

RELATIONSHIPS BETWEEN COMMON GRAPHICAL REPRESENTATIONS USED IN SYSTEM ENGINEERING

James E. Long
Vitech Corporation
2070 Chain Bridge Road, Suite 105
Vienna, Virginia 22182-2536

Abstract. Most system engineers today use graphical representations of a system to communicate its functional and data requirements. The most commonly used representations are the Function Flow Block Diagram (FFBD), Data Flow Diagram (DFD), N2 Chart, IDEF0 Diagram, and Behavior Diagram (BD). This paper discusses the characteristics of each and shows how they are related.

When analyzed in the context of specifying functional control and data modeling, it appears that the FFBD and DFD representations are limiting, special cases of the Behavior Diagram. The N2 Chart has the same capability as the DFD, with a more formal format. The IDEF0 is essentially an N2 Chart with some control definition (no constructs) capability. The IDEF0 has the capability to indicate the allocation of functions to system components.

Thus, the Behavior Diagram features comprise a “parent/unified” set of graphical system representations. To achieve the same level of specification completeness, you would have to use an integrated set of the FFBD and one of the data models or augment the FFBD with a graphical representation of the data model, as was done at TRW (then called Function Sequence Diagrams, FSDs).

BACKGROUND

Over the past several years, system engineers have evolved to a few graphical representations to present the functional and data flow characteristics of their system design. The most common of these are the Function Flow Block Diagram (FFBD), Data Flow Diagram (DFD), N2 (N-Squared) Chart, IDEF0 Diagram, and Behavior Diagram (BD). All of these graphical representations allow the engineer to decompose the functional and/or data models hierarchically. The objective of this paper is to analyze the representation capability of these graphic “languages” to see if there is a unifying view available.

TERMINOLOGY

Let us introduce two terms that we use in describing the conditions that allow/cause a function to begin execution. Considering the control and data environment, a function can begin execution if it is both enabled (by control) and triggered (by data). In the case where there is no data trigger specified, a function begins execution upon being enabled. A function is enabled if the function(s) that precede it in the control flow specification have completed execution (e.g., satisfied their completion criteria). A function is triggered when the required stimulus data item becomes available to the function. We are not concerned here with other execution requirements (such as the availability of necessary resources) which could be represented by either control or data structures as necessary.

FUNCTION FLOW BLOCK DIAGRAM

The Function Flow Block Diagram (FFBD) was the first to be favored by system engineers and continues to be widely used today (DSMC 1989, Blanchard and Fabrycky 1990). Figure 1 shows a sample FFBD. An FFBD shows the functions that a system is to perform and the order in which they are to be enabled (and performed). The order of performance is specified from the set of available control constructs shown in Figure 2. The control enablement of the first function is shown by the reference node(s) which precede it, and the reference node(s) at the end of the function logic indicate what functions are enabled next. The FFBD also shows completion criterion for functions as needed for specification (for example, the exits for the multiple-exit function in Figure 1). The FFBD does not contain any information relating to the flow of data between functions, and therefore does not represent any data triggering of functions. The FFBD only presents the control sequencing for the functions.

