handles exceptions allows ensuring that the application is running in controlled manner. Such a component should ensure that for unexpected inputs the application remains operational. This is very important especially when the application is available in SaaS mode because the error caused by a user can impact the user experience of other users.

A restore mechanism for the solution is available.

A cluster is used to ensure the high availability of the system.

A mechanism for restoring solution reduces downtime of the service. This mechanism should ensure that all the necessary components and services are correctly restarted, in the correct order and without major loss of information.

Recommendations

None.

3. Performance Efficiency

The capability of the software product to provide appropriate performance, relative to the amount of resources used, under stated conditions.

Strengths

A mechanism for resources consumption estimation is available.

An estimation of the resources consumption is done based on an XCode[1] tool called: Instruments[2]. This tool allows examining the behavior of application and tracking the resource consumption.

A mechanism for measuring the consumption of resources allows planning the investments based on statistics and projections. In the case of SaaS solutions, this mechanism allows providing operating cost elements for determining the pricing of the service.

The time spent by each component for the use of the resources is estimated.

The developers are aware about the time spent by each component for the use of the resources.

Performing a profiling application can identify which parts of the application (modules, and treatments algorithms, etc.) require a longer running time. These parts should be played special attention during the optimization phase.

Recommendations

None.

[1] https://developer.apple.com/xcode/

[2] https://developer.apple.com/library/mac/documentation/DeveloperTools/Conceptual/InstrumentsUserGuide/Introduction/Introduction.html

4. Usability

The capability of the software product to be understood learned, used and attractive to the user, when used under specified conditions (the effort needed for use).

Strengths

A help section is available in the application.

Some functionalities description is available. In addition, the mobile applications relay on standard user interface widgets, which makes the applications more intuitive.

Define a help in the application facilitates the handling and understanding of the application and contributes to user satisfaction.

A user interface modelling is available.

The user interface prototypes are available. In addition, the storyboards of the applications are defined using XCode.

A user interface modelling allows validating the sequence of screens (navigation diagram), and taking into account the interactions of the user interface with the business layer.

Recommendations

None.

5. Security

The capability of the software product to protect system components from accidental or malicious use: access, modification, destruction or disclosure.

Strengths

A security management component is available.

The security aspect is taken into account in the development of the applications.

A security management component is important whatever the application. The level and type of security that are managed by this component depends on the application context. These levels and types of security should be described in the component documentation.

The application log traceability is taken into account.

Crashlytics[3] tool is used to manage the application crashes (logs, notifications, reporting, etc.). In addition, Log4j[4] is used to keep trace of the database transaction and Web services execution.

The traceability of the application log is useful for identifying the cause of issues and application problems. The application log can also be used to establish application usage statistics. This traceability should be pondered to ensure that consistent information is reported when using logging mechanisms (log4j, log4net, etc.).

Recommendations

None.

[3] https://www.crashlytics.com/
[4] http://logging.apache.org/log4j/

6. Compatibility

The capacity of two or more software components to exchange information and / or perform their functions by sharing the same hardware or software environment.

Strengths

An Application Programming Interface is implemented.

The VNF system offers several Web services. These Web services allow providing large waterway information like events and locks information.

An Application Programming Interface (API) enables data sharing between internal IT systems and between one organisation and another. API also allows ensuring interoperability, usability, and reusability.

A list of alternatives external components is available

The native iOS and Android libraries can be used as alternative to the current third-party components.

The identification of the alternatives external components allows ensuring an alternative solution in case one of them becomes unavailable. These alternatives should be compatible with the exiting architecture components.

Recommendations

None.

7. Maintainability

The capability of the software product to be modified. Modifications may include corrections, improvements or adaptations of the software to changes in the environment and in the requirements and functional specifications (the effort needed to be modified).

Strengths

The licenses of the tiers components are available.

Several third party components are used. For example, a networking library for iOS called AFNetworking[5] library. This library allows implementing a secure communication with the server. Apple libraries are also used to design the iOS application. Last but not least, Google map libraries are used to display the map and the marks on Android application. These libraries are open-sources.

Having licenses on third-party components is important to avoid legal problems, branding, negative campaigning, etc. A support contract should be concluded and the conditions properly analyzed, mainly in terms of availability, time management and bug fixing time.

Middleware software is used to connect software components or enterprise applications.

The Hibernate framework is used as a persistence layer of the system.

A middleware layer provides uniform, standard, high-level interfaces to the application developers and integrators, so that applications can be easily maintained, composed, reused, ported, and made to interoperate.

Recommendations

Use UML diagrams to describe the architecture.

UML language is not used to describe the architecture [3].

UML is the standard language for documenting and modelling system. UML provides multi-level diagrams to describe the structure and the dynamic aspect of the architecture in understandable way. So that, handover the architecture to new team becomes easier. In addition, The development of the UML diagrams can be reduced by using the different UML support tools that are available on the market.

[5] https://github.com/AFNetworking/AFNetworking

8. Transferability

The capability of the software product to be transferred from one environment to another. The environment may include organizational, hardware or software environment.

Strengths

A physical view (deployment diagram) of the system is available.

The system is hosted on the VNF server network. The physical view of this network is well documented.

A physical view depicts the system from a system engineer's point of view. It is concerned with the topology of software components on the physical layer, as well as the physical connections between these components. This view is also known as the deployment view. UML Diagrams used to represent physical view include the Deployment diagram (source: Wikipedia)

A setup mechanism is available.

The iOS and Android applications are available on App Store (Apple application store) and Google Play (Google application store) respectively. The setup of the application is managed by these stores. In addition, documentation is available to describe the setup process of the test version of the iOS and Android applications.

A setup mechanism allows automating (part of) the installation of the solution. This mechanism enables to simplify and speed up the installation of a new version of the solution.

The detailed network image of the architecture is available.

The VNF network image is available.

Having a network image allows identifying and monitoring the achievement of the physical view (deployment view). In corrective or preventive maintenance, this image allows speeding up and improving the corrective or preventive maintenance.

Recommendations

None.