DOI: 10.17323/2587-814X.2024.4.61.80

Development of a high-level design of an analytical software complex for an enterprise that provides end-to-end planning

Natalya N. Seredenko
E-mail: Seredenko.NN@rea.ru

Plekhanov Russian University of Economics, Moscow, Russia

Abstract

Currently, many companies engaged in the production of medium-term turnover goods, are faced with the need to create a high-level design of a software complex that allows them to support a full cycle of sales planning, production, logistics and marketing campaigns. This is due to the economic development of the enterprise and the integration of independent software systems/modules that allow for the implementation of limited business processes without connection with related functions and business processes of the enterprise. Thus, enterprises find themselves in a situation where various departments have implemented a disparate set of information systems and software modules within which local accounting and analytics of various operations are carried out, while the software and analytical complex as a whole does not provide a complete, connected and cyclical planning process. This paper presents a model of requirements for functions, information objects and data flows, providing end-to-end planning, as well as an approach to identify missing objects of the existing information complex of the enterprise. An analytical network consisting of missing elements has been developed, taking into account the dependencies and interrelationships of information objects and software modules, which makes it possible to form a priority vector of the relative importance of software components. This vector represents a set of priorities for improvements to the enterprise software package and allows you to more effectively allocate the resources of the development team for the software implementation of missing functions, information objects and integration data flows between software modules.

Keywords: high-level design of an enterprise analytical software complex, end-to-end planning at the enterprise, data flow modeling, design of the enterprise information architecture, integration of enterprise software modules, functional composition of enterprise systems, Analytic Network Process

Citation: Seredenko N.N. (2024) Development of a high-level design of an analytical software complex for an enterprise that provides end-to-end planning. *Business Informatics*, vol. 18, no. 4, pp. 61–80. DOI: 10.17323/2587-814X.2024.4.61.80

Introduction

Companies engaged in the production of medium-term turnover goods are constantly looking for effective solutions to improve the accuracy of sales planning. The quality of planning depends not only on the analysis of the company's historical sales and market conditions, but also on the analysis of the activities of internal processes: production, marketing, logistics. In order for business plans and operational sales plans to correspond to the future reality as much as possible, it is necessary to take into account the processes of related functional divisions of the enterprise when planning [1–3]. Thus, the task of building a system of end-to-end planning at the enterprise is relevant, which represents the interrelation of processes reflected in the system of plans and reports and ensuring end-to-end planning of all production, economic, financial and other processes of the enterprise's activities [4, 5].

An information complex supporting end-to-end interconnected cyclic planning requires the implementation of a large number of complex functions, as well as a high level of integration solutions for the operational exchange of information objects between modules [6, 7]. At the same time, the life cycle of the development of the information complex of an enterprise often represents a disparate, time-spaced implementation of separate modules to solve local problems of a separate division. There are corporate information systems on the software market that contain modules for automating the activities of production, sales, marketing and logistics departments, as well as having a complete and consistent data model at the heart of their architecture which allows you to organize complex interconnected processes within a single platform. These are ERP systems, monolithic integrated solutions that are designed to connect different tasks and business processes. However, often, by the time a company realizes the need to build end-to-end interconnected processes and has the financial opportunity to implement large platforms, it already has a large information complex of various, loosely interconnected solutions. A one-time abandonment of all available software solutions in order to implement a single platform can lead to a halt in the organization's activities and is extremely difficult to implement. In addition, monolithic solutions are difficult to scale, they have weak fault tolerance and the risk of dependence on a single platform and technologies.

If a company has a specific strategy for the architectural development of its software package, the solution to the problem of building an information complex that provides end-to-end planning may have its own characteristics. As part of the current study, the question of the applicability of the most common software architectures to the effective solution of the problem of end-to-end planning is considered (*Appendix 1*). Based on this analysis it can be concluded that enterprises faced with the need to solve end-to-end planning tasks should develop an information complex in accordance with the principles of microservice or hybrid architecture [8, 9], or layered architecture. The use of cloud services is also possible, but the task of securing the internal contour of the enterprise and access to it from cloud services should be addressed as a matter of priority. If an enterprise has a disparate information complex by the time it realizes the need to solve the problem of end-to-end planning, it makes sense to simultaneously build a strategy for the development of the enterprise's information complex by choosing an architectural approach.

Theoretical research and practical developments in the field of end-to-end planning consider various approaches to the organization of the enterprise's software and analytical complex [10]. Most of them offer unified solutions, such as 1C:Integrated Automation [11], 1C:ERP Enterprise Management [4], Oracle Hyperion Planning [12, 13], IBM Planning Analytics [14], SAP Analytics [15]. However, scientific research on this topic does not examine the state of an organization when, at a certain point in development, various systems have already been implemented in key functional units in accordance with some selected architectural principles, and interaction between them has been set up.

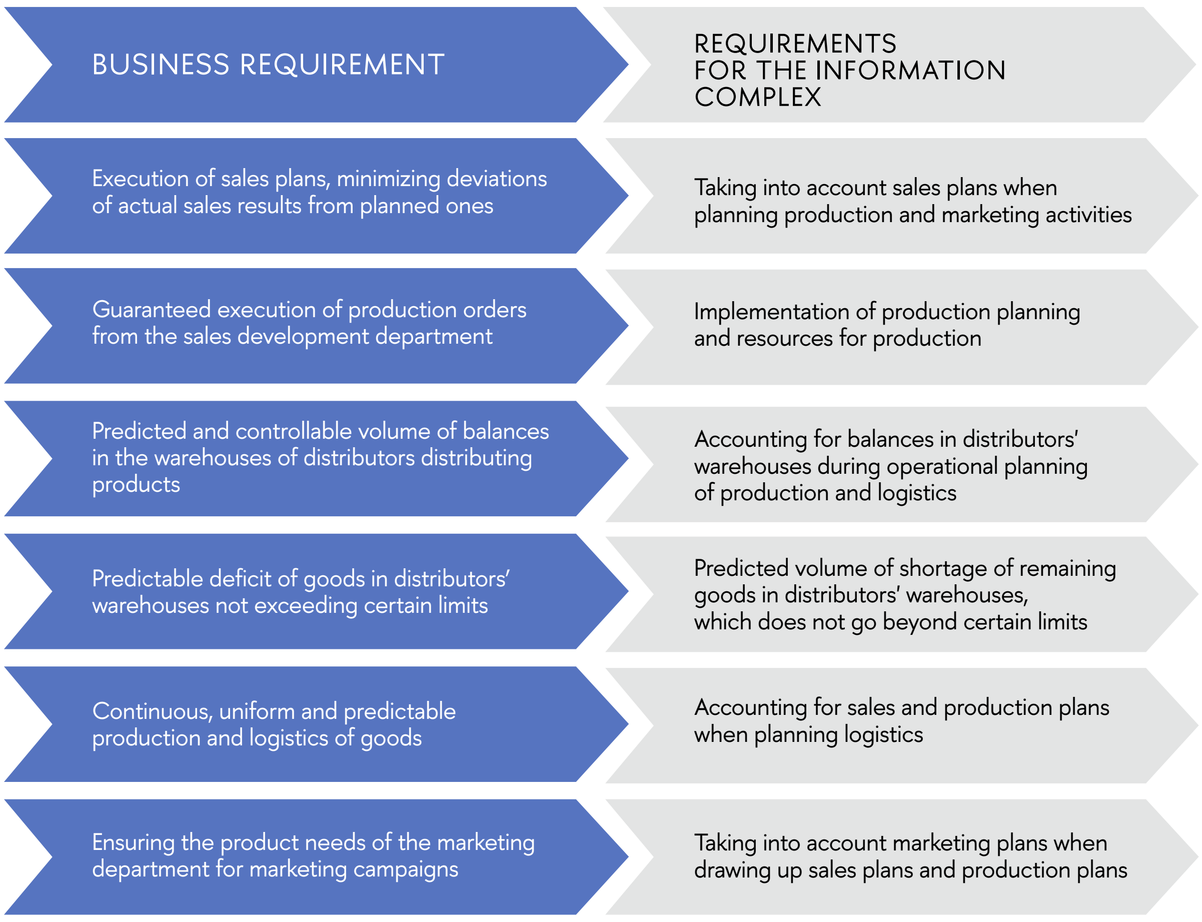
This paper examines enterprises that are faced with the need to refine an existing software package in order to increase the efficiency of the end-to-end planning process, and using microservice, hybrid, layered architecture, or a disparate software package. The process of finalizing the information complex of an enterprise is the formation and implementation of a portfolio of IT projects, in which each project is the configuration of the missing interaction (integration) between the systems / modules of the enterprise software complex.

