UDC: 004.9:911.5/.9:528.94

MAIN CONCEPTUAL PROVISIONS OF CREATION OF AN ELECTRONIC STATE REGISTER OF IMMOVABLE CULTURAL HERITAGE OF UKRAINE. PART 2: PROCESSES

V.S. CHABANIUK, *Ph.D. Physical and Mathematical Sciences, Senior Researcher, Institute of Geography of the National Academy of Sciences of Ukraine, Email:* chab3@i.ua

O.P. DYSHLYK, *CEO, "Geomatic solutions" LLC,*

Email: dyshlyk@geomatica.kiev.ua

K.A. POLYVACH, *Ph.D. of Geographical Sciences, leading researcher,*

Institute of Geography of the National Academy of Sciences of Ukraine

Email: kateryna.polyvach@gmail.com

/ V.I. PIORO, *director*

PO "Ukrainian Museum Development Center", Kyiv

Email: pioro@ukr.net

I.M. KOLIMASOV, head of production of *"Intelligence systems-GEO" LLC,*

Email: kolimasov@ukr.net

Y.V. NECHIPORENKO, *Department of Cultural Heritage Protection and Museums, Ministry of Culture and Information Policy of Ukraine,*

Email: julnech@ukr.net

Abstract. *Part 2 describes the processes of creation of a new modern electronic State Register of Immovable Cultural Heritage (CH) of Ukraine. They are part of the methodology based on AGIS-CH1 Solutions Frameworks (SoFr), where AGIS-CH1 represents the first queue of the hierarchically structured CH Atlas GeoInformation System (AGIS). AGIS generally consists of four strata: Operational (ω), Application (α), Conceptual (β) and General (γ). The processes considered in the article refer to AGIS-CH1 αSoFr, which defines the activity between subsystems of AGIS-CH1*

Application and Operational strata. Also mentioned are the processes related to AGIS-CH1 βSoFr, which determines the activity between subsystems of AGIS-CH1 Conceptual and Application strata.

AGIS-CH1 SoFr is defined by packages and relations between these packages in AGIS-CH1 Publications-Products-Processes-Basics-Services "petrad". Packages Products-Processes-Basics of AGIS-CH1 and the relations between them are called the main triad of AGIS-CH1 SoFr. This triad is the basis of the Main Conceptual provisions 1-3. For the latter, the following relations apply: SoFr.Products – provision 1, SoFr.Processes – provision 2, SoFr.Basics – provision 3.

Part 2 recommends the development and quality assurance processes from SoFr AGIS-CH1.Processes of activities for the creation of AGIS-CH1. These recommendations are actually Main Conceptual Provision 2.

Key words: *Solutions Framework (SoFr), Atlas Geo-Information System (AGIS), State register of immovable cultural heritage, processes of development and quality assurance.*

Introduction

The consideration of the activities that are recommended to be organized for the successful implementation of the project to create a new modern electronic State Register of Immovable Cultural Heritage (CH) of Ukraine continues. To organize this activity, at the beginning of the project, it was proposed to apply a methodology based on the so-called Solutions Framework (SoFr) of the first phase of this Spatial Information System (SpIS). The first stage of the SpIS should be an instance of the Atlas GeoInformation Systems (AGIS, [1]) class, denoted by AGIS-CH1.

Instead of complicating the notations already used in Part 1 [2], and which will be used later, we provide here additional explanations for them. The notations refer both to the final result of creation - AGIS-CH1, and to the methodology of its creation - AGIS-CH1 SoFr. In this article, we distinguish between the notions of "creation" and "development". The first is most often used together with the notion of "system". We understand the "development" notion as a part of the "creation" notion. Most often, we apply it to a part of the system, and the part can be, for example, a certain state of the system. Here we mean, for example, the state of the system before and after testing. That is why the processes of "development" and "quality assurance" of products are understood as processes of "creation". We hope that the explanations will help to better understand the results of all parts of the paper, as well as explain why we do not complicate the notations.

Explanation of notations. AGIS-CH1 $\in$ AGIS, where $\in$ means that an instance of the AGIS-CH1 object belongs to a set of AGIS objects. We could use the notation/formula AGIS-CH1 $\in$ {AGIS-CH1}, but this notation narrows the set of "acceptable" implementations of AGIS-CH1. We could denote the class of systems AGISs and some translation of it into the Ukrainian language (АГІСи?), but in informatics the UML (Universal Modeling Language) notation is used. According to it the class of objects is usually denoted by the name of the class in the singular **AGIS-CH1** in "bold" font, and the instance of the object by the name of the class in the singular, but with the name of AGIS-CH1 underlined.

The record "AGIS-CH1 SoFr" is a little more difficult to understand. There are four results of applying this SoFr: 1) AGIS-CH1 SoFr – an instance of AGIS-CH1, which can be created by applying a corresponding instance of SoFr, 2) **AGIS-CH1** SoFr – a set {AGIS-CH1} of acceptable instances of AGIS-CH1, which can be created by applying a corresponding specific SoFr, 3) **SoFr** AGIS-CH1 – an admissible instance of AGIS-CH1, which can be created by using some SoFr from the set {SoFr}, 4) **SoFr AGIS-CH1** – combining options 2) and 3). The given notations are quite conditional, since the concept of SoFr is not used without the concept of subject X, to which he (it) is applied. That is, the entry SoFr without subject X does not have a separate meaning - only as an abbreviation of the entry "Solutions Framework".

Many years of experience in the development of SpIS, as well as theoretical studies [3], allow us to assert that for each SpIS, including the AGIS class system, there is always a corresponding SoFr. It is quite easy to make sure that in every project of SpIS creation, there exists, explicitly or implicitly, the main triad of SpIS: so-called packages Products-Processes-Basics of SpIS elements. After all, some product element is created in each project: database, map, atlas, GIS, SpIS, etc. And a more or less complex product cannot be created without the processes of its creation.

Unfortunately, in information product creation projects, the main focus is very often on the product itself and almost no attention is paid to the processes of its creation. And then developers wonder why there is such a low percentage (20% according to various sources) of successful implementation of information projects. Recall the definition of the product-process dualism from Part 1: 1) without a process, it is impossible to create a product; 2) without a product, the process has no meaning [2]. The content of Part 2 is the Main Conceptual Provision 2, which is a process part of the main triad of AGIS-CH1 SoFr. It is formulated as follows: "AGIS-CH1 SoFr processes: The process of AGIS-CH1 creation should be the PIE (Projects Implementation Environment) portal".

Part 1 of the article describes the Main Conceptual Provisions received up to 2019, including Main Conceptual Provisions 0: "The usage of appropriate SoFr is mandatory for the success of activities to create a new electronic State Register of immovable CH" and 1: "SoFr X products: The first queue of the final X system should be AGIS-CH1 as an element of the set of acceptable AGIS". AGIS-CH1 must be an element of the set of acceptable AGIS [1] and is denoted by AGIS-CH1 $\in$ **AGIS** (or {AGIS}), AGIS-CH1 $\subset$ AGIS.

Until 2019, we did not pay attention to the processes of AGIS-CH creation, because then it was more important to decide on the ideas about the expected end product itself. In fact, a pilot project was launched in Vinnytsia [4] to get specific ideas about the product. AGIS-CH was then presented as a product with the help of the Conceptual Framework (CoFr) of AGIS-CH, and the processes of creation the individual components-products were to be determined then by "secondary" "inter-strata" βSoFr and αSoFr, but not γSoFr of AGIS-CH. γSoFr are used for systems more general than AGIS-CH. Such as, for example, AGIS-LT (LT - Large Territories) [5] and simply AGIS [1].

In 2021, we completed a project with the tentative title "Introduction into exploitation in the Ministry of Culture and Information Policy of Ukraine of the declarative electronic register of the CH". This implementation was carried out with the help of the Declaration Module (MD) of the CH developed by us in the pilot project [4]. The results of the completed project forced us to draw several very important conclusions.

1. In the activity of AGIS-CH creation, it is very difficult to rely on the application of some predetermined process of developing a particular product. However, the described own experience can help in shaping the process. After all, the existence of a process is mandatory; without it there will be no product.

2. We cannot quickly influence the level of IT training of the Customer. Otherwise, decision-makers will have arbitrary IT training. Perhaps before deciding on the development of the current phase of AGIS, a workshop should be held on the expected development processes.

