A COMBINED METHODOLOGY FOR INFORMATION SYSTEMS ANALYSIS AND DESIGN BASED ON ISAC AND NIAM

MICHAEL Z. HANANI AND PERETZ SHOVAL

Department of Industrial Engineering and Management, Ben Gurion University of the Negev,
P.O. Box 653, Beer Sheva 84100, Israel

(Received 4 June 1985; in revised form 7 January 1986)

Abstract—A Combined Methodology for information systems analysis and design is presented. The basis
of the methodology is a hybrid of two existing methodologies, ISAC and NIAM. The business analysis
and information analysis of ISAC are combined with the conceptualization process of NIAM, and thus
the stronger parts of each methodology are combined to form a more inclusive, applicable methodology,
while the weaker parts of the original methodologies are discarded.

Categories and subject descriptors: D.2.2 [Software Engineering]; D.2.1. Requirements/specifications; D.2.2.
Tools and techniques; H.2. [Database Management]; H.2.1. Logical design, data management.

Key words: Methodologies for systems analysis and design, conceptual schema, functional analysis,
information structure diagrams.

1. INTRODUCTION

In Ref. [1] an insight is given to the field of information systems analysis and design, by classifying the
stages on the one hand, and categorizing the approaches into perspectives, on the other hand. Thus,
four stages are identified: strategic information systems planning, business analysis, external information
systems design, and internal information systems design; and three perspectives: data oriented perspective,
process oriented perspective, and behaviour oriented perspective.

The Report states that: "Traditionally, one is tempted to regard an information systems design methodology as having a main perspective, because many design methodologies have been produced with a heavy emphasis on one perspective. Indeed, the CRIS I [2] Review Committee found it useful to categorize the submissions according to the perspective they emphasize".

This traditional categorization emanates from the visible fact, that in existing methodologies there is a correspondence between the perspective that is dominating, and the stage which the methodology is mostly addressing. Commonly, a methodology that is categorized as process oriented would emphasize the business analysis stage, while a data oriented methodology would emphasize the design stage.

ISAC [3] and NIAM [4] are two methodologies that are typical to this situation. ISAC, in spite of its attempt to cover the entire information system development process, has its strength in the business analysis stage, and that is where its major contribution lies. Indeed, ISAC is of process oriented perspective. NIAM provides a complete tool for conceptual database design, and is clearly categorized as of data oriented perspective. A comparative analysis of ISAC and NIAM is included in [5]. As part of that analysis, an integration of ISAC and NIAM is suggested: "ISAC is very strong in the early stages of systems design, "Change Analysis" and "Activity Analysis". However, the data model which is developed in "Information Analysis" is cumbersome, and its development cannot guarantee integrity...NIAM could replace the Information Analysis's phase of ISAC".

In this paper, a methodology is presented that combines ISAC and NIAM in a complementary fashion, and thus addresses itself in a homogeneous way to the entire analysis and design section of the information system life cycle (Business Analysis, External System Design and Internal System Design). The presentation starts with a brief description of ISAC, and an analysis of its strong points and its weak ones. A description and discussion of NIAM follows. The Combined Methodology is then presented, supported by an extended example. Finally, further possible developments are viewed.

2. ISAC

ISAC (Information Systems Work and Analysis of Change) [3] is a methodology for the analysis and design of information systems, that makes extensive use of graphic formalism both as a working tool during the design process, and as a design product presentation device [1].

ISAC philosophy is based on the conviction that in introducing information systems, one changes an
existing environment, rather than creates a new one. ISAC thus starts with "Change Analysis", where the environment is studied, and improving changes are suggested. User's participation in this work is clearly very important, and the graphic tool used, Activity graphs (A-graphs), is both simple enough to ensure communication with the user, yet quite sophisticated to enable development and presentation of complex environments.

An Activity graph (A-graph) is composed of the following elements:

—a rectangle that delimits an environment that is being analysed,
—parallelograms that represent sets (information sets are distinguishable),
—nodes that represent activities, and
da directed arcs, where an arc leads either from an activity to a set, or from a set to an activity.†

A parallelogram may appear above the rectangle, in which case it represents a set entering from the outside into the studied environment. It may appear underneath the rectangle, in which case it represents a set exiting the studied environment, and it may appear inside the rectangle, representing a set produced by one activity, and entering another. Figure 1 is a simple A-graph.

An activity node is generally a microcosm of activities, that can be shown explicitly by enlarging the node into a full fledged A-graph. An example is given in Fig. 2.

In such a descendant graph, naturally, all the sets that were produced by the activity node appear underneath the rectangle, and all the sets that entered that node, appear above it.