According to the classifier developed in this study [16], the modernization of this kind of state of the organization is classified as an optimization task of finding competitive solutions to improve the infrastructure of the enterprise. This implies highlighting a set of unrealized tasks, as well as determining the priorities according to which it is advisable to implement them. This task is relevant for the company, since it leads to the achievement of strategic goals.

1. Statement of the problem

The development of an IT strategy in an organization is a key process and is of particular interest due to the huge role and high level of digitalization of most modern companies. The object of the study is companies whose information complex consists of separate modules, the interconnection of which is realized, among other things, through the transfer of information objects. Each microservice (module) includes its own stack of functions, technologies, ways of organizing data and software interfaces, depending on the software architecture of the enterprise [17]. The goal-setting of these organizations within the framework of the development of the IT landscape in the context of supporting end-to-end planning implies the development of a plan for the implementation of coordinated improvements to each microservice (module).

It is worth noting that the conceptual scheme of end-to-end planning may vary depending on the economic, production, financial and other processes of the enterprise [4]. Based on statistical observation of the main directions [18 example of the goals set by the company's management, 19], we formulate a possible in organizing end-to-end planning, and the resulting business requirements for the corresponding software package (*Fig. 1*).



BUSINESS REQUIREMENT	REQUIREMENTS FOR THE INFORMATION COMPLEX
Execution of sales plans, minimizing deviations of actual sales results from planned ones	Taking into account sales plans when planning production and marketing activities
Guaranteed execution of production orders from the sales development department	Implementation of production planning and resources for production
Predicted and controllable volume of balances in the warehouses of distributors distributing products	Accounting for balances in distributors' warehouses during operational planning of production and logistics
Predictable deficit of goods in distributors' warehouses not exceeding certain limits	Predicted volume of shortage of remaining goods in distributors' warehouses, which does not go beyond certain limits
Continuous, uniform and predictable production and logistics of goods	Accounting for sales and production plans when planning logistics
Ensuring the product needs of the marketing department for marketing campaigns	Taking into account marketing plans when drawing up sales plans and production plans

Fig. 1. Connection of the enterprise's business requirements in the field of end-to-end planning with the requirements for the information complex.

Achieving these goals is possible only if there is a software package that implements a full set of relevant functions, information objects and integrations.

In order to help organizations implement information technology development strategies that enable end-to-end planning, the current study proposes a methodology based on designing an effective high-level system design template and bringing the enterprise information complex to the developed template. According to the component methodology of business process reengineering proposed in the works by Telnov [20, 21], an enterprise system is a tuple of components $S = <G, E, E_n, T, F, R, Z>$. In relation to the area under consideration, let us clarify the definition of components:

G – a set of business goals related to the requirements for the information complex presented in *Fig. 1.* Within the scope of work, market and financial goals are relevant.

E – a set of information elements, namely, modules and systems that are part of the information complex that provides end-to-end planning.

E_n – a set of elements of the market and social environment. In the context of the problem being solved, they imply economic circumstances that force organizations to increase the efficiency of end-to-end planning in the enterprise.

T – a set of time periods characterizing the cyclical nature of end-to-end planning processes.

F – a set of functions included in the end-to-end planning process, implemented within the framework of many information modules.

R – a set of relationships that we will interpret as a set of interfaces (integration interactions) between information modules *E*. Each interface is a customized transfer of a separate information object from module E_n to module E_m.

Z – a set of patterns (strategies, methods) of the functioning of the information complex. As methods in the work, it is proposed to use the method of analytical networks by Saaty [22] for priority ranking of missing objects.

Thus, the statement of the problem within the proposed terms will sound as follows. We propose to identify a typical set of business processes (functions) F_{full}, divided into software modules E_{full}, and a complete typical set of transferred used data (information objects) R_{full} from the E_n module to the E_m, which the enterprise needs to organize end-to-end planning. Next, we propose to compare the actual set of implemented functions and transmitted information objects of the organization with the standard set of the model and select the set *F* and *R*, which represents the difference between the actual and standard set. As an example of a set *F*, one of the possible states of an enterprise's information infrastructure can be cited: lack of accounting and logistics planning functions; unrealized marketing planning function; formation of a production plan without taking into account the fact of sales. An example of a set *R* is the following set of missing information objects: lack of production plans in the logistics module, sales data in the marketing module, sales facts in the production module. Next, the task comes down to the implementation of the missing information objects in the corresponding modules, as well as the missing functions that use these objects. To increase the efficiency of implementation of this task, it is necessary to calculate the priorities of the portfolio of missing objects.

The stages of the described methodology are shown in *Fig. 2*.

2. Life cycle of the end-to-end planning process

In order to formulate requirements for data flows for a high-level design of an enterprise software and analytical complex that meets the stated goals, it is necessary to determine the requirements for the relationships between modules:

- Actual sales data must be taken into account when planning sales, marketing campaigns and production [23, 24].
- Plans for sales and marketing campaigns should be formed interdependently [25–27].
- Remains in the warehouses of departments performing the sales function (this can be their own chain of stores, sales through distributors or through any other sales channels [23]), must be taken into account when planning sales, planning production, planning logistics.
- Requests for production generated by the sales department must be included in plans for production orders, and the list of goods available for ordering for production must be available when generating requests for production [28–30].
- The balance of raw materials for production must be taken into account when planning production [28].
- The production plan should be taken as input in logistics planning [31].
- The fact of production and requests for delivery of goods must be taken into account when planning logistics.
- Additional applications for product release, generated by the marketing department based on the planning of marketing campaigns, must be taken into account when planning production [26, 32].

Thus, based on the listed requirements for data exchange between software modules, it is possible to create a data flow model that provides a sufficient