In the described situation, taking into account the results of Part 1, we consider it necessary to reveal the following questions in Part 2.

1. Processes of creating not individual products, but systems similar to AGIS-CH, AGIS-LT and AGIS are possible.

2. Processes of individual product development and their advantages and disadvantages have been verified by our own practice, which should help to choose the most necessary ones, taking into account the processes of creating a system.

3. An introduction to the issue, if not methodology, then at least a defined strategy for guaranteeing the quality of developed products/results.

In Part 2, attention is paid to the so-called "process" provisions, which are dualistic to the "product" conceptual provisions formulated in 2019 and described in Part 1 of the work. Dualism means that for each of the "product" conceptual provisions, there must be dualistic "process" conceptual provisions.

Among the "process" conceptual provisions at the moment, we distinguish two groups: 1) development; 2) product quality assurance. Both processes refer to the process of creation and are part of the dualism "product $\leftrightarrow$ process" in the sense that the process or processes must explain how to create one of the products AGIS-CH1$\subset$AGIS-CH$\subset$AGIS-LT. We have chosen the group of product quality assurance processes as one of the important, but little-used processes of creating a SpIS. They are the second part of the processes that are shown in the second, verification part of the letter "V" in the V-model of creating AGIS-CH1.

Processes of creation of AGIS-CH1 and its parts

AGIS-CH1 creation processes

The first section of Part 2 examines the processes of AGIS-CH1 creating. For both this chapter and the next, it is important to understand the differences between a "system" and its "parts". Thus, in the SpIS very often, such notions as software of the SpIS or software technology for creating the SpIS are not distinguished.

The first problem here is the usage of software development standards in the "system environment". Scientific and technical literature is saturated with descriptions of standards and software development standards themselves. When creating a system, the question arises as to what to use: "system" standards or "software" standards as part of the system. Software standards have been successfully created and used for a long time, and "system" standards became popular much later. Therefore, it is necessary to coordinate something with something.

The second problem here is understanding the differences between software or software technology and the SpIS that is created using it. In our practice, we often encountered the phrase "ArcGIS GIS", which was most often unconsciously understood by its author as "GIS created using ArcGIS software technology". Unfortunately, we heard the original phrase from the decision-makers. It actually means "let's buy ArcGIS and thus have GIS". The problem here is wasted money in the case of spending only on software or even on software technology. After all, IS is much bigger than software or software technology.

In 2020-2021, we came to the conclusion that none of the known processes for creating the resulting IS/SpIS allows for the creation of such "system" products as AGIS-CH1 $\subset$ AGIS-CH $\subset$ AGIS-LT $\subset$ AGIS. Therefore, we had to pay attention not only to the process (method), but also to a more fundamental concept - the methodology of creating systems similar to AGIS. However, the methodology is an element of the SoFr AGIS-CH1 Basics package, which will be described outside of Part 2. At the same time, we could not separate the notion of methodology from the notion of "strategy" for creating/developing IS/SpIS, which also needs clarification.

Let us point out that the phrase "methodology strategy" makes sense. Without going into explanations, we note that we distinguish two methodologies: prescriptive and descriptive. Prescriptive methodologies can also be called normative or constructive. This is the methodology we propose to use to create AGIS-CH1. In the first strategy, the researcher constructs or builds an information technology (IT) meta-artifact as a general solution concept for solving a class of problems and then applies those solutions in a specific context. Thus, a constructive or, in other words, normative approach is used. In the second strategy, the researcher attempts to solve a specific customer problem by building a specific IT artifact in that specific context. Thus, a descriptive approach is used. Then constructive (prescriptive, normative) knowledge is extracted from the experience gained, which forms a general concept of a solution for solving a class of problems [6].

It would seem that in the case of a specific human activity, such as the creation/development of IS, the term "methodology" could be perceived or understood more easily, but this is not the case. In addition to the choice of strategy, the methodology of IS creation itself is a complex notion. To confirm this opinion, consider the definition from: “The IS creation methodology consists in organizing the IS construction process and in managing this process in order to guarantee the fulfillment of the requirements both for the system itself and for the characteristics of the development process" [7; p. 60].

Methodologies, technologies and design tools form the basis of any IS project. The methodology is implemented through specific technologies and their supporting standards, methods and tools, which ensure the implementation of IS life cycle processes".

The following definitions are given in the standard [8]:

- *life cycle* - the evolution of a system, product, service, project or other human-created entity, from conception to cessation of existence,

- *life cycle model* - a framework of processes and activities related to the life cycle that can be organized in stages, which also serve as a general reference point for communication and understanding,
- *process* - a set of interrelated or interacting activities that transforms inputs into outputs.

In the DSTU ISO/IEC/IEEE 15288:2016 (ISO/IEC/IEEE 15288:2015, IDT) standard, all processes are divided into four groups:

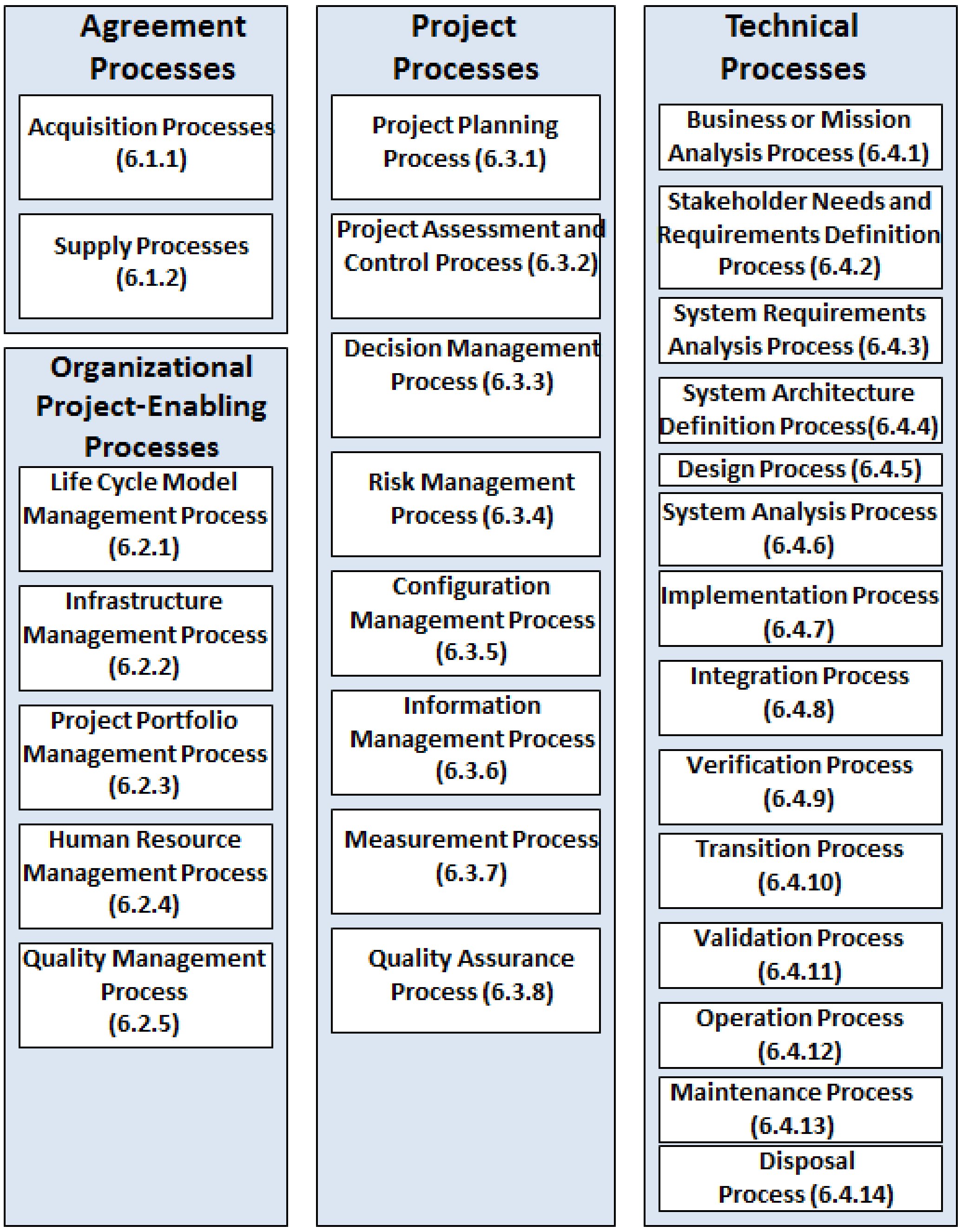

Agreement Processes
Acquisition Processes (6.1.1)
Supply Processes (6.1.2)
Organizational Project-Enabling Processes
Life Cycle Model Management Process (6.2.1)
Infrastructure Management Process (6.2.2)
Project Portfolio Management Process (6.2.3)
Human Resource Management Process (6.2.4)
Quality Management Process (6.2.5)
Project Processes
Project Planning Process (6.3.1)
Project Assessment and Control Process (6.3.2)
Decision Management Process (6.3.3)
Risk Management Process (6.3.4)
Configuration Management Process (6.3.5)
Information Management Process (6.3.6)
Measurement Process (6.3.7)
Quality Assurance Process (6.3.8)
Technical Processes
Business or Mission Analysis Process (6.4.1)
Stakeholder Needs and Requirements Definition Process (6.4.2)
System Requirements Analysis Process (6.4.3)
System Architecture Definition Process(6.4.4)
Design Process (6.4.5)
System Analysis Process (6.4.6)
Implementation Process (6.4.7)
Integration Process (6.4.8)
Verification Process (6.4.9)
Transition Process (6.4.10)
Validation Process (6.4.11)
Operation Process (6.4.12)
Maintenance Process (6.4.13)
Disposal Process (6.4.14)

Рис. 1 –Fig. 1. Systems life cycle processes

We can also refer to [9]: "The software development methodology is a set of methods used in the software life cycle and have a general philosophical approach". Today, there are not so many methodologies, especially complete ones, that is, those that take into account all stages of the software and IS life cycle. It is the methodology that determines which languages and systems will be used for the creation/development of software/IS and, in many respects, recommends which technological approach will be used.

SpIS/GIS is a special type of IS, so everything said above about methodologies for creating/developing IS is also true for SpIS/GIS. Unfortunately, the definitions of both components of the IS ◁— SpIS/GIS[1] relation are ambiguous. Much has been said about the ambiguity of GIS definitions in works [10], [5] and in the references given there. However, more important for us are the differences in the definitions of such terms as IS and SpIS/GIS in the "narrow" and "broad" understandings. For these terms, we have been using the definition from [11] for a long time, namely: 1) Information system in the broader sense (ISb) is a set of all formal and informal representations of data and actions with them in the organization, including the exchange associated with the first and second, as internal as well as with the external world; 2) Information system in the narrow sense (ISn) is computer-based subsystems designed to provide registration and support services for operation and management of the organization.

It is easy to see from the quote below that A.V. Zatonskyi [12; p. 3-5] chose the definition of IS as the main one, which, in fact, details the definition of ISb from [11], and the "traditional" definition of IS is a detailing of the definition of ISn.

The *business model* is a description of the enterprise as a complex system, with a given accuracy. Within the frame of the business model, all objects (entities), processes, rules of execution of operations, the existing development strategy, as well as criteria for evaluating the effectiveness of the system are shown. The form of

[1] —denotes generalization in the direction of the arrow. The reverse relationship is called specialization.

presentation of the business model and the level of its detailing are determined by the goals of modeling and the accepted point of view.

An *information model* is a subset of a business model that describes all existing (including non-documented) information flows in the enterprise, rules for processing and routing all elements of the information field.

The *information system* (IS) is the entire infrastructure of the enterprise involved in the process of managing all information and document flows, including the following mandatory elements:

- an information model, which is a set of rules and algorithms for IS functioning: the information model includes all forms of documents, the structure of directories and data, etc.;
- regulations for the evolutiont of the information model and rules for making changes to it;
- personnel resources (development department, involved consultants) responsible for the formation and evolution of the information model;
- software complex (SC), the configuration of which meets the requirements of the information model (the SC is the main driver and, at the same time, the IS management mechanism); in addition, there are always requirements for the SC supplier that regulate the procedure for technical and user support throughout the entire life cycle;
- human resources responsible for SC configuration and its compliance with the approved information model;
- regulations for making changes to the SC configuration and the contents of its functional modules;
- hardware and technical base that meets the requirements for SC operation (computers at workplaces, peripherals, telecommunication channels), system software and database management systems (DBMS);
- operational and technical human resources, including personnel for servicing the hardware and technical base;
- SC usage rules and user manuals, training regulations and user certifications [12]..

According to the existing tradition, it is accepted to call IS a software-hardware complex for processing (including automatic) user information, developing solutions, forming documents, etc., which is not correct, since this is only part of the normal functionality of IS. There are many definitions of the notion of system, but all of them imply the unity of the laws of movement (evolution) of the constituent elements. If we are talking about a system built by a person, then the laws of motion must be determined by specific goals - for example, those already listed. And, for example, software in the absence of an information model (in the context of this issue) is devoid of its own evolution laws and is no more than a necessary tool for building a system.

IS provides part of the information processing, so the global goal of IS should be to ensure that it contributes as much as possible to the organization's goals through the use of information technology (IT)".

The methodology for creating AGIS class systems developed by us is based on patterns and is normative. This means that a specific SpIS in the broader or narrow sense is created using appropriate patterns, and if such patterns have not yet been identified, then appropriate normative models are used.

There are three variants of this methodology, which we call: 1) Atlas Extension AtEx (according to the epistemological hierarchy it is "bottom-up"), 2) GeoInformation Extension GIE (according to the epistemological hierarchy it is "top-down"), 3) Combined. The mock-up of the Atlas Extension AtEx methodology is described in [13].

Almost twenty years ago, we developed not only the GeoSolutions Framework (GeoSF) method, [14], but also GeoSF means [15], see **Fig. 2**. The cited articles indicate that the method has not changed for almost 20 years, and the means need updating. Analyzing them, we can see that in modern conditions it is quite simple to update the TriNet portal software solution, which was the basis of GeoSF means. Therefore, we consider it expedient and possible to implement the PIE (Projects Implementation Environment) portal with an update of the portal solution and with the specialization of AGIS-CH1 development processes. It will be recalled that the

GeoSF portal (means) [15] was proposed almost twenty years ago for the construction of PIE execution of a group of interconnected projects.

First, it was necessary to install the GeoSF portal on the intranet/extranet/Internet with the initial values of all the patterns and templates for the projects that were planned to be executed. Three groups of users responsible for different parts of the project were created: readers, authors and coordinators, and specific users belonging to these groups were determined. After that, PIE was continuously updated by adding new results that belonged to one of the five GeoSF packages. That is, the project implementation process was reduced to the iterative development of the PIE, and with it the resulting ISb.

In our case, ISb coincides with AGIS-CH1. The integrated echeloned AGIS-CH1 will probably be developed by several enterprises. The activities of these enterprises can be divided into "project" and "daily". "Project" activity can be organized according to the Project Solutions Framework (ProSF). "Non-project" or "daily" activities can be organized according to ComSF (Company SF, GeoSF=ProSF∪ComSF). The non-project activities of these enterprises are less important in this context, so we can ignore them. If we assume that AGIS-CH1 is some part of NSDI - Cultural SDI (CSDI), then it is appropriate to mention the 4th dynamic principle of NSDI [3]: D4. CoFr CSDI as a constructor of Spatially Enabled Society (SES) in Ukraine. **Fig. 2** explains the usage of this principle. At the same time, it explains our pattern-based methodology for creating SpIS in general and AGIS-CH1 in particular. A more complete description of the methodology is outside of Part 2.

We know of only one complete declarative methodology for creating IS/SpIS. It is based on the waterfall model, therefore it is called "waterfall". The waterfall methodology is not only the most famous. It is also very important because in Ukraine the usage of the waterfall model is mandatory when creating state IS and software. Previously, there was even a Unified set of standards and guidance documents for automated systems, which we designated GOST 34.***. At the moment, some of these standards and documents have been updated and are Interstate

Standards. For example, GOST 34.602-2020 - Technical task for the creation of automated systems. Complex of standards for automated systems.