N2 CHART

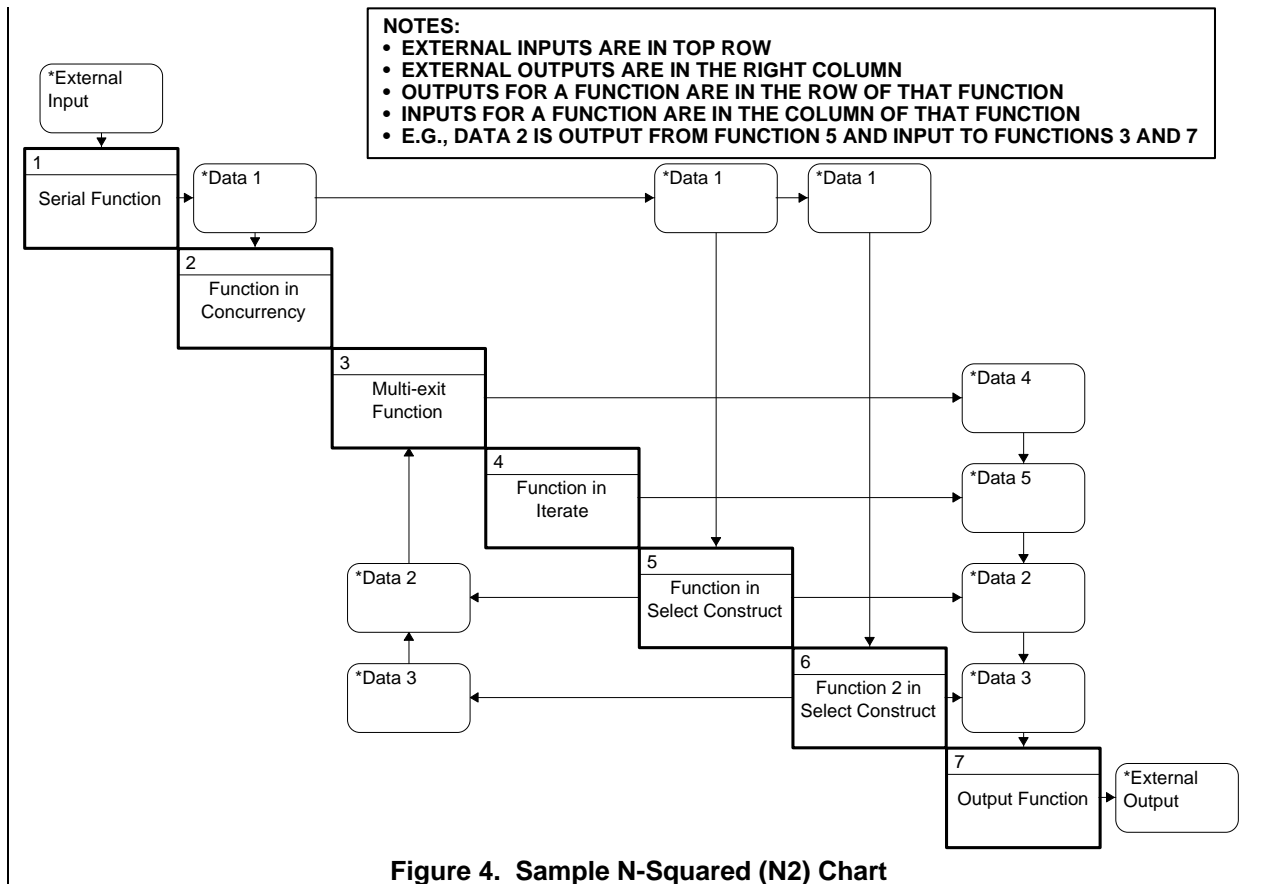
The N-Squared (N2) Chart, shown in Figure 4, was developed to show and specify interfaces between the elements of a system (Long et al. 1968, Lano 1977). Figure 4 is the N2 Chart that corresponds to the FFBD in Figure 1. When used to show the interfaces between the functions in a system, the N2 chart is equivalent to a DFD — it contains all the information and differs only in format. The N2 chart is commonly used as a complement to the FFBD to provide the data flow information as inputs and outputs of the system functions.

The N2 Chart is structured by locating the functions on the diagonal, resulting in an N x N matrix for a set of N functions. For a given function, all outputs are located in the row of that function and all inputs are in the column of the function. If the functions are placed on the diagonal in the nominal order of execution, then data items located above the diagonal represent normal flowdown of data. Data items below the diagonal represent data item feedback. External inputs can optionally be shown in the row

above the first function on the diagonal, and external outputs can be shown in the right-hand column. If desired, data repositories can be represented by placing them on the diagonal with the functions.