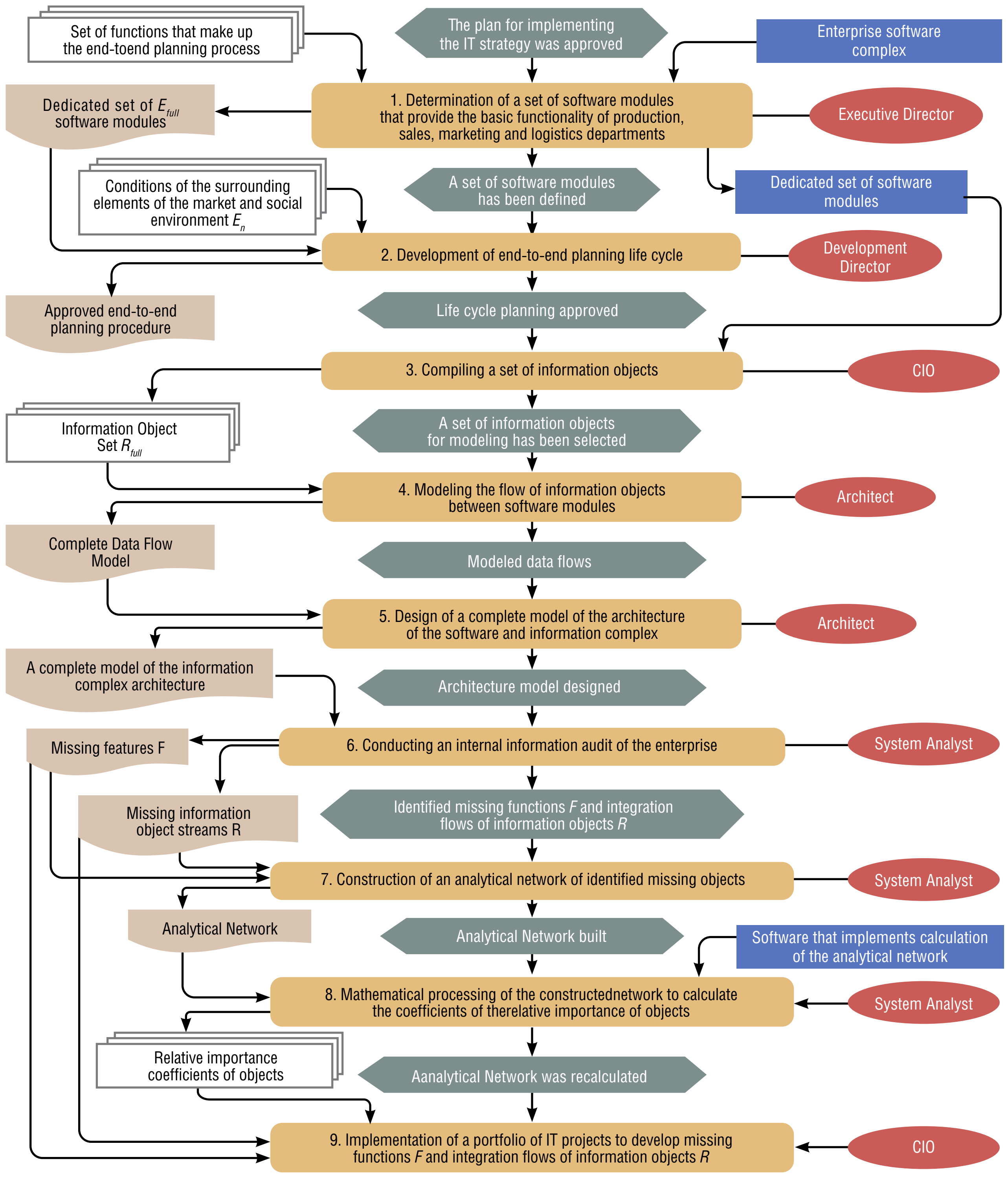

Set of functions that make up the end-toend planning process
The plan for implementing the IT strategy was approved
Enterprise software complex
Dedicated set of E_{full} software modules
1. Determination of a set of software modules that provide the basic functionality of production, sales, marketing and logistics departments
Executive Director
Conditions of the surrounding elements of the market and social environment E_n
A set of software modules has been defined
Dedicated set of software modules
2. Development of end-to-end planning life cycle
Development Director
Approved end-to-end planning procedure
Life cycle planning approved
3. Compiling a set of information objects
CIO
Information Object Set R_{full}
A set of information objects for modeling has been selected
4. Modeling the flow of information objects between software modules
Architect
Complete Data Flow Model
Modeled data flows
5. Design of a complete model of the architecture of the software and information complex
Architect
A complete model of the information complex architecture
Architecture model designed
Missing features F
6. Conducting an internal information audit of the enterprise
System Analyst
Missing information object streams R
Identified missing functions F and integration flows of information objects R
7. Construction of an analytical network of identified missing objects
System Analyst
Analytical Network
Analytical Network built
Software that implements calculation of the analytical network
8. Mathematical processing of the constructednetwork to calculate the coefficients of therelative importance of objects
System Analyst
Relative importance coefficients of objects
Aanalytical Network was recalculated
9. Implementation of a portfolio of IT projects to develop missing functions F and integration flows of information objects R
CIO

Fig. 2. Methodology for introducing architectural changes that ensure end-to-end planning into the enterprise information complex.

number of integration interactions to link all planning processes. The model of data exchange requirements is presented in *Fig. 3.*

In addition to the requirements for data exchange between modules, the model takes into account the requirements for the cyclicality and interconnectedness of business processes for sales planning, production, marketing campaigns and logistics.

3. Information architecture design

The previous sections described the prerequisites that lead large enterprises to carry out end-to-end planning of sales, production, marketing and logistics. A life cycle model of end-to-end planning is presented, and requirements for the software package are formulated. The current section contains the proposed model of a software and information complex that allows for a continuous planning process that links the activities of various divisions of the company, as well as providing a high level of analytical substantiation of plans.

In order to move from a data exchange model to a high-level system design model, a notation is needed that matches a set of requirements. Based on the analysis and generalization of existing types of diagrams and the experience of IT architects of various enterprises, we will highlight the requirements for notation based on the needs for solving the tasks:

1. The diagram must explicitly demonstrate the microservices of the enterprise, since they are the main structural elements of the software package.
2. The diagram should reveal the functional content of the modules.
3. Illustration of the complete set of information object flows between modules.
4. Demonstration that information objects belong to microservice databases. At the same time, it is important to understand in which microservices information objects are created, and in which they are used and not changed.
5. The diagram should allow you to demonstrate the nesting of modules.

A scheme that meets these requirements will make it possible to take into account key factors when making decisions on the development of a high-level design of an enterprise information complex and not to miss important entities. Such a representation will also make it possible to carry out an examination of the prioritization coefficients, which will be calculated as a result of applying the proposed methodology. This scheme can be a convenient auxiliary tool for managing and monitoring the implementation of a portfolio of projects to refine the information enterprise, since it corresponds to the language of both technical specialists and IT managers.

As a part of the study, existing notations were studied and popular solutions for visual modeling were considered. It was revealed that none of the structural diagrams, such as UML, STR, ERD, FDD [33, 34] satisfy all of the listed requirements. UML behavior diagrams [33] emphasize scenarios and business processes. Object-oriented UML and RUP diagrams do not demonstrate the structure of modules (microservices). IDEF and DFD [34, 35] emphasize system functions and the flow of information and physical objects, but provide little insight into the high-level design of the information complex. ARIS [36] focuses on event and function flows, but does not emphasize high-level systems design.

In order to satisfy the listed requirements for clarity and completeness, let’s take the UML deployment component diagram as a basis: it implies a look at the enterprise architecture from the point of view of deploying modules (microservices). We propose the following structural additions to the diagram notation: we will include a list of implemented functions inside each module; we will add a list of specific transmitted information objects and the directions of their transmission; let’s include an explicit indication of entities in the database and characteristics of master systems and recipient systems. We will call the resulting scheme “Information Architecture”.

Let us formalize the proposed definition. Information architecture is a diagram that shows the architecture of a high-level design system, including nodes such as modules (microservices) of the information