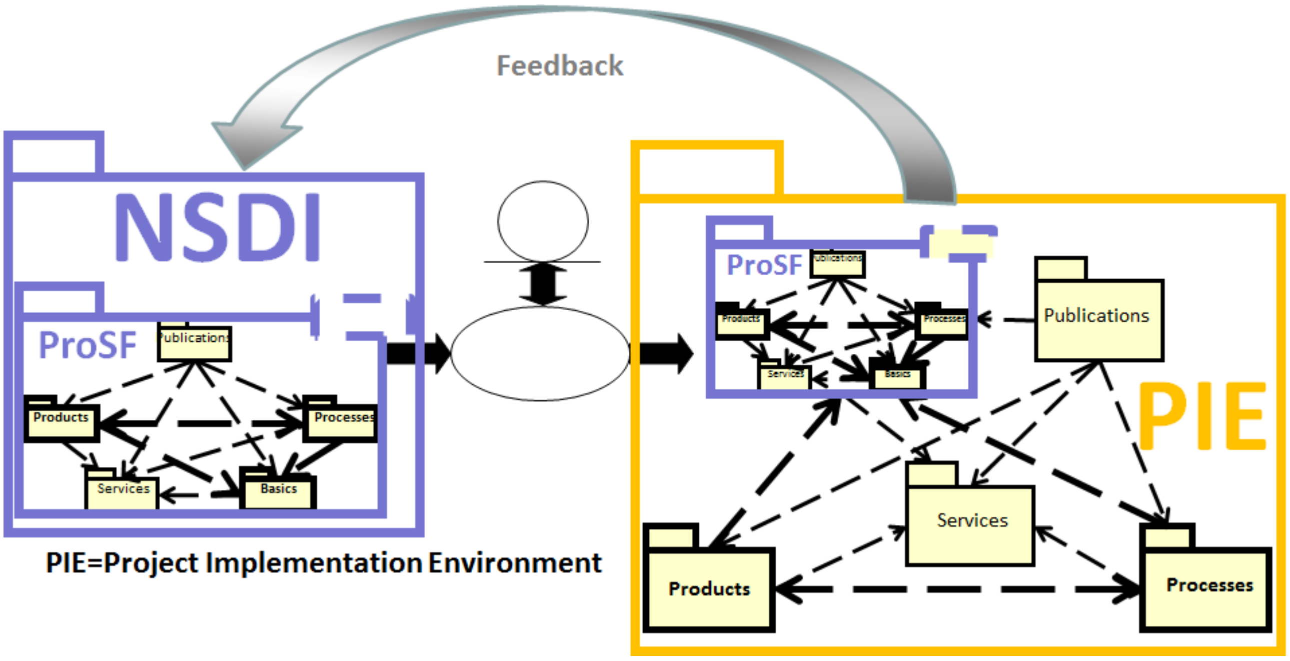


Fig. 2. The principle of using ProSF for the development of the PIE portal

We refer to the current Decree of the Cabinet of Ministers of Ukraine No. 121 of February 4, 1998, Kyiv "On approval of the list of **mandatory**[2] stages of work during design, means of informatization" {With changes introduced in accordance with Decrees of the Cabinet of Ministers N 1758 dated 12.27.2001, N 1191 dated 11.11.2009, N 675 dated 07.21.2010, N 915 dated 08.31.2011}. In the actual title and text of the Decree, the words "systems and means of automated processing and data transmission" have been replaced by the words "informatization means" in accordance with the Decree of the CM No. 1191. From this, we conclude that the Decree also refers to the project of AGIS-CH1 creating. The specified Decree lists the following mandatory etaps (stages) of works:

- Definition the need for informatization means and studying the issue of the possibility of their modernization to ensure the performance of the necessary functions.

[2] Highlighted by the authors

- Development of a technical and economic rationale for the creation or modernization of informatization means.
- Justification of the need to use personal non-property and (or) property rights of intellectual property for informatization means.
- Development of a technical task (Terms of Reference) for the creation or modernization of informatization means.
- Development of a technical and working or techno-working project for the creation or modernization of informatization means.
- Conducting training of specialists to ensure the functioning of informatization means.
- Performing commissioning works.
- Conducting tests of created or modernized informatization means.
- Introduction of informatization means into operation.
- Execution of work on the maintenance of informatization means in accordance with warranty obligations.
- Post-warranty maintenance of informatization means.

The processes of individual components of AGIS-CH1 development

The following subsections describe four software/IS development processes that we used in practice, so we will insist on our conclusions. This, on the one hand, means that we have a lot of additional evidentiary information. On the other hand, we consider it necessary to recommend our conclusions to those who will create AGIS-CH1. Perhaps our experience will help determine the processes of creation/development of the first queue of AGIS-CH or similar systems.

Waterfall or Cascade model

The waterfall model of IS/software development is also called Cascade. In it, the creation/development process looks like a flow that successively goes through the stages of requirements definition, design, implementation (construction and implementation), testing and debugging, integration (installation) and support. It is also applied to IS development. The article [17] is often cited as the source of the name and essence of the model. Royce described the basic concepts of what is now commonly called the "cascade" model and discussed the shortcomings of this model. There he also showed how this model can be refined to an iterative model. The original had:

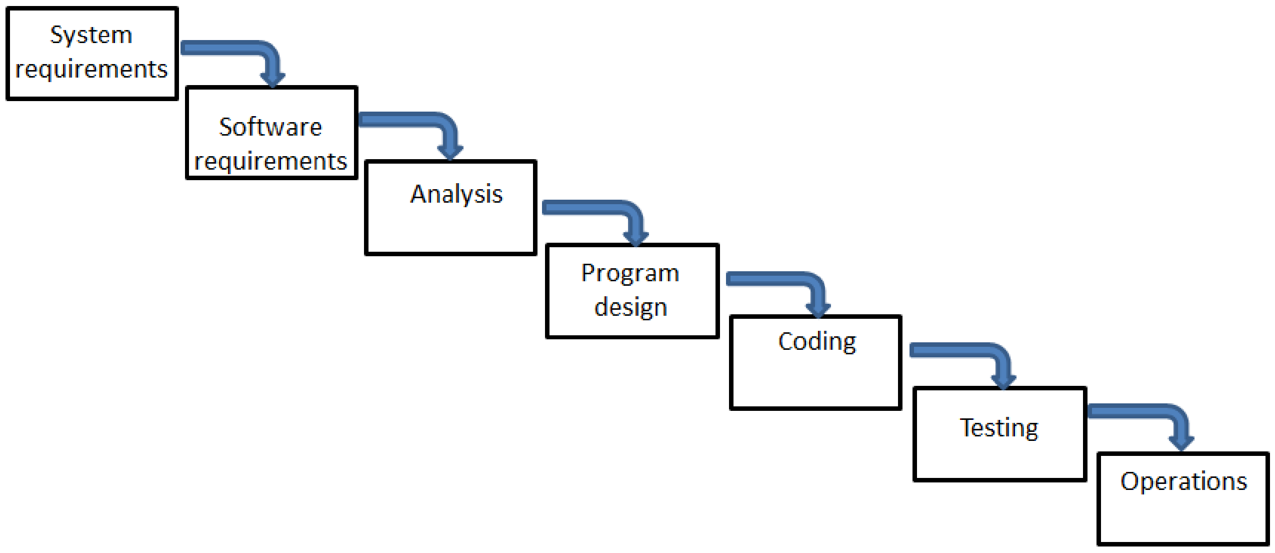


Fig. 3 Waterfall model according to [17]

The transition from this stage to the next is carried out only after the full and successful completion of this stage. First, the "requirements definition" stage is completely completed, the result of which are the requirements for the system in general and for the software in particular. After the requirements are fully defined, there is a transition to analysis, which should be perceived as preliminary or conceptual design. After that, logical design is carried out, during which documents are created that describe in detail to programmers the method and plan for implementing the documented requirements. After the design is completely completed, the programmers implement (coding) the resulting project. Although it is not shown in **Fig. 3**, after implementation, the integration of individual components takes place, which in large projects are often made by different teams of programmers. After the implementation and integration are completed, stages are performed, which in the V-model are called "validation" [8, 19]. In particular, the product is tested and debugged; at this stage, all the shortcomings revealed at the previous stages of development are eliminated. After that, the software product is implemented and its support is provided – implementation of new functionality and elimination of errors [17, 19].