IDEF0 Diagram

The IDEF0 Diagram (see Figure 5) appears to be a derivative of the DFD with a format like the N2 Chart (Groveston 1989). The primary content of the IDEF0 Diagram is the specification of data flow between system functions. The IDEF0 diagram does allow the specification of control as an input to a function, but does not have the capability to characterize that control in terms of constructs, as the FFBD and Behavior Diagrams do. The specification of control with the IDEF0 notation is incomplete and, therefore, not executable. The IDEF0 Diagram also represents the mechanism (usually the system component to which the function is allocated) which performs the function. Figure 5 is the IDEF0 Diagram that corresponds to the Behavior Diagram in Figure 6.



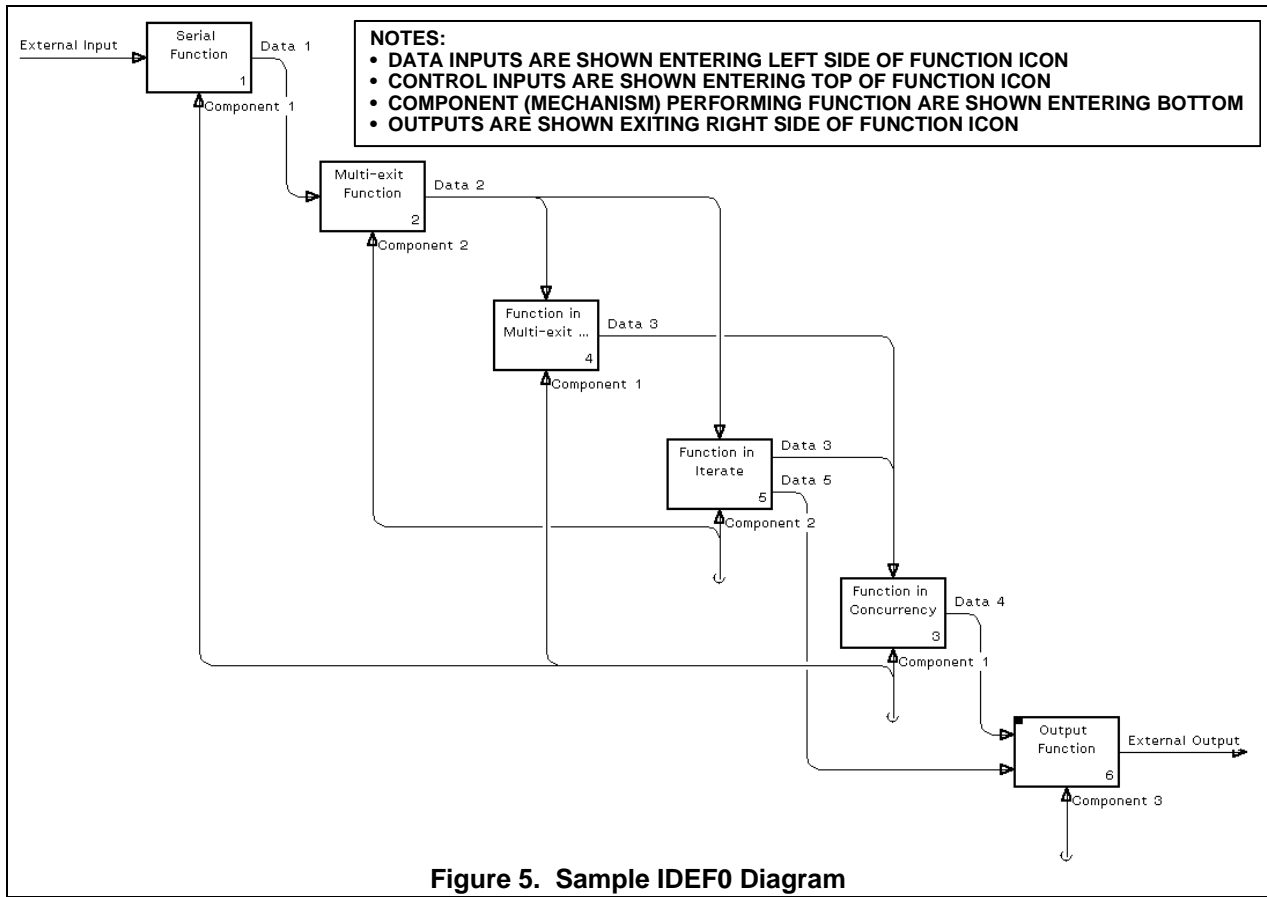


Figure 5. Sample IDEF0 Diagram

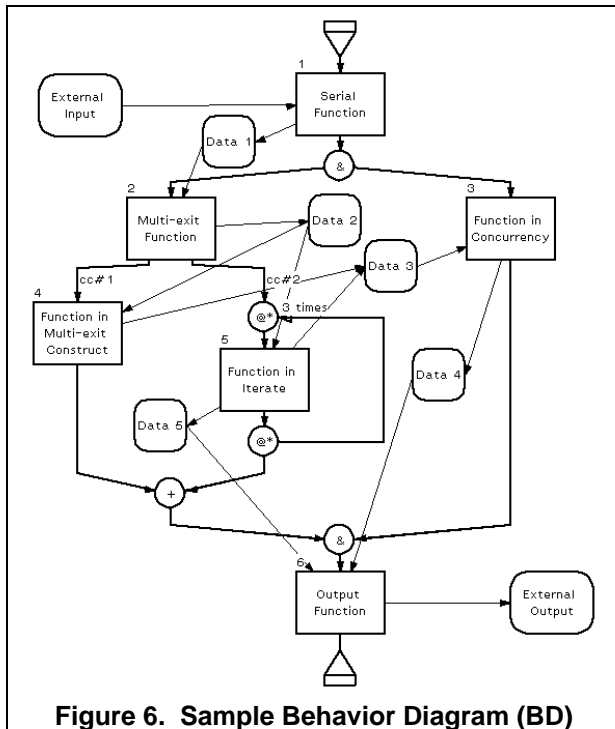


Figure 6. Sample Behavior Diagram (BD)

BEHAVIOR DIAGRAM

The Behavior Diagram (Figure 6) represents both the control flow and the data flow models on the same diagram. Historically, it started as an “Enhanced FFBD”, with the additional representation of data as inputs and outputs to the functions on an FFBD (Long et al. 1968, Oliver 1994). While it is not shown on the graphical construct, the Behavior Diagram (BD) model allows data inputs to a function to be characterized as either triggering (a control capability) or data update (not a control implementation).