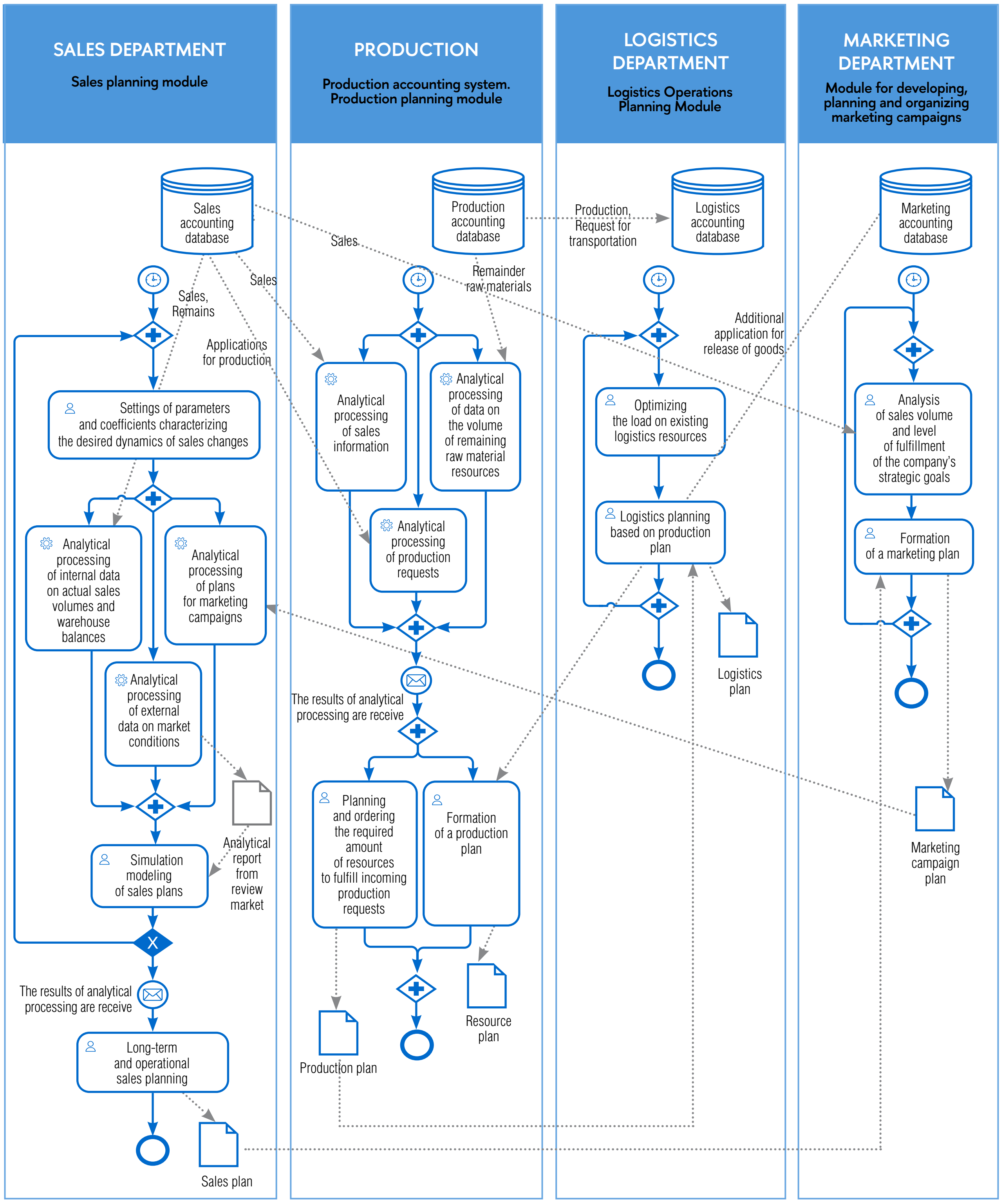

SALES DEPARTMENT
Sales planning module
PRODUCTION
Production accounting system. Production planning module
LOGISTICS DEPARTMENT
Logistics Operations Planning Module
MARKETING DEPARTMENT
Module for developing, planning and organizing marketing campaigns
Sales accounting database
Production accounting database
Logistics accounting database
Marketing accounting database
Sales
Sales
Sales, Remains
Applications for production
Production, Request for transportation
Remainder raw materials
Additional application for release of goods
Settings of parameters and coefficients characterizing the desired dynamics of sales changes
Analytical processing of internal data on actual sales volumes and warehouse balances
Analytical processing of plans for marketing campaigns
Analytical processing of external data on market conditions
Analytical report from review market
Simulation modeling of sales plans
The results of analytical processing are receive
Long-term and operational sales planning
Sales plan
Analytical processing of sales information
Analytical processing of data on the volume of remaining raw material resources
Analytical processing of production requests
The results of analytical processing are receive
Planning and ordering the required amount of resources to fulfill incoming production requests
Formation of a production plan
Production plan
Resource plan
Optimizing the load on existing logistics resources
Logistics planning based on production plan
Logistics plan
Analysis of sales volume and level of fulfillment of the company's strategic goals
Formation of a marketing plan
Marketing campaign plan

Fig. 3. Data flow model for high-level design of an enterprise software and analytical complex providing end-to-end planning.

complex; functions implemented inside the module; stored, processed and produced between measuring object modules.

Figure 4 suggests an information architecture model developed by the author which is an effective prototype for designing a software and information complex that ensures effective end-to-end planning in an enterprise.

Accompanying information on the revised notation:

1. Large rectangles represent systems/modules installed in the enterprise. It must be borne in mind that in real practice there is a different number of modules and distribution of functions between them. The proposed scheme is a unified average architectural solution.
2. Blocks with functions implemented within systems/modules. These functions are derived in section 3 based on the requirements for the information complex by enterprises producing goods with medium-term turnover.
3. Arrows emanating from some systems/modules included in other systems/modules are information flows transmitted within the framework of the described business process.
4. Databases in the form of "cylinders" and an indication of the information objects contained in these databases. It is necessary to keep in mind the following important factor: requirements for the integrity and a single version of master data at enterprises oblige strict monitoring of data source systems and data recipient systems. This aspect is important when developing integrations between systems, as well as when implementing an analytical warehouse and data model, on which analytical reporting is usually based. Therefore, this aspect is reflected in the information architecture, namely, information objects for which the system/module is the master system are depicted in a white rectangle inside the database, and objects for which the system/module is the recipient and within which these objects are not changed, are depicted in a gray rectangle inside the database.

Thus, we propose to include the following systems and modules in the information complex that ensures effective end-to-end planning:

1. **CRM system** (Customer Relationship Management). To support the production planning process, this system must contain two modules: *the Distributor Sales Accounting Module*, which collects data on distributor sales, and *the Production Order Formation Module*, which allows you to place product orders.

For the successful implementation of these functions in the modules of the CRM system, it is necessary to ensure the formation and transfer of the following information objects to consumer modules:

- ♦ Sales. Both sales to distributors and sales to end customers (through distributors or directly) can be accounted for and planned. To account for sales to end customers, separate CRM systems are being implemented that transmit data through distributors.
- ♦ Leftovers in distributors' warehouses.
- ♦ List of products available for production order.
- ♦ Application for production.

2. **Sales planning module**. This system is usually an analytical module that is part of the operating system, or a separate independent system, integration with which is configured through an ETL ("Extract, Transform, Load") solution. Taking into account the often-encountered complexity of the planning process in modern enterprises, we will assume that it is more correct to take this module into account as a separate independent module in the architecture requirements.

For the successful implementation of the listed functions in the *Sales Planning Module*, it is necessary to ensure the formation and transfer of the following information objects:

- ♦ Sales plans.
- ♦ Analytical summary with market analysis.

3. **Production accounting system**. This system is used to account for production processes, planning the volume of production of goods and raw materials. Unlike sales planning, which involves a large num-