We can talk not just about the process, but about the creation/development methodology based on the cascade model. This methodology is quite often criticized for lack of flexibility and for declaring formal project management as an end in itself to the detriment of terms, cost and quality. Nevertheless, when managing large projects, formalization was often of great value, as it could dramatically reduce many of the project's risks and make it more transparent. Therefore, even in the PMBOK (Project Management Body of Knowledge) 3rd version, only the "cascade model" methodology was formally established and no alternative options, known as iterative project management, were proposed.

Starting with the PMBOK 4th version [18], it was possible to reach a compromise between methodologists who favor formal and progressive project management with methodologists which rely on flexible iterative methods. Thus, since 2009, the Project Management Institute (PMI) formally offers as a standard a hybrid version of the project management methodology, which combines both the advantages of the waterfall methodology and the achievements of iterative methodologists.

V-model of development

We used this model in the project "Information and Analytical System for provision of urban planning activities" (IAS UPA) of the Kyiv city in 2018.

It so happened that we joined the project "from the inside", immediately before the implementation stage with limited influence on such components of the project as the plan, the execution team, the tools (ArcGIS), etc. Many important project decisions were made before us and we had to follow them. In the project, we used the V-model for development and quality assurance. Of course, the V-model has been simplified, which is a consequence of the initial analysis of the project (**Fig. 4**). Note that the V-model is described in the monograph [19], but in the next edition of the same monograph there was no description of the V-model. We did not look for reasons for removal from the works of the specified authors, but there are such works as “The Death Of The V-Model”, Ed Liversidge, Jun 25, 2015 [20].

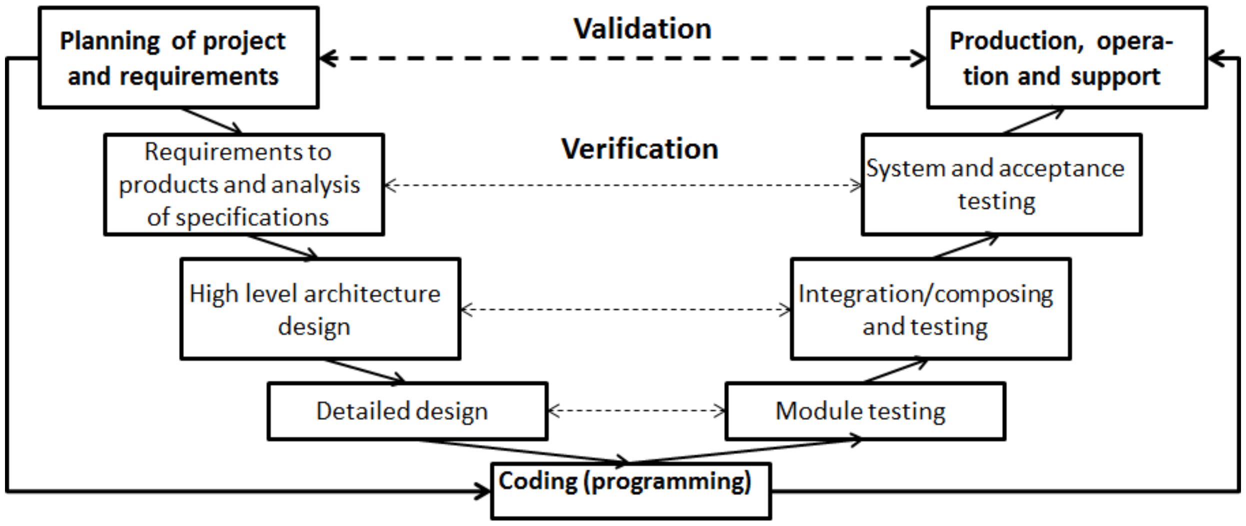

Validation
Planning of project and requirements
Production, operation and support
Verification
Requirements to products and analysis of specifications
System and acceptance testing
High level architecture design
Integration/composing and testing
Detailed design
Module testing
Coding (programming)

Fig. 4 - A simplified variant of V-model development for the IAS UPA project

On **Fig. 4** elements of the V-model highlighted in bold, which we focused on in the IAS UPA project. At the same time, this focusing applied both to the system as a whole and to each individual module that was developed in the project. The concepts of "validation" and "verification" additionally explain the priorities of the design works of the project (all works except coding (programming)).

The concepts of "verification" and "validation" are closely related to the processes of testing and quality assurance. They are often confused, although the differences between them are quite significant.

Verification is a process of evaluating the system or its components in order to determine whether the results of the current stage of development satisfy the conditions formed at the beginning of this stage. That is, whether the tasks, goals and terms of product development are fulfilled.

Verification - confirmation through the provision of objective evidence that specified requirements have been met.

Validation - confirmation by providing objective evidence that the requirements for a specific intended use or application have been met.

Validation is the determination of compliance of the developed software with user expectations and needs, system requirements. **Tabl. 1** helps highlight the key differences between these concepts.

Tabl. 1 – Verification and validation

№	**Verification**	**Validation**
1	Are we building the product *correct*?	Are we building the correct product?
2	Is *all* functionality implemented?	Is the functionality implemented *correctly*?
3	Verification is performed earlier and includes checking the correctness of writing documentation, code, etc.	Validation is performed after verification and is usually responsible for evaluating the product as a whole
4	Executed by developers	Executed by testers
5	Includes static analysis: code inspection, requirements comparison, etc	Includes dynamic analysis: running the program to compare its actual performance with specified requirements
6	It is based on an objective assessment of the compliance of implemented functions	A subjective process that includes a personal evaluation of the software's performance

With the help of validation, it is ensured that the "correct" product is created, which fully satisfies the customer. With the help of verification, it is possible to make sure that the product is created "correctly": following the necessary methods, tools and standards.

In practice, the differences between verification and validation are of great importance: the customer is more interested in validation (satisfaction of his own requirements); the performer, in turn, is concerned not only with compliance with all quality standards (verification) during product implementation, but also with the compliance of all product features with the customer's wishes. In connection with verification and validation, it is worth taking into account the following in **Fig. 5** representation the V-model of development.

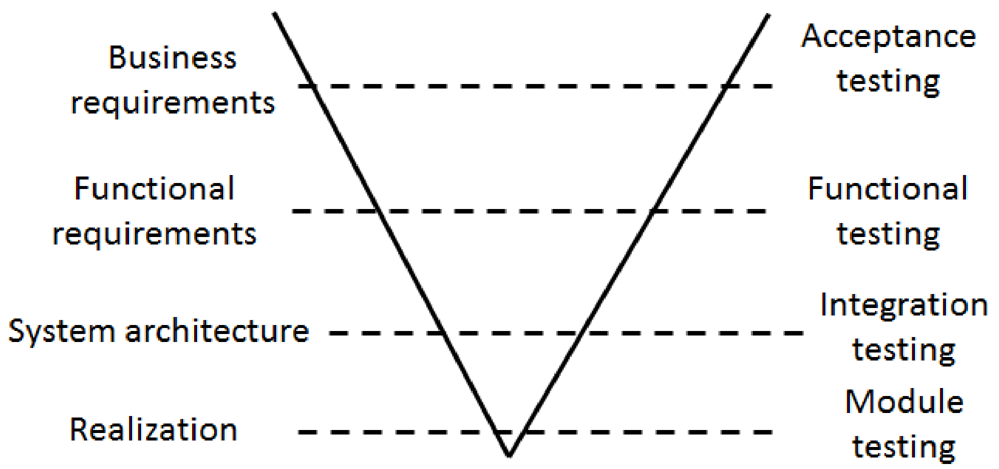


Fig. 5 Another variant of the V-model of development

Staged delivery model

The staged delivery model [21] has been successfully used by us in several projects for the development of large SpIS. An example is the Analytical and Information System (AIS MTS) of the second largest mobile operator in Ukraine - Vodafone (formerly - MTS).

Staged delivery avoids the problem of a waterfall model where the system is not developed until all parts of the system are developed. Once the architecture design is complete, the system can be developed and delivered incrementally. The staged delivery model is another life cycle model in which the software is presented to the customer in successive stages of improvement. Unlike the evolutionary prototyping model, when incremental delivery is used, it is known exactly what will be created and when it will be created. A feature of the staged delivery model is that the software is not delivered at the end of the project "in one fell swoop". It is delivered in successive stages throughout the project.