In this way, a tree of A-graphs can be created. A node in that tree is an Activity graph, representing a delimited environment. Property tables that accompany the A-graphs add semantic information that cannot be included in the syntactic oriented graph structure.

In Change Analysis, changes in the environment are suggested, to remedy deficiencies in the functioning of the business organization. From the effect of these changes on the structure of an A-graph, it is seen whether they involve handling of information. Only in those cases work proceeds towards Activity Studies. A-graphs continue to be the major tool, and the activity nodes that are affected by the changes are further analyzed, expanded into more detailed A-

†In the ISAC texts, two types of arcs exist, one representing move of information, and the other of material. This distinction is practically unnecessary, as is also stated in Ref [1].
graphs, until the point where Information Precedence Analysis is to commence.

Information Precedence Analysis uses I-graphs. I-Graphs have many features in common with A-graphs, and it has been suggested in [5] that these two families of graphs can be combined. There are, however, several features that distinguish I-graphs from A-graphs. This topic is not being dealt with here.

I-graphs demonstrate clearly the precedence/succedence relationships among the information sets, and make quite straightforward to delimit subsystems as part of a modular system construction. (An example of an I-graph that has been derived from activity node 1 in A-graph M3 of Fig. 2, is presented in Fig. 3.) An I-graph is refined by exploding nodes into descendant I-graphs, in a similar way as is done in A-graphs (see Fig. 4).

Nodes in an I-graph represent processes. The parallelograms are information sets. The structure of the information sets is constructed using C-graphs, and based on them, the processes are designed in "process tables". A C-graph describes the contents and structure of an information set. Its use is based on the assumption that "an information set consists of a number of messages of the same type" ([3], p. 190). Each message is composed of an identification part, and a properties part, which may have repeating sections and sections that are in themselves sub-messages, thus creating composite C-graphs. As an example, the C-graph that corresponds to the information set "Buses Work Plan" in the I-graph of Fig. 4 is given in Fig. 3. In order to construct the C-graph of an information set, one considers, for example, a "visual layout" of a form that represents the set ([3], pp. 171, 172), and based on the data that is included there, creates the appropriate C-graph.

It is in the reliance on C-graphs, that ISAC falters. The weakness of this tool is two fold: first, the very structure of C-graph assumes that only hierarchical schemas exist, and thus no normalization considerations are involved. Second, in the construction of a C-graph one is expected to decide on the "identifying part" of a message and on its "properties part" without being given any directions as to how to do it. Beside being an unsystematic design work, this can result in the same information set being analyzed in two different ways, resulting in two different C-graphs (see also Ref. [5]).

In ISAC, when a visual document is being studied as a basis for the design of the relevant C-graph, that document is regarded not just as a source of knowl-

---

1It is not clear from the ISAC text [3] how precisely this point is recognized. This missing is also addressed in Ref. [5]. The rule that we have followed in practicing ISAC is: a node in an A-graph that is surrounded by information sets only, is a starting point to Information Precedence Analysis (e.g. node 1 in Fig. 2).
edge about the data items that are involved in the prospective system, but also as a guide to what entities are involved, their structure, keys and attributes. This approach hampers a proper independent design of a database that would efficiently serve the whole system that is being developed.

Things get even less pleasant as the development of the system continues with ISAC. As is stated in Ref.
[3] (p. 29), data structures and programs are designed by "putting the 'building bricks' from the information analysis together. We document our results...in a data system design graph (D-graph)." However, no systematic way is suggested for doing it. D-graphs, which are supposed to be concretizations of the I-graphs, where information is concretized by data, and processes by programs, are not designed in any methodological way. Thus, a major gap exists in ISAC between the products of information analysis and the design of the data system. The Combined Methodology addresses this problem, by suggesting the use of a different methodology, NIAM, for the database design stage, and moving in a systematic way from the design product of Information Precedence (ISAC) to the NIAM database design process.

3. NIAM

NIAM (Nijssen Information Analyses Method) is a data oriented methodology for information systems design. It concentrates mainly on the "information analysis" phase, where the conceptual schema of the database is described. It ensures that the conceptual schema reflects in a precise unambiguous manner the semantic information structure of the universe of discourse [6]. Although NIAM has gradually been enhanced to include other aspects of the information systems design, its information analysis (IA) part is considered to be the best developed part of the methodology [5].

NIAM is based on the principle that all communications involving an information system can be considered to consist of deep structure elementary sentences, which can be described completely by a conceptual grammar [7]. This grammar is based on the Binary Relationships approach to conceptual modelling. A discussion of the model and of NIAM can be found in Refs [6, 8, 9]. In the following, a review of some of the building blocks of the model is given, followed by a description of the process of conceptualization.