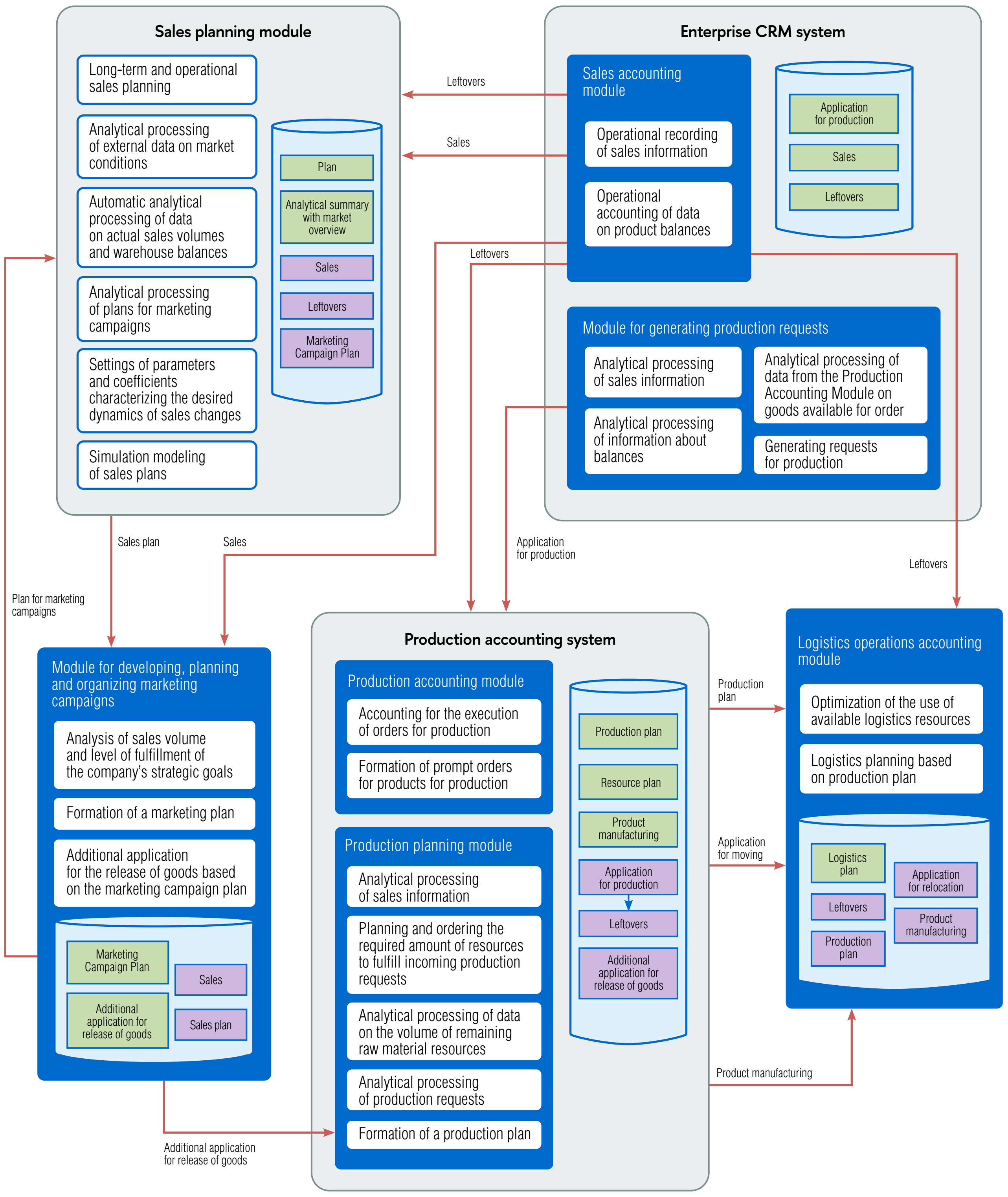

Sales planning module
Long-term and operational sales planning
Analytical processing of external data on market conditions
Automatic analytical processing of data on actual sales volumes and warehouse balances
Analytical processing of plans for marketing campaigns
Settings of parameters and coefficients characterizing the desired dynamics of sales changes
Simulation modeling of sales plans
Plan
Analytical summary with market overview
Sales
Leftovers
Marketing Campaign Plan
Enterprise CRM system
Sales accounting module
Operational recording of sales information
Operational accounting of data on product balances
Application for production
Sales
Leftovers
Module for generating production requests
Analytical processing of sales information
Analytical processing of data from the Production Accounting Module on goods available for order
Analytical processing of information about balances
Generating requests for production
Leftovers
Sales
Leftovers
Sales plan
Sales
Application for production
Leftovers
Plan for marketing campaigns
Module for developing, planning and organizing marketing campaigns
Analysis of sales volume and level of fulfillment of the company's strategic goals
Formation of a marketing plan
Additional application for the release of goods based on the marketing campaign plan
Marketing Campaign Plan
Sales
Additional application for release of goods
Sales plan
Production accounting system
Production accounting module
Accounting for the execution of orders for production
Formation of prompt orders for products for production
Production planning module
Analytical processing of sales information
Planning and ordering the required amount of resources to fulfill incoming production requests
Analytical processing of data on the volume of remaining raw material resources
Analytical processing of production requests
Formation of a production plan
Production plan
Resource plan
Product manufacturing
Application for production
Leftovers
Additional application for release of goods
Production plan
Application for moving
Product manufacturing
Logistics operations accounting module
Optimization of the use of available logistics resources
Logistics planning based on production plan
Logistics plan
Application for relocation
Leftovers
Product manufacturing
Production plan
Additional application for release of goods

Fig. 4. Model of the architecture of a software and information complex that ensures effective end-to-end planning.

ber of factors, such as market dynamics, external analytical reports, simulation modeling and marketing campaign planning, production planning is primarily based on approved sales plans, and therefore planning is most often a module built into the Production Accounting System. Thus, it makes sense to present the architecture of the production accounting system with two built-in modules: *Production Accounting Module* and *Production Planning Module*.

For the successful implementation of these functions in the Production Accounting System, it is necessary to ensure the formation and transfer of the following information objects:

- Production plan.
- Ordering products for production.
- Ordering of raw materials for production.
- Production of products.
- The rest of the products in production.
- The rest of the raw materials for production.

4. **Logistics operations accounting module**. This module registers logistics movements in accordance with logistics plans, as well as optimizes the utilization of available logistics resources.

For the successful implementation of the listed functions in the *Logistics Operations Accounting Module*, it is necessary to ensure the formation and transfer of the following information objects:

- Logistics plan.
- Application for relocation.
- Moving.
- Balance in logistics.

5. **Module for the development, planning and organization of marketing campaigns**. The operations performed in the marketing department are directly related to the production and sales process, as they directly affect the structure and volume of goods sold. Sales planning cannot be carried out with a high degree of accuracy without taking into account the activities of the marketing department.

For the successful implementation of the listed functions in the *Module for the development, planning and organization of marketing campaigns*, it is necessary to ensure the formation and transfer of the following information objects:

- Plan for marketing campaigns.
- Additional application for the release of goods.

Thus, it can be concluded that the listed modules dealing with operational accounting and planning of the activities of these departments should be integrated with each other to transfer and receive information objects, as well as the successful implementation of functions using these objects.

4. Sequence of implementation of the IT project portfolio

The formulated requirements for the high-level design of a software-analytical complex that provides end-to-end planning in an enterprise clearly demonstrate the logical architecture. This architecture will provide a continuous and interconnected planning process for various divisions of the organization, taking into account factors and information objects arising from adjacent divisions, which ensures a high level of analytical justification and high accuracy of planning.

It is assumed that the enterprise has the opportunity to compare its current information architecture and business processes with the template diagram presented and, if necessary, launch projects to implement missing integrations between its current software modules. This will ensure the availability of the necessary information objects in the appropriate modules, as well as the integration into current business processes of new missing functions that use these information objects.

It must be borne in mind that information systems integration projects are labor-intensive and complex tasks, each of which requires the allocation of a large amount of resources and a high level of qualifications of team members [37, 38]. Therefore, after checking with the proposed basic layout and identifying weak points in the current information architecture and business processes, it is necessary to create a list of missing integrations and implementations of functions in the systems and prioritize them. Future projects to implement dedicated integrations and features should be carried out in accordance with certain priorities, as this will ensure a more optimal allocation of resources and higher efficiency.

Thus, the next management task that needs to be solved is ranking projects for the implementation of missing integrations/functions. The main requirement for ranking is to take into account the dependencies and interdependencies that exist between software modules and business process elements implemented in different modules. The Analytic Network Process (ANP), which is a continuation of the Analytic Hierarchy Process (AHP), developed by Saaty, has the ability to take these dependencies into account [22, 39]. In accordance with this method, it is necessary to build a network of information objects, software modules and their relationships that arise when developing a complex high-level design. Let's build a complete view of the network structure of information objects and program modules (*Fig. 5*).

Objects included in the analytical network will be interpreted as follows:

- An Alternatives Cluster is a complete set of Information Objects for which the priority of relative importance needs to be calculated. In accordance with these priorities, the management of the enterprise will be able to draw up a portfolio of IT projects for implementation.
- Arrows from software modules to information objects mean that the objects influence the modules. This influence is expressed in the implemen-