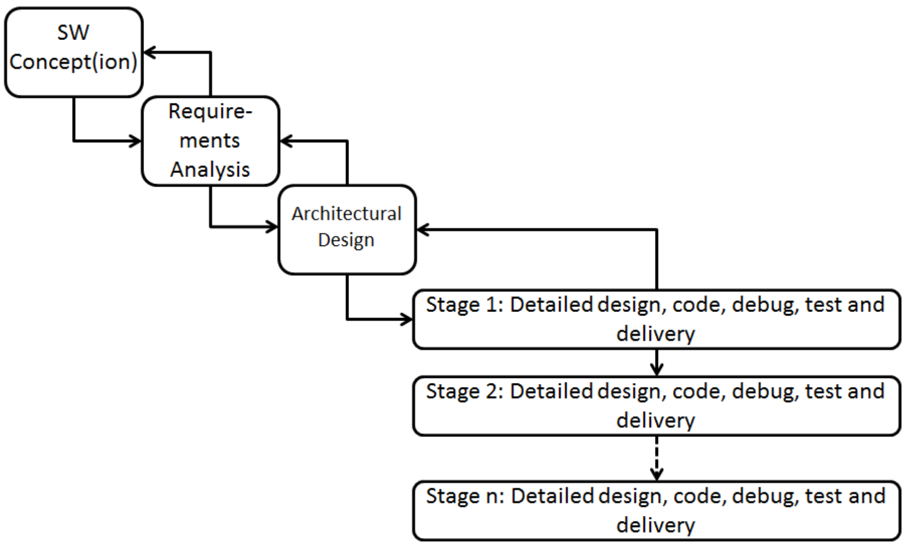

SW Concept(ion)
Require-ments Analysis
Architectural Design
Stage 1: Detailed design, code, debug, test and delivery
Stage 2: Detailed design, code, debug, test and delivery
Stage n: Detailed design, code, debug, test and delivery

Fig. 6 Staged delivery model

Fig. 6 shows how the model works. With the help of step-by-step delivery, the stages of the waterfall model are passed: definition of the software conception (concept in original), requirements analysis and architecture development of the entire program to be created. It then moves on to detailed designing, coding, debugging and testing at each stage. The main advantage of staged delivery is that it allows useful features to be delivered to customers earlier than when 100 percent of the project is completed at the end of the project. If the stages are carefully planned, the most important functionality can be delivered as soon as possible, and customers can start using the software at this stage. Incremental delivery also provides tangible signs of progress earlier in the project than less incremental approaches. Such signs of progress can be a valuable ally in keeping schedule pressure under control.

The main disadvantage of staged delivery is that it will not work without careful planning at both the management and technical levels. At the management level, ensure that the planned stages are important to the customer and that the work is distributed among the project team in such a way that they can complete their work in time for the stage deadline. At the technical level, you need to make sure that all the technical dependencies between the various components of the product are taken into account. A common mistake is to postpone the development of a component until stage 4, only to discover that the component planned for stage 2 cannot function without it.

Agile methodics

We cannot ignore the Agile methodics, although we have a negative experience using one of its methods. A few years ago, we were dealing with a project where it was necessary to perform another "sprint" for an existing system. At the same time, we did not develop the system architecture, and there was no corresponding architecture documentation either. We were led by the apparent simplicity of the additional functionality ("sprint") and took on the project of its implementation. Functional testing of this functionality was successful. However, acceptance testing of the entire system revealed unexpected troubles. We couldn't deal with them quickly and ended up failing the project. The section uses information from the monograph [22] and other sources.

Agile is a system of ideas and principles of "flexible" project management, on the basis of which the popular Scrum, Kanban, Lean, XP, and other methods were developed. The key principle is development through short iterations (cycles), at the end of each of which the customer (user) receives working code or a product.

17 American IT specialists from the state of Utah are responsible for the emergence of this flexible development methodics. Together with the "Manifesto for Agile Software Development", in which the term "Agile" was first used, they prescribed 12 principles of Agile development. Their essence boils down to the following key points that determine the nature of the flexible development methodology:

- People and interactions are more important than processes and tools.
- A working product is more important than comprehensive documentation.
- Cooperation with the customer is more important than agreement on the terms of the contract.
- Being ready for change is more important than sticking to the original plan.

Agile became the basis for a number of flexible methods, among which the most famous are Scrum, Lean and extreme programming (XP) [22].

Scrum is a method of flexible development based on Agile, which is based on a "sprint" - a period of time from 1 to 4 weeks, at the end of which a working version of the product must be obtained.

Lean is a method that grew out of the Toyota Production System. It is based on the philosophy of continuous improvement at all levels of the organization, where one of the key concepts is value (what the customer is willing to pay for) [22]..

Extreme programming (XP) is one of the Agile methods, where an important role is assigned to periodic planning with the involvement of the customer. It allows you to determine the shortcomings of the previous iteration, the priority of tasks, the desired functionality of the product, taking into account the wishes of the customer [22].

The advantages of the methodics include:

- short and clear iterations – development cycles last from 2 weeks to 2 months, after which the customer receives a working version of the product [22],
- a high degree of involvement of executors, organizers and project customers [22],
- at the head of the corner is the work product as the main indicator of progress — this can be considered both a plus and a minus, because in this case high demands are placed on the project team in terms of self-organization [22],
- risk minimization thanks to a flexible system of making changes [22],
- popularity of the method among developers of programs for business management [22].

Disadvantages of Agile, which organically "complement" its advantages:

- encouraging constant project change: the flexibility of product development may lead to the fact that it never reached the final version [22],
- increased requirements for the qualifications and experience of the team: in addition to development of the product, the team must analyze possible ways to improve the efficiency of its own work, continuously exchange information about the project, be motivated and self-organized. Project resources do not always allow to attract such specialists [22],

- philosophical nature of the methodology: Agile is not a clear instruction for action, but a whole philosophical concept. The team cannot mechanically apply the mechanics of "agile" development, it is necessary to adopt the key principles of the system [22],
- complexity of calculating the final amount of work: stimulation of changes and improvement of the final product leads to a floating value of the project cost [22].

Product quality assurance

In the standard [8] Processes of product quality assurance refer to "Technical processes" and are called "Verification process" and "Validation process" (**Fig. 1**). Between them is the "Transition process", which in practice is difficult to separate from the first two. For example, the processes of guaranteeing the quality of products belong to the second part of the stages of work, which are listed in the Decree of the CMU No. 121, although it is not easy to see it, namely:

- Conducting training of specialists to ensure the functioning of informatization tools [16].
- Performance of commissioning works [16].
- Conducting tests of created or modernized informatization means [16]..
- Introduction the informatization means into exploitation [16]..
- Performance the maintenance of informatization means in accordance with warranty obligations [16].
- Post-warranty maintenance of informatization means [16].

In specific projects, we used certain "relevant" standards to define different parts of the specified activity. To organize the activity of creating AGIS-CH1, we recommend starting with the Operational Concept, which is prepared using the DI-IPSC-81430 "OPERATIONAL CONCEPT DESCRIPTION (OCD)" standard. We quite successfully implemented this standard in the IAS UPA project. Further in the section, some facts from the Working Project/Operational Concept of the specified project are given.

We pay special attention to the use of the V-model. We have to say here that it is too early to "write off" this model in state projects in Ukraine. After all, the V-model allows streamlining the waterfall model (CMU Resolution No. 121) with quality assurance processes. Quality assurance processes refer to production processes that are "out of reach" in real projects. The use of the V-model makes it necessary to pay attention to guaranteeing the quality of the product from the very beginning of the project.