The basic element in the description of a "thing", or an entity in the universe of discourse is the "object-type" (also referred to as Non-Lexical Object Type, or NOLOT). Facts about entities are represented as binary relationships between the object types, and are referred to as "ideas". Thus, an idea imposes two roles on its two associated object types, a role for each object type. Between the two roles, one of three types of possible dependencies (or constraints) exists: one-to-one (1:1), one-to-many (1:n) and many-to-many (m:n). Occurrences of object types are abstract things, that exist through their properties, which link them to other occurrences. So, an object type can be referenced or identified by some naming convention, called "object-type-name" (also known as Lexical Object Type or LOT).

A graphic formalism is used to represent the binary relationships. Figure 6 is an example of an elementary sentence diagram that presents the idea of bus production. There are two object types: bus and manufacturer; each has an object-type-name, written in parentheses (licence-number and manufacturer-code, respectively). An object type may have more than one name (e.g. manufacturer-code and manufacturer-name). According to our notation, a name that is a unique identifier of its object type is underlined. The arrow above the role "manufactured-by" indicates a 1:n dependency between the two object types in the sentence: a bus is manufactured by only one manufacturer, whereas a manufacturer may manufacture many buses. To indicate 1:1 dependency, two separate arrows above the two roles are drawn, and to indicate m:n dependency, a single long arrow is drawn above the pair of roles.

Diagrams representing a set of sentences can be combined to form an Information Structure Diagram.
(ISD). Figure 7 is an example of a small ISD that represents the following elementary sentences:

- "A bus is of a model" (1:n dependency),
- "a bus has a purchase date" (1:n),
- "buses have maintenance dates" (m:n),
- "a maintenance-affair is of a maintenance-type" (1:n), and
- "a maintenance-affair is performed by an employee" (1:n).

Note in particular, the m:n relation between "bus" and "date", named "maintenance-affair". This relation is also defined as an object type, and is in turn related to other object types. This phenomenon creates a "binary-nested" diagram.

NIAM formalism enables the definition of additional constraints within and among the sentences and a distinction can be made between object types and subtypes. For more details, the reader is referred to the above references.

A complete ISD, that is based on the combination of all the elementary sentence diagrams is a conceptual schema. It models the information structure of the universe of discourse, a model that can later be converted into a database schema, expressed in terms of record types, key fields and attributes. This is done by applying a "grouping" algorithm on the ISD, which groups binary sentences into normalized record types. The algorithm considers the types of dependencies among the object types, as well as other constraints. A detailed description of such an algorithm can be found in Refs [10, 11]. Figure 8 shows the normalized record types that are the outcome of applying the algorithm to the ISD of Fig. 7.

The NIAM process of conceptualization is based on a semantic analysis of "examples". Examples are various documents that either implicitly or explicitly contain information related to the problems that are to be addressed in the information system. The documents are output reports, input forms, screen forms, text, tables, etc. In the first phase of the conceptualization these examples are described in natural language sentences. Then, each sentence is semantically analyzed, and decomposed into a collection of deep structure elementary sentences, that are composed of the basic elements described above. Each elementary sentence is then represented in a
diagram, and the dependencies between the object types of the sentences are defined. In the next step, all the elementary diagrams are combined to form the ISD. Occasionally, in the process, new object types are uncovered (this is typically the case where an m:n relation exists between two object types and this relationship is defined as an object type all by itself, sometimes with other object types related to it).

Once all the constraints among the sentences of the ISD are defined, the conceptual schema is completed. It is precise and unambiguous, and serves as a clear communication medium between the analyst and the user. As already stated, it can be easily converted into a database schema of normalized record types, thus enabling a "smooth" transition to the implementation phase.

Considering some of the shortcomings of NIAM, the methodology is basically a "bottom-up" systems design technique. It commences with an analysis of documents and "examples" taken from the environment being analyzed, and there is an obvious lack of assurance that a thorough job of information requirements analysis has been done. NIAM has been referred to as "conceptualization-by-example". To guarantee that these examples actually cover the information needs of the environment, a systematic functional analysis of the universe of discourse has to be carried out. That analysis could be done the ISAC way.

It is stated in Ref. [5] that NIAM could be improved in the area of "Business Analysis". According to the evaluation presented there, NIAM spends 80% of the total development cycle on the "Data Analysis" and "Physical Design" phases, and only 20% on the earlier phases of "Change Analysis" and "Activity Analysis". Based on these numbers, and more evaluations, it is suggested that ISAC is the best complementing methodology to NIAM. ISAC's strength is in the early phases of "Change Analysis" and "Activity Studies", which are the weak points of NIAM, while NIAM provides a much better tool for the phases of conceptual schema and database design.