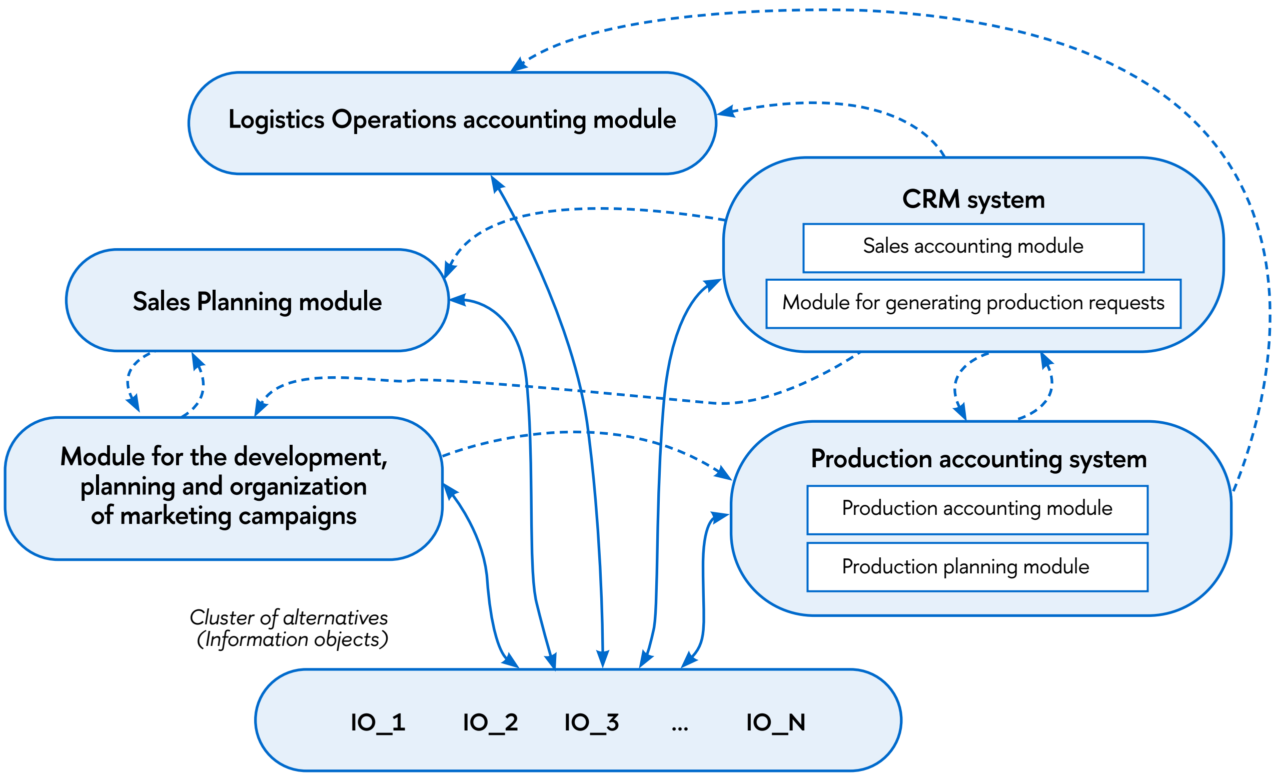

Logistics Operations accounting module
CRM system
Sales accounting module
Module for generating production requests
Sales Planning module
Module for the development, planning and organization of marketing campaigns
Production accounting system
Production accounting module
Production planning module
Cluster of alternatives (Information objects)
IO_1
IO_2
IO_3
...
IO_N

Fig. 5. Analytic network reflecting dependencies and relationships of program modules and information objects.

tation of new functions that can be implemented in these modules when new information objects appear in them.

- ♦ Arrows from information objects to software modules mean that the modules affect the objects. This impact is due to the complexity of implementing missing integrations and information object storage systems.
- ♦ Arrows between software modules indicate the flow of influence that software modules have on each other due to the presence of information flows between them.

Each dependence (each generated arrow) must be specified by a set of matrices of pairwise comparisons of the relative importance of the corresponding elements. When making pairwise comparisons, the following question is asked: "for a given network element and a pair of elements being compared, how much stronger is the influence of a given object from the pair on the element being evaluated compared to the other element?" Ratings are given on a nine-point scale, where a value of 1 point means that the compared elements are equivalent, and a score of 9 points means complete superiority.

Let $R = (R_1, R_2, ..., R_n, R_N)$ $n = (1, ..., N)$ – a set of alternatives (information objects), $E = (E_1, E_2, ... E_k, ..., E_K)$ $k = (1, ..., K)$ – a set of software modules within which it is necessary to make improvements to implement new business processes and integrations.

Relying on the structure of the constructed analytical network, we have the following types of dependencies:

- ♦ Dependencies of software modules relative to each other (type *I* dependencies, shown in *Fig. 5* with dotted arrows). When making pairwise comparisons of software modules relative to each other, the following question is asked: "which module software development has a higher priority for business at the present time?" Thus, we obtain a set of matrices of pairwise comparisons of relative importance $P^{I}_{(x,y)}$, where $x, y = (E_1, ..., E_K)$.
- ♦ Dependencies between alternatives and software modules:
  - Dependencies between software modules, taking into account the influence of information objects on them (Type *IIa* dependencies, shown in *Fig. 5* with solid arrows coming from a cluster of alternatives). When making pairwise comparisons of modules regarding the impact of alternatives on them, the following question is asked: "for this information object (alternative) in which module is it more important to implement integration for the subsequent implementation of missing functions in this module?". Thus, we obtain a set of matrices of pairwise comparisons of the relative importance of modules, the elements of which are denoted as $P^{IIa}_{n(p,t)}$ – is the result of comparing the relative importance of modules $p, t = (E_1, ..., E_K)$ relative to the importance of implementing the information object $n = (R_1, ..., R_N)$.
  - Dependencies between information objects (alternatives), taking into account the influence of software modules on them (dependencies of type *IIb*, shown in *Fig. 5* with solid arrows belonging to the cluster of alternatives). When making pairwise comparisons of alternatives regarding the influence of software modules on them, the following question is asked: "for this module, which information object (alternative) is more important to implement?". Thus, we obtain a set of matrices of pairwise comparisons of the relative importance of alternatives, the elements of which we denote as $P^{IIb}_{k(m,l)}$ – is the result of comparing the relative importance of alternatives $m, l = (R_1, ..., R_N)$ relative to the importance of implementing them in the software module .

Next, for a matrix of pairwise comparisons of the relative importance of software modules, the eigenvector $Z_{\text{own}.E} = (Z_{\text{own}.1}, ..., Z_{\text{own}.e}, ..., Z_{\text{own}.E})$, corresponding to the maximum eigenvalue of the matrix. A general view for calculating the eigenvector based on the definition of the eigenvector:

$$P^{I}_{(x,y)} \cdot Z^{I}_{(\text{own}.E)} = \lambda^{I}_{\max E} \cdot Z^{I}_{(\text{own}.E)}. \qquad (1)$$

The elements of the resulting vector are transformed according to the following rule:

$$w_m^I = \frac{Z^I_{(\text{own.}\,m)}}{\sum_{m=1}^{M} Z^I_{(\text{own.}\,E)}}. \qquad (2)$$

The vector $W^I = (w_1, \ ..., w_E)$ is a vector of coefficients of relative importance of software modules. Similarly, eigenvectors are calculated for matrices of pairwise comparisons of type *IIa* and type *IIb*. A general view for calculating eigenvectors:

$$P^{IIa}_{n(p,t)} \cdot Z^{IIa}_{\text{own.}\,n(p,t)} = \lambda^{IIa}_{\max n(p,t)} \cdot Z^{IIa}_{\text{own.}\,n(p,t)}; \qquad (3)$$

$$P^{IIb}_{k(m,l)} \cdot Z^{IIb}_{\text{own.}\,k(m,l)} = \lambda^{IIb}_{\max k(m,l)} \cdot Z^{IIb}_{\text{own.}\,k(m,l)}. \qquad (4)$$

The vector elements are subject to the following transformation:

$$w^{IIa}_{n(p,t)} = \frac{z^{IIa}_{\text{own.}\,n(p,t)}}{\sum_{m=1}^{M} z^{IIa}_{\text{own.}\,n(p,t)}}; \qquad (5)$$

$$w^{IIb}_{k(m,l)} = \frac{z^{IIb}_{\text{own.}\,k(m,l)}}{\sum_{m=1}^{M} z^{IIb}_{\text{own.}\,k(m,l)}}. \qquad (6)$$

The resulting vectors w_m^I, $w^{IIa}_{n(p,t)}$, $w^{IIb}_{k(m,l)}$ are further grouped into the supermatrix (the concept of a supermatrix was introduced in [22]). The coefficients of the relative importance of software modules will serve to weigh the blocks of the supermatrix. Next, the supermatrix is raised to the limiting power until the result stabilizes:

$$P^{\lim}_{SuperMatr} = \lim_{k \to \infty} \frac{1}{N} \sum_{k=1}^{N} P^k_{SuperMatr}. \qquad (7)$$

The values of the desired coefficients of the relative importance of information objects will be calculated in the appropriate blocks and can be used when planning resources for the implementation of missing software improvements.

For the practical application of the proposed methodology in the enterprise, various software can be used that implements the method of analytical networks. In the work of Latypova, a comparative analysis of software tools implementing AHP and ANP was carried out [40]. As part of the current work, the free educational software SuperDecisions was used [41, 42].