System quality assurance

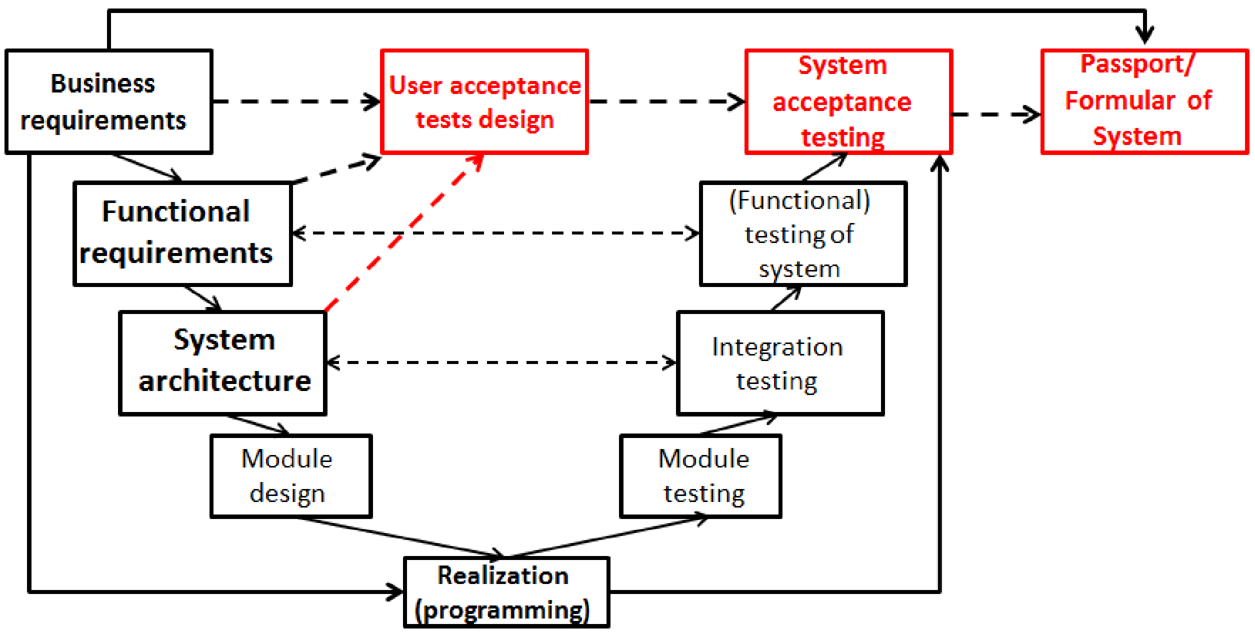
The desired system quality assurance is difficult to provide. However, in any case, we recommend developing the Passport and/or Formular of the existing system/situation according to GOST R 59795–2021 (**Fig. 7**). For development needs, Formular is more suitable, but for purchased components (such as ArcGIS), Passport is better. Therefore, the specified documents are combined in the document Passport/Formular of the existing system/situation.

As a result of the 1st stage, it is desirable to have an Operational Concept. The **1st substantive section of the Operational Concept** describes the existing system or situation. The main ones in this description should be Business requirements, Functional requirements and System architecture.

Using the description of the existing system/situation, you need to perform **Custom Acceptance Test Design**. Note that in normal development according to the V-model, the arrow from the Functional requirements is reversed, the arrow from the System Architecture does not exist. Therefore, red color was used.

The next step should be **Acceptance Testing** (after the fact, by any way possible); when performing which you also need to "look" into the Implementation (programming) and use the result in the Acceptance Testing (arrow from the Implementation).

Subsections 2.9.2-2.9.6 of the Formular (and/or Passport) of the system are prepared with the involvement of Business requirements, Functional requirements and System Architecture. Sections 2.9.7-2.9.8 of the Formular (and/or Passport) of the system are prepared using the results of the Acceptance test. The content of these subsections actually coincides (corresponds) with the content of the **2nd substantive section of the Operational Concept**, which describes the rationale and essence of changes to the existing system/situation. The resulting **Passport/Formular of the system** can even be considered a modernization of the architecture, since the production of this document will certainly lead to both technical and organizational changes to the system.



Business requirements
User acceptance tests design
System acceptance testing
Passport/ Formular of System
Functional requirements
(Functional) testing of system
System architecture
Integration testing
Module design
Module testing
Realization (programming)

Fig. 7 System quality assurance (on the example of the IAS UPA project)

Fig. 7 "reads" as follows.

The red color highlights the elements that need to be done in the project in relation to the existing system. Backdated Business requirements, Functional requirements and System architecture are not highlighted in red because they are included in the Operational Concept and/or System Passport/Formular.

Module quality assurance

Fig. 8 explains module quality assurance on the example of the IAS UPA project. It is performed for functional sets of tasks (FST or modules) that were developed in the project: assignment of postal addresses, assignment of building addresses, issuance of street renaming certificates, issuance of an extract from the urban planning cadastre, issuance of a construction passport for land development.

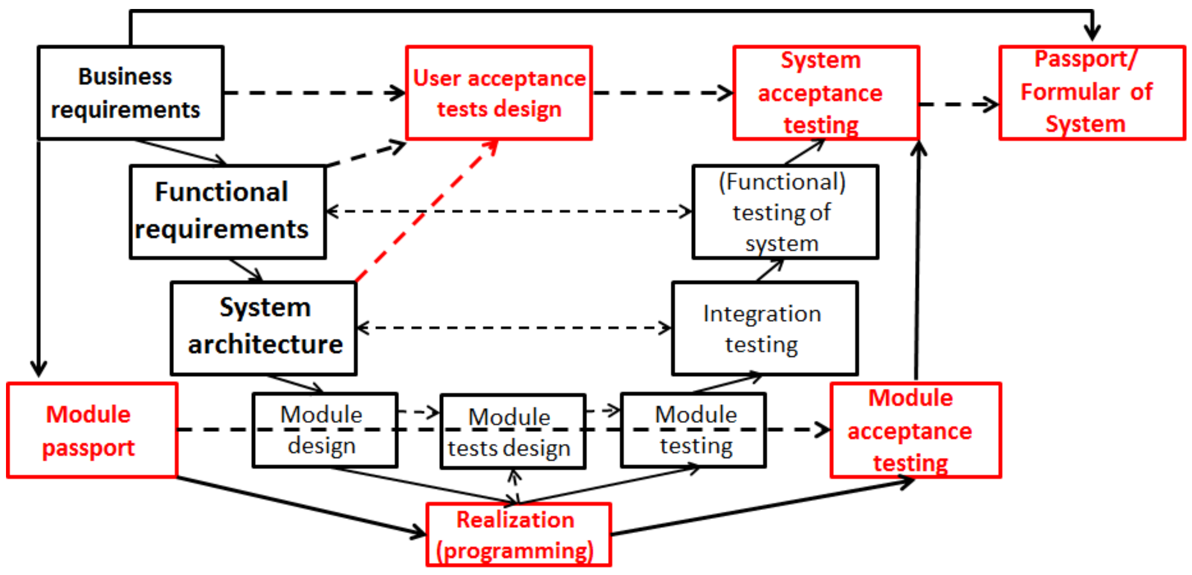

Business requirements
User acceptance tests design
System acceptance testing
Passport/ Formular of System
Functional requirements
(Functional) testing of system
System architecture
Integration testing
Module passport
Module design
Module tests design
Module testing
Module acceptance testing
Realization (programming)

Fig. 8 Modular quality assurance in the IAS UPA project, which complements the System quality assurance

The concept of **Module passport** is similar to the concept of Passport from GOST R 59795–2021, however, **Module acceptance testing** and its consequences are separated from it into "separate production" and are recorded in separate documents.

Module acceptance testing includes both module validation and verification, but validation is preferred.

The last, third element of the so-called **Passport Set of the module** is **Realization (programming)**, which includes the code design standard of the developer organization, as well as the code itself designed according to this standard.

Conclusions

The conclusions to Part 2 are recommendations based on the material presented. Recommendations are related to each other.

1. To create AGIS-CH1, you need to use the pattern-based methodology "AGIS-CH1 Solutions Framework". Elements of the methodology are described in Parts 1 and 2 of this article.

2. The processes of AGIS-CH1 creating should be selected using the standard DSTU ISO/IEC/IEEE 15288:2016.

3. When choosing the processes of AGIS-CH1 creating as a whole or its important system components, give preference to the V-model.

4. When choosing processes for the development of non-existent software, use the staged delivery model. When choosing existing software, give preference to open source software.
5. Creation of the AGIS-CH1 concept and its architecture is mandatory. The content of the DI-IPSC-81430 standard is recommended for the concept.
6. The means of coordination should be a set of specific templates, patterns and frameworks organized into a portal. The PIE (Projects Implementation Environment) portal should be similar to the GeoSF portal [15]. The portal is gradually (with the progress of the project) filled with the results of the project, which are obtained according to predefined templates, patterns and frameworks.
7. If not a method/methodology, then at least a defined strategy for guaranteeing the quality of the products/results is needed.