4. THE COMBINED METHODOLOGY

In the previous two sections, ISAC and NIAM were reviewed, with an emphasis on discussions of their strengths and weaknesses. Their complementary strong parts suggested the potentiality of combining...
them into a single, extended methodology, that would cover a larger section of the systems development process. In this section, the Combined Methodology is presented.

The Combined Methodology is based on the use of ISAC for the early stages of analysis, and the use of NIAM for the resulting database conceptual design. Work commences with Change Analysis, Activity Studies and Information Precedence Analysis, the ISAC way, and proceeds with NIAM’s Semantic Analysis, Database Conceptual Design and the normalized records design.

Change Analysis assures a thorough Business Analysis, Requirement Analysis and the design of an improved business environment before Information Analysis starts. Activity Studies and Information Precedence Analysis lead to the identification of needed information subsystems, and their corresponding information sets.

Completeness, from the point of view of the business environment, on the one hand, and modularity, from the point of view of information systems design, on the other hand, are achieved.

When Information Precedence Analysis is completed, the design product we have is a set of bottom level I-graphs, each containing several nodes, repre-
sented processes, and their corresponding information sets. Now, for each information set of each
1-graph, an "example document" is considered. This may be either an existing form, report or even a
descriptive example, or a newly designed one. This particular step partially overlaps ISAC (in preparation
for C-graphs design) and NIAM initial step. These "examples" are the starting point for the
Semantic Analysis in the NIAM sense. Each
"example" is described in natural language com-
pound sentences, and then semantically analyzed
with the purpose of deriving the deep structure
elementary sentences.† † Each elementary sentence is
described in an elementary diagram, and then all the
elementary diagrams that belong to a particular
information set are combined to form a Local
ISD (Local in the sense that it corresponds to one
information set.)

In the next step, the Local ISDs that belong to a
particular 1-graph, are combined to form an 1-graph
ISD, or IGISD. This is done for all the bottom level
1-graphs, resulting in a set of IGISDs, one for each
1-graph. Each IGISD may now be viewed as a
conceptual subschema, that corresponds to the delimited
subsystem that is represented by the original
bottom level 1-graph. However, it might be decided
to group together several related 1-graphs, to form a
larger "user view", in which case the corresponding
IGISDs will be grouped together to form the respec-
tive conceptual subschema. Such decisions have
to do with project management, with the design of the
information system development work and the asign-
ment of team subprojects, and are thus outside the
scope of this paper.

The design process proceeds now in two parallel
routes. In the one route, development of the pro-
cedures of the information system is carried out, and
in the other route, database design is done. The
IGISDs along with the original 1-graphs serve as
the major tools in both routes. Regarding procedures
development, 1-graphs provide details of process
nodes, each with its input and output information
sets, while the appropriate IGISD provides details of
the corresponding information structures. Given
these, the designer can check the correspondence
between object types of the input and output sets of
each process node. With the achievement of proper
balance between the object types, actual process
design and program development can proceed. (As
mentioned already, the details of this aspects of the
system development are outside the scope of this
paper.)

As to the database design, work proceeds with the
conversion of the IGISDs into normalized record

† † The procedure of natural language sentences converted
into elementary sentences is not always needed. In many
cases it is possible to directly derive diagrams from the
"examples".

5. SUMMARY

Numerous methodologies for information systems
design have been developed and are available today.
Most of these methodologies address the entire de-
velopment process, but each has its strength in a par-
ticular stage of the development process. They differ
in their approaches in the tools they use in the design
process, and in presenting the design product. Some
are closely related in their basic reasoning and logic,
others represent different and unrelated school of
thought. It has not been the purpose of this study to
develop another, different methodology, but rather to
make a step towards the establishment of an encom-
passing methodology, based on existing achieve-
ments. In combining existing methodologies, we have
extracted the stronger parts of each, made certain
changes, and devised a linkage that made a single,
unified entity, out of two separate ones.

The Combined Methodology, as is presented in
this paper, is a practical tool, can be relatively easily
learned by system analysts, and has the potential of
supporting them in the more efficient development of
better information systems.

REFERENCES

[1] T. W. Olle et al. Information systems design method-
oologies. A draft report. IFIP Working Group 8.1 Task
Information Systems Design Methodologies: A Com-
eration database management systems. In Proceeding
of the IFIP Congress, 1977 (Edited by B. Gidsrist), pp.
CIAI, ISAC and NIAM. In Ref [12].
for the conceptual schema and the information base.