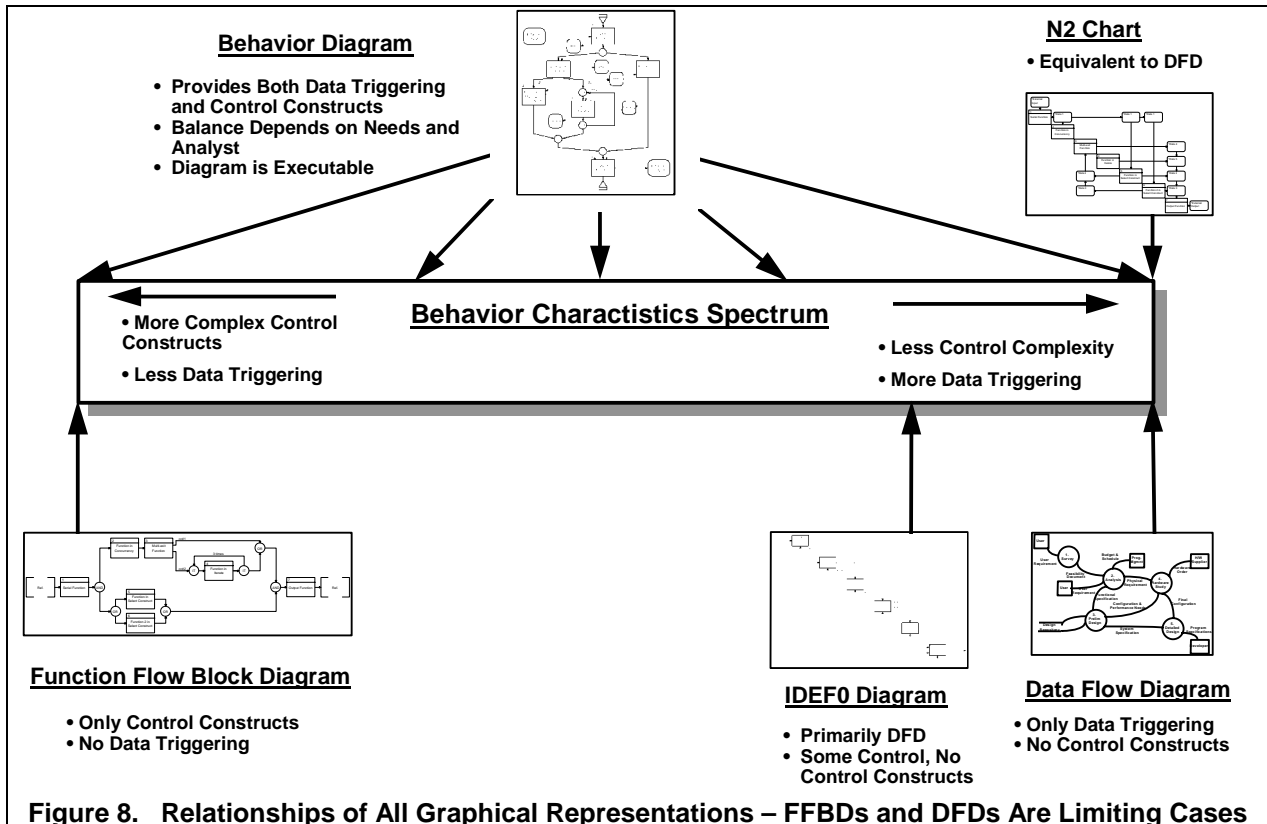
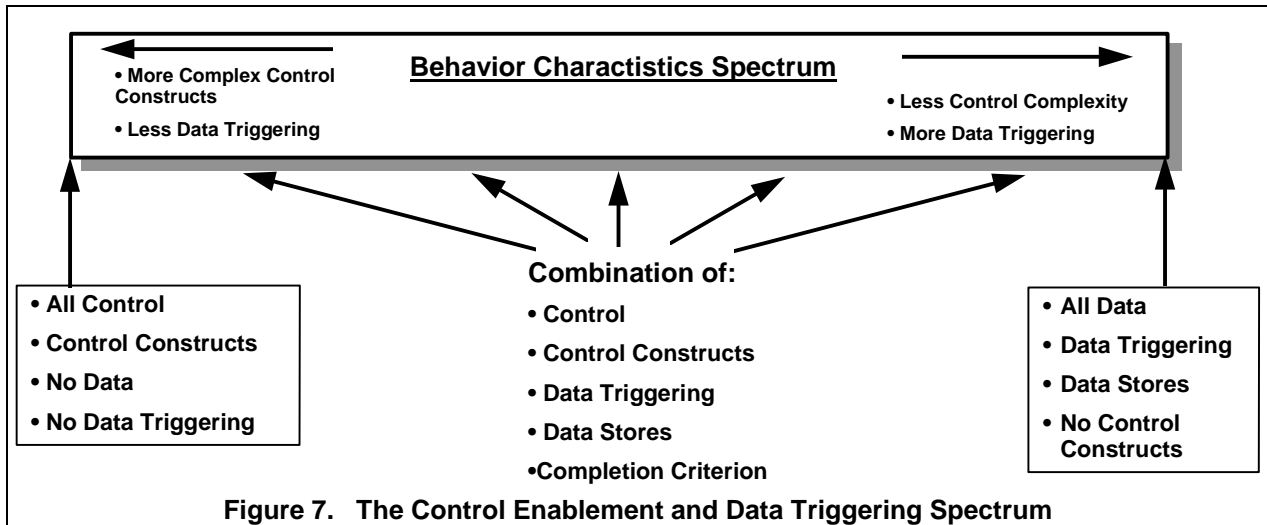
The Behavior Diagram specification of a system is complete enough that it is executable as a discrete event model, providing the capability for dynamic, as well as static, validation. A fundamental rule in the interpretation of a BD specification is that a function must be enabled (by completion of the function(s) preceding it in the control construct) and triggered (if any data input to it is identified as a trigger) before it can execute. This allows the engineer maximum freedom to use either control constructs or data triggers (or a combination of both) to specify execution