Conclusion

The paper presents an approach to the development of a high-level design of a software and analytical complex that provides end-to-end enterprise planning. A step-by-step methodology for introducing architectural changes is proposed, which will help enterprises engaged in the production of goods with medium-term turnover to carry out an internal audit of the software package that provides key functions of the enterprise, and to launch a portfolio of improvements that will ensure the implementation of the identified missing functions.

It should be noted that the proposed high-level design is a generalization and supports the average process of end-to-end production planning at a logical level. In real practice, deviations from the presented architecture may occur. These deviations are due to different initial levels of informatization of enterprises, differences in software systems, financial resources available to the enterprise, the choice of specific software solutions for each module, and other factors that are individual for each specific company. However, the architecture developed within the framework of the article is proposed to be used as a basis for conducting an information audit in order to optimize end-to-end planning business processes.

The following areas can be identified as directions for further development of the methodology:

- expansion of the functional composition of program modules, as well as development of the attribute composition of the described information objects;
- use of the described approach to evaluate the software implementation of missing functions in the enterprise software package;
- development of an approach to assessing the effectiveness of proposed improvements to the information complex;
- development of the proposed analytical network in accordance with a methodology based on an assessment of benefits, opportunities, costs and risks;
- generalization of the proposed approach to other key functions of the enterprise.

References

1. Yankovskaya V.V. (2023) *Planning at an enterprise*. Moscow: INFRA-M (in Russian).
2. Ilyin A.I. (2014) *Planning at an enterprise*. 9th ed. Moscow: INFRA-M; Minsk: New Knowledge (in Russian).
3. Telnov Yu.F., Kazakov V.A., Danilov A.V. (2021) Technology for designing innovative processes for creating products and services of a network enterprise using a knowledge-based i4.0 system. *Business Informatics*, vol. 15, no. 4, pp. 76–92. https://doi.org/10.17323/2587-814X.2021.4.76.92
4. Danilochkina N.G. (2019) Concept of end-to-end planning at industrial enterprises. *Current Problems of Socio-economic Development of Russia*, no. 4. pp. 38–41 (in Russian).
5. Hahn D. (1999) *Planning and control. The concept of controlling*. Moscow, Finance and Statistics (in Russian).
6. Bragina A.V., Vertakova Yu.V., Evchenko A.V. (2020) Development of end-to-end technologies for planning the activities of an industrial enterprise in the context of digitalization of the economy. *Production Organizer*, vol. 28, no. 1, pp. 24–35 (in Russian).
7. Minaev V.A., Mazin A.V., Zdiruk K.B., Kulikov L.S. (2019) Digital twins of objects in solving control problems. *Radio Industry*, vol. 29, no. 3, pp. 68–78 (in Russian). https://doi.org/10.21778/2413-9599-2019-29-3-68-78.
8. Kopeliovich D.I., Kurguz M.A., Lebedev V.V. (2022) Microservice architecture as a type of service-oriented architecture. *Scienceosphere*, no. 4-2, pp. 230–235 (in Russian).
9. Ibatulin M.Yu., Terentyev V.A. (2018) Application of microservice architectures for building enterprise business architecture in the era of digital transformation. Proceedings of the *XXI Russian scientific conference on Enterprise Engineering and Knowledge Management (EEKM 2018). Moscow, April 26–28, 2018.* Vol. 1, pp. 376–381 (in Russian).
10. Ford N., Richards M. (2020) *Fundamentals of software architecture: An engineering approach*. O'Reilly Media.
11. Kurzova Yu.A. (2022) Overview of the 1C program: Integrated automation. *1C-Business Architect*. Available at: https://www.1ab.ru/blog/detail/1s-kompleksnaya-avtomatizatsiya-obzor/ (accessed 06 July 2024) (in Russian).
12. *Certificate of state registration of a computer program No. 2022665892 Russian Federation. SSC. System-2 of business planning and budgeting based on the Oracle Hyperion Planning 11.1.2.4*, software package No. 2022664704, application 08/03/2022, publ. 08/23/2022; applicant Joint Stock Company "Siberian Service Company (in Russian).
13. Minyailo A.S. (2020) Study of information risks of Russian industrial companies in budget process management systems. *Corporate Economics*, no. 4(24), pp. 4–21 (in Russian).
14. Pirogov M.V., Abdulganiev A.N., Martinovich D.A. (2022) Application of multidimensional data warehouse technology (OLAP) based on the system for managing financial performance and enterprise activity IBM Planning Analytics. *Regional Economy Issues*, no. 4(53), pp. 144–162 (in Russian).
15. Nazarov D.M., Nazarov A.D., Kovtun D.B. (2020) Building technology and predictive analytics models in the SAP analytic cloud digital service. Proceedings of the *2020 IEEE Conference on Business Informatics (CBI 2020). Antwerp, June 22–24 2020.* Vol. 2. pp. 106–110 (in Russian). https://doi.org/10.1109/CBI49978.2020.10067
16. Bragina A.V. (2021) *Technological modernization of an industrial enterprise using end-to-end planning technology*. Dissertation of a Cand. Sci. (Econ.). Kursk (in Russian).
17. Smirnov M. (2017) Microservice architecture in the corporate IT landscape. *Open Systems. DBMS*, no. 4, pp. 38–41 (in Russian).
18. Terentyev A.V., Polyakov Yu.N. (2020) Concept of design and construction of the basic algorithm of the business planning process in small and medium-sized businesses. *Management: Theory and Practice*, no. 1-3, pp. 99–105 (in Russian).
19. Shataeva O.V., Akimova E.N., Nikolaev M.V. (2021) *Economics of an organization (enterprise)*, 2nd ed. Moscow, Berlin: Direct-Media (in Russian).

20. Telnov Yu.F. Fedorov I.G. (2017) *Enterprise engineering and business process management. Methodology and technology*. Moscow: UNITY-DANA (in Russian).

21 Telnov Yu.F. (2005) *Business process reengineering: Component technology* (2nd ed.). Moscow: Finance and Statistics (in Russian).

22. Saaty T.L. (2015) *Decision making with dependence and feedback: The Analytic Network Process*. Moscow: LENAND (in Russian).

23. Gusakov I.V. (2014) *Analysis and planning of sales in FMCG market companies*. Moscow: Nobel Press (in Russian).

24. Lehmann D.R. (2017) *Product management*. Moscow: UNITY-DANA (in Russian).

25. Zhukova T.N. (2021) *Management and organization of marketing activities*. Moscow: INFRA-M (in Russian).

26. Kotler P., Saunders D., Armstrong G., Veronica V. (2007) *Fundamentals of marketing*. Moscow: Williams (in Russian).

27. Toymentseva I.A., Mikhailov A.M. (2015) The influence of marketing and management decisions on the process of optimizing the enterprise budget. *Bulletin of the Samara Economic University*, no. 12(134), pp. 16–19 (in Russian).

28. Smurnov E.S. (2010) *Automation of material accounting*. Moscow: Laboratory of Books (in Russian).

29. Semenova N.V., Baygulova A.A. (2014) *Fundamentals of production management: electronic training course*. Ulyanovsk: UlSU (in Russian).

30. Farakhutdinov Sh.F. (2021) *Modern trends and innovative methods in marketing research*. Moscow: INFRA-M (in Russian).

31. Gorsky M.A., Khalikov M.A. (2020) Models and methods for assessing the optimal size of an enterprise's production segment. *Bulletin of the Altai Academy of Economics and Law*, no. 1-1, pp. 23–32 (in Russian).

32. Oyner O.K. (2022) *Marketing performance management*, 2nd ed. Moscow: Urait (in Russian).

33. Petrova I.R., Fakhrtdinov R.H., Suleymanova A.A., Razzhivin I.O., Fazulzyanov A.G. (2018) *Methodology of object-oriented modeling. The UML language*. Kazan: Kazan University (in Russian).

34. Freeman E., Robson E., Sierra K., Bates B. (2024) *Head First design patterns*. Saint Petersburg: Piter (in Russian).

35. State Standard of Russia (2000) *Methodology of functional modeling IDEF0. The guidance document*. Moscow.

36. Nazarova O.B., Novikova T.B., Maslennikova O.E. (2023) *ARIS: Theory and practice of business modeling.* Moscow: FLINT (in Russian).