References

1. Rudenko, at al., (2018) Kulturna spadshchyna v Atlasnii heoinformatsiinii systemi staloho rozvytku Ukrainy: L.H. Rudenko, K.A. Polyvach, V.S. Chabaniuk ta in. / Za red. L.H. Rudenka.- Kyiv: Instytut heohrafii NAN Ukrainy, 2018.- 172 s. (Ukrainian)

2. Chabaniuk Viktor, Dyshlyk Oleksandr, Polyvach Kateryna, Pioro Vlad, Kolimasov Ivan, Nechyporenko Julia (2022). Main Conceptual Provisions of the Creation of an Electronic State of Immovable Cultural Heritage of Ukraine. Part 1.- Land management, cadastre and land monitoring, No. 2, 30 p. (Ukrainian, English)

3. Chabaniuk V. (2018) Relational cartography: Theory and practice.- Kyiv: Institute of Geography of the NAS of Ukraine, 2018.- 525 p. (in Ukrainian)

4. Ministerstvo kultury Ukrainy. Pilotnyi proekt vprovadzhennia elektronnoho obliku ob’iektiv kulturnoi spadshchyny (URL: https://www.kmu.gov.ua/ua/npas/pro-realizaciyu-pilotnogo-proektu-vprovadzhennya-elektronnogo-obliku-obyektiv-kulturnoyi-spadshchini, accessed 2022-jul-25).

5. Chabaniuk V., Polyvach K. (2020) Critical properties of modern geographic information systems for territory management.- Cybernetics and Computer Engineering, 2020, № 3(201), pp. 5-32.

6. Iivari J. Distinguishing and contrasting two strategies for design science research.- European Journal of Information Systems (2015), Vol. 24, Iss. 1, pp. 107–115.

7. Yzbachkov Yu.S., Petrov V.N. (2008) Ynformatsyonnыe systemы: Uchebnyk dlia vuzov.- SPb.: Pyter, 2-e yzd.- 656 s.

8. DSTU ISO/IEC/IEEE 15288:2016 (ISO/IEC/IEEE 15288:2015, IDT). Inzheneriia system i prohramnoho zabezpechennia. Protsesy zhyttievoho tsyklu system. System Engineering and Software. System Life Cycle Processes.- Natsionalnyi standart Ukrainy.- 83 (89) s.

9. Odyntsov Y.O. (2004) Professyonalnoe prohrammyrovanye. Systemnyi podkhod.- SPb.: BHV-Peterburh, 2004, 2-e yzd.- 624 s.

10. Chabaniuk V., Kolimasov I. (2020) Analysis of the Practical Use of Geoinformation Systems for Territorial Management and Determination of their Critical Properties.- Cybernetics and Computer Engineering, 2020, № 2(200), pp. 5-26.

11. Falkenberg E.D., Lindgreen P., Eds. (1989) Information System Concepts: An In-depth Analysis.- Amsterdam et al., North-Holland, 1989.- 357 p.

12. Zatonskyi A.V. (2014) Ynformatsyonnыe tekhnolohyy. Razrabotka ynformatsyonnыkh modelei y system: Ucheb. posobye.- M.: RYOR: YNFRA-M, 2014.- 344 s.10.

13. Chabaniuk V., Rudenko L. (2019) Relational geospatial technologies: background theory, practical example and needs in education, pp. 63-83 // in Geospatial Technologies in Geography Education. Edited by: de Miguel González Rafael, Donert Karl, Koutsopoulos Kostis.- Springer.- 219 p.

14. Chabaniuk V., Dyshlyk O. (2018) GeoSolutions Framework Reinvented: Method, pp. 115-138 // in Analysis, Modeling and Control. Vol. 3, Collection of Scientific Papers of the Department of Applied Nonlinear Analysis. Edited by prof. Makarenko A.S.- Institute for Applied System Analysis at the Igor Sikorsky Kyiv Politechnic Institute, Kyiv, 2018.- 250 p.

15. Chabaniuk V. (2018) GeoSolutions Framework Reinvented: Means (Portal Realization), pp. 90-114 // in Analysis, Modeling and Control. Vol. 3, Collection of

Scientific Papers of the Department of Applied Nonlinear Analysis. Edited by prof. Makarenko A.S.- Institute for Applied System Analysis at the Igor Sikorsky Kyiv Politechnic Institute, Kyiv, 2018.- 250 p.

16. Postanova KMU N 121 vid 4.2.1998 {iz zminamy N 1758 vid 27.12.2001, N1191 vid 11.11.2009, N 675 vid 21.07.2010, N 915 vid 31.08.2011} (URL: https://zakon.rada.gov.ua/laws/show/121-98-%D0%BF)

17. Royce Winston W. (1987) Managing the Development of Large Software Systems.- ICSE '87: Proceedings of the 9th international conference on Software Engineering, March 1987, pp. 328–338.

18. PMBOK Guide (2008) A guide to the Project Management Body of Knowledge (PMBOK® Guide, 4th Ed.). An American National Standard ANSI/PMI 99-001-2008.- Project Management Institute, 2008.- 459 (496) p.

19. Pressman Roger S. (2010) Software Engineering: A Practitioner's Approach.- (McGraw-Hill Series In Computer Science).- McGraw-Hill, 7th Ed., 2010.- 895 p.

20. “The Death Of The V-Model”, Ed Liversidge, Jun 25, 2015, (URL: https://harmonicss.co.uk/project/the-death-of-the-v-model)

21. McConnell Steve (1996) Rapid Development: Taming Wild Software Schedules.- Microsoft Press, 1996.- 647 (672) p.

22. Pressman Roger S., Maxim Bruse R. (2015) Software Engineering: A Practitioner's Approach.- (McGraw-Hill Series In Computer Science).- McGraw-Hill, 8th Ed, 2015.- 941 (971) p.

В.С. Чабанюк, О.П. Дишлик, К.А. Поливач, В.І. Піоро, І.М. Колімасов, Ю.В. Нечипоренко

ГОЛОВНІ КОНЦЕПТУАЛЬНІ ПОЛОЖЕННЯ СТВОРЕННЯ ЕЛЕКТРОННОГО ДЕРЖАВНОГО РЕЄСТРУ КУЛЬТУРНОЇ СПАДЩИНИ УКРАЇНИ. ЧАСТИНА 2: ПРОЦЕСИ

Анотація. *У Частині 2 описані процеси діяльності зі створення нового сучасного електронного Державного реєстру нерухомої культурної спадщини*

(КС) України. Вони є частиною методології, що базується на Каркасах Рішень (КаРі) АГІС-КС1, де АГІС-КС1 позначає першу чергу ієрархічно структурованої Атласної ГеоІнформаційної Системи (АГІС) КС. АГІС загалом складається з чотирьох страт: Операційної (ω), Аплікаційної (α), Концептуальної (β) і Загальної (γ). Процеси, що розглядаються у статті, відносяться до αКаРі АГІС-КС1, який визначає діяльність між підсистемами АГІС-КС1 Аплікаційної і Операційної страт. Згадуються також процеси, які відносяться до βКаРі АГІС-КС1, який визначає діяльність між підсистемами АГІС-КС1 Концептуальної і Аплікаційної страт.

КаРі АГІС-КС1 визначається пакетами і відношеннями між ними «петради» Публікації-Продукти-Процеси-Основи-Сервіси АГІС-КС1. Пакети Продукти-Процеси-Основи АГІС-КС1 і відношення між ними називаються головною тріадою КаРі. Ця тріада є основою Головних Концептуальних положень 1-3. Для останніх справедливі такі співвідношення: КаРі.Продукти – положення 1, КаРі.Процеси – положення 2, КаРі.Основи – положення 3.

У Частині 2 рекомендуються процеси розроблення і гарантування якості діяльності зі створення АГІС-КС1 із КаРі АГІС-КС1.Процеси. Ці рекомендації є фактично Головним Концептуальним положенням 2.

***Ключові слова**: Каркас Рішень (КаРі), Атласна ГеоІнформаційна Система (АГІС), Державний реєстр нерухомої культурної спадщини, процеси розроблення і гарантування якості.*