conditions for individual system functions.

RELATIONSHIPS BETWEEN THE DIAGRAMS

From the descriptions of the graphic representations, it is seen that the FFBD characterizes only the functional control model and the DFD characterizes only the functional data model for a system. The Behavior Diagram captures both these

limiting models and the continuum between them. Figure 7 is drawn to show the characteristics of the Behavior Spectrum from one limiting case to the other. Figure 8 shows where all these graphical representations fit on the Behavior Spectrum and, therefore, places them in perspective relative to each other in the sense of specifying control and triggering for functions.



CONCLUSION

The graphical representations that system engineers commonly use to describe and specify the functionality and data requirements of a system can be shown to be very closely related when analyzed in the context of data and functional control capabilities. In particular, the FFBD and DFD are limiting cases of the Behavior Diagram representation. The N2 Chart is equivalent to the DFD, so it is, likewise, a limiting case of the BD on the data modeling end of the spectrum. The IDEF0 is essentially a DFD, except that some control capability (no control constructs) is added. The IDEF0 also allows the explicit representation of functional allocation (i.e., what system component performs each function).

Thus, the Behavior Diagram features essentially comprise a “parent” or unifying set of graphical system representations. To achieve the same level of specification completeness, you would have to use an integrated set of the FFBD and one of the data models or augment the FFBD with a graphical representation of the data model, as was done in the 1960s at TRW (where they were called Function Sequence Diagrams, FSDs).

As stated earlier, a fundamental rule in the interpretation of a BD specification is that a function must be enabled (by completion of the function(s) preceding it in the control construct) and triggered (if any data input to it is identified as a trigger) before it can execute. This allows the engineer maximum freedom to use either control constructs or data triggers (or a combination of both) to specify execution conditions for individual system functions. It also means that if a system engineer draws BDs with minimal use of data triggering, the engineer is essentially generating an FFBD. If the BD makes minimal use of the control constructs (the structure is mostly parallel/concurrent), the engineer is generating a DFD representation of the system.

REFERENCES

- Blanchard, B. and Fabrycky, W., *System Engineering and Analysis*, Prentice Hall, Englewood Cliffs, N.J., 1990.
- Defense Systems Management College, *Systems Engineering Management Guide*, U.S. Government Printing Office, Washington D.C., 1989.
- DeMarco, T., *Structured Analysis and System Specification*, Prentice Hall, Englewood Cliffs, N.J., 1979.
- Groveston, P. K., “Functional Diagramming Methods”, Unpublished MITRE Briefing, March 1989.
- Lano, R., *The N² Chart*, TRW Software Series, Redondo Beach, CA, 1977.
- Long, Dinsmore, Spadaro, Alford, et al., “The Engagement Logic and Control Methodology as Derived, Defined, and Applied at TRW”, unpublished notes, 1968 - 1972.
- Oliver, D., “Systems Engineering and Object Technology”, *Proceedings of the Fourth Annual International Symposium of NCOSE*, August 1994.

ABOUT THE AUTHOR

James E. Long is the President of Vitech Corporation – the developer of the system engineering support tool CORE[®]. He has been a performing system engineer and innovator since creating the first behavior diagrams (then called Function Sequence Diagrams) at TRW in late 1967. He played a key technical and management role in the maturing and application of that technology at TRW. Both CORE[®] and RDD-100[®] are based on the TRW developments.