37. Pervukhin D.V., Isaev E.A., Rytikov G.O., Filyugina E.K., Airapetyan D.A. (2020) Comparative analysis of theoretical models of cascade, iterative and hybrid approaches to managing the life cycle of an IT project. *Business Informatics*, vol. 14, no. 1, pp. 32–40. https://doi.org/10.17323/2587-814X.2020.1.32.40

38. Matveev A.A., Novikov D.A., Tsvetkov A.V. (2005) *Models and methods of project portfolio management*. Moscow: PMSOFT (in Russian).

39. Saaty T.L. (1993) *Decision making. The Analytic Hierarchy Process. Moscow: Radio and Communications* (in Russian).

40. Latypova V.A. (2018) A comparative analysis and a choice of tools implementing analytic hierarchy process. *Modeling, Optimization and Information Technologies*, vol. 6, no. 4(23), pp. 322–347 (in Russian). https://doi.org/10.26102/2310-6018/2018.23.4.024

41. Saaty R.V. (2016) *Decision making with dependencies and feedbacks. A tutorial for SuperDecisions software*. Pittsburgh: Foundation for Creative Solutions.

42. Creative Decisions Foundation (2024) *Super Decisions software*. Available at: https://www.superdecisions.com/ (accessed 06 July 2024).

Appendix 1.

Analysis of the applicability of software architecture for solving end-to-end planning problems

Architecture	Researches	Advantages	Disadvantages	Applicability for solving end-to-end planning problems
Monolithic architecture	Edsger Dijkstra formulated the principles of structural programming in the 1970s and 1980s. With the advent of personal computers and the development of the graphical interface, software has become more accessible and widespread. In the 1990s, monolithic architecture became the dominant approach for application development. The principles of monolithic architecture did not develop as a separate concept, but rather were applied and improved in the context of software development and evolution and were the prevailing approach. The very concept of «monolithic architecture» appeared later, when alternative approaches began to develop.	• Effective for small and simple systems • Easy to deploy • Stable operation	• Difficult to scale • Have weak fault tolerance • The risk of dependence on a single platform and technologies	Monolithic complexes are applicable to solving end-to-end planning problems only for small organizations in which the approach to planning is poorly developed and a small number of factors are taken into account when planning. Monolithic complexes can provide a certain process within their framework, but are extremely difficult to develop and integrate with external modules.
Event-driven architecture (EDA)	The development of event-driven architecture is related to object-oriented programming (EDA). In the 1970s, the concept was developed by Alan Kay and David Parnas. Further in the 1980s, Björn Stroustrup combined elements of procedural programming with an object-oriented approach. In the early 2000s, OOP was actively developing, and microservice architecture began to develop based on these principles.	• High overall adaptability • High performance • Scalability	• Difficult to implement due to asynchronous, distributed nature • It is necessary to solve problems of accessibility of remote processes, • The need to build logic for reconnecting the broker • Lack of atomic transactions for one business process	Event-driven architecture of information systems is used for basic enterprise tasks related primarily to operational accounting and building business processes. In order to design an information complex that allows solving end-to-end analytical planning problems, it is necessary to use approaches focused on the transfer and use of information objects, rather than on processing events/operations.

Architecture	Researches	Advantages	Disadvantages	Applicability for solving end-to-end planning problems
Microservice architecture	Microservice architecture is based on the concepts of Service Oriented Architecture (SOA). Microservices architecture is the result of the collective effort of many scientists, engineers and practitioners who contributed to its development. The main ideas were laid down in 2014 by Robert Martin (formulated the principles of modularity and independence in software development), James Lewis and Martin Fowler (defined the basic principles of this architecture), Fred George (proposed the use of the concept of "services" in the context of distributed systems), Dan Rosen (formulated the principles of "self-directed teams", which are an important element of microservice architecture). Modern Russian research on the topic of using microservice architecture belongs to Yu.F. Telnov, D.I. Kopeliovich, M.A. Kurguz, V.V. Lebedev, M.Yu. Ibatulin, V.A. Terentyev.	• Architecture provides flexibility and scalability • Independent deployment • Increased fault tolerance	• Challenges of development and deployment complexity • Increased control complexity • The need to implement interservice interaction	Microservice architecture is well suited to solve the end-to-end scheduling problem. It is within the framework of microservice architecture that it becomes possible to develop the complex in any direction, which means that the task of prioritizing the order of software development of new services and transferred information objects to ensure a comprehensive architecture is relevant.
Hybrid architecture	The concept of «hybrid architecture» arose as a result of a natural evolutionary process associated with the emergence of new technologies and an understanding of the advantages and disadvantages of both monolithic and microservice architectures. The beginning of the formation of hybrid approaches can be considered the late 1990s and early 2000s, when new technologies and architectural patterns began to appear: the spread of the Internet, the emergence of distributed systems, the development of containerization technologies. Important contributions to these methodologies were made by Ernest	• Allows for a gradual transition to microservices • Provides a flexible choice of optimal solutions for different parts of the system • Combines the advantages of monolithic and	• Difficult to control • Complexity of technical implementation • Compatibility issues between the components of the complex	Companies in a state of hybrid transition (either in a state of transition from one solution to another, or introducing new business models that are implemented based on microservice approaches and which integrate with monolithic applications) must solve the problem of prioritizing the development of modules for new services and order development of integrations for the exchange of information objects between systems.

Architecture	Researches	Advantages	Disadvantages	Applicability for solving end-to-end planning problems
	Clarke, who proposed the concept of "service orientation" in the 1990s, Martin Fowler, who identified SOA as an important design pattern in the early 2000s, and Jim Brandel developed the concept of "containers" in the 2010s-2020s.	microservice architectures		
Cloud computing	In the early 1980s, Ken Thompson created the first version of the multi-user UNIX operating system. Then virtual machines appeared, providing platform-independent programming environments. The term «cloud computing» appeared in the 1990s in the specifications of Compaq and Apple. In the first half of the 2000s, there was a tendency to transfer local software to cloud programs operating on the SaaS principle – Software as a Service.	• Flexibility • Automatic scalability • Availability of resources on demand • Reduced infrastructure costs	• Heavy dependence on cloud service provider • Limitations in the choice of technologies • Difficulty in debugging and data security aspects	Integration of enterprise information systems with modules deployed in the cloud is always a complex project, which, in addition to technical issues, involves solving security issues. To carry out integration, it is necessary to provide access to the enterprise's internal network, which imposes additional risks. If enterprises deploy solutions in the clouds, integration issues are resolved separately and have their own limitations, and therefore the set of integration tasks of cloud services is difficult to consider in a single portfolio with other integrations.
Layered architecture	Components within the layered architecture model are organized into horizontal layers, each of which performs a specific role in the application. Most layered architectures consist of four standard layers: presentation layer, business logic layer, data access layer, and database abstraction layer.	• Separation of tasks between components • Each layer of the layered architecture pattern has a specific role and responsibility in the application	• Failure to document the system and communicate which layers are open and which are closed leads to tightly coupled and fragile architectures that are difficult to deploy and maintain.	The ability of each module to interact with any other module without a clear structure and hierarchy can lead to a complex and confusing network of connections. In particular, cyclic dependencies may appear, which can lead to endless recursion and mutual locking. This architecture is characterized by high connectivity and complexity of interactions between components. This makes any changes problematic, as it is not completely cleaFor layered architecture, techniques are especially relevant to take into

Architecture	Researches	Advantages	Disadvantages	Applicability for solving end-to-end planning problems
	In the 1980s, design patterns emerged, for example, that implement layer structure in applications. In the 1990s, layered architecture became a popular approach for developing various software systems, including web applications and enterprise systems. In the 2010s, with the advent of new technologies such as web services, REST APIs, microservices, layered architecture became even more flexible and extensible.	• Changes made to one layer of the architecture generally do not affect components in other layers	• Layered architecture tends to create monolithic applications, even if the presentation layer and business logic layer are separated into separate deployable units • Low productivity	account the influence of components on each other and the dependencies between them, as well as to prioritize the software development of new dependencies and business logic taking into account these dependencies.

About the author

Natalya N. Seredenko

Cand. Sci. (Econ.);

Associate Professor, Department of Applied Informatics and Information Security, Plekhanov Russian University of Economics, 36, Stremyanny lane, Moscow 115054, Russia;

E-mail: Seredenko.NN@rea.ru

ORCID: 0009-0009-8113-0758