2001

B-nodes: a new scalable high level abstraction model

Stanislaw P. Maj
Edith Cowan University

Craig William Caulfield
Edith Cowan University

10.1109/ICSMC.2001.972913

B-Nodes: A New Scalable High Level Abstraction Model

S P Maj, C Caulfield

Department of Computer Science,
Edith Cowan University, Western Australia

Introduction

In order for any Information Technology system (e.g. E-Commerce, Digital Library) to provide a prompt and possibly worldwide service they must be based on a complex infrastructure of hardware and software. Furthermore this system must provide a Quality of Service to meet the Service Level Agreement. In order to control the complexity associated with designing an IT system standard analysis and design methods are used. A method consists of phases or stages that in themselves may consist of sub-phases. It should be noted that,

It is a reasonable estimate that hundreds of more or less similar methodologies have been published. In practice, probably tens of thousands of more or less different approaches are being used. Most organizations have developed their own methodology and prescribed it in the organization's (data processing) handbook' [22]

However it is possible to classify methods into three different categories: formal, structured and soft. Formal methods consist of a mathematically based formal language and deductive apparatus. This type of method is designed to eliminate software errors by using rigorous engineering practices. Despite some notable successes formal methods are not widely used due to their complexity and the requirement for analysts to have a very strong mathematical background. Structured methods use a variety of tools and techniques to model data and processes. This type of method emphasizes the need for structure and hence repeatability but without the need for mathematical rigor. According to White,

'...the acid test of a methodology is its repeatability. Given the same access to people and facts about the organization, will different designers come up with essentially the same solution? If the answer is no, then the business is placing itself in the hands of its system builders and gambling on their flair and judgment to come up with a satisfactory approach. This is a risk that many organizations are unwilling to take as they progress towards their goal of becoming 'computerized companies'. Instead they are looking for an approach, which can be treated as an engineering discipline and which will provide them with the means of verifying the completeness and correctness of any analysis and of the assumptions made at each stage in the development process. It is this type of approach that will give us a 'repeatable' methodology'. [25]

The issue of repeatability is however problematic. Checkland argues that it is impossible to prove if success or failure of an information system can be attributed to the methodology used [5]. Furthermore structured methods tend to stress only the technical aspects which may arguably lead to a less than ideal solution as these methods underestimate the importance and difficulties associated with the human element. According to systems theory a method should consider the organization as a whole and also be aware of externalities, which led to the development of 'soft' methods. According to systems theory a method should consider the organization as a whole and also be aware of externalities, which led to the development of 'soft' methods. Such 'soft' methods lack rigor and are sometimes used in conjunction with a structured method. In the final analysis there is no single 'best' method – each has both strengths and weaknesses. Regardless of the method employed the information system must provide useful, correct and timely information to an organization. The chosen method must therefore, at some point, provide tools and techniques to convert the requirements of the information system to a hardware implementation. A wide range of methods was analyzed for their ability to map the desired information design onto hardware. The methods analyzed were: ad hoc [13], waterfall [23], participative [20], soft systems [5], prototyping [21], incremental [10], spiral [4], reuse [18], formal [1], rapid application development [17], object oriented [7] and software capability [11]. None of the methods provided a simple technique that will model the digital infrastructure (hardware and software) to determine if it will perform to an acceptable standard required by the analysis and design specifications.

0-7803-7087-2/01/$10.00 © 2001 IEEE

The Structure Systems Analysis and Design Method (SSADM) was then evaluated in-depth.

Structured Systems Analysis and Design Method (SSADM)

SSADM is mandatory for UK central government software development projects. This method is sponsored by the Central Computer and Telecommunications Agency (CCTA) and the National Computing Center (NCC) thereby further ensuring its importance within the software industry within the UK. SSADM is a framework employing a wide range of techniques (Data Flow Diagrams, Entity Models, Entity Life Histories, Normalization, Process Outlines and Physical Design Control). SSADM is divided into six stages (Analysis, Specification of Requirements, Selection of System Option, Logical Data Design, Logical Process Design and Physical Design). The Physical Design translates the logical data design into the database specification and the logical process designs into code specifications. SSADM is recognized to be a highly structured method with numerous cross checks to ensure quality control. The final stage of SSADM (Physical Design stage) includes a step called, 'Create Performance Predications and Tune Design (Step 630)'. The objectives of this step are to produce a tuned physical data and process design that meets the Performance Objectives previously agreed with users in a previous stage. SSADM provides tools that allow the estimation of storage requirements. From the Composite Logical Data Design and Logical Design Volumes, detailed information about the data volumes may be extracted. It is possible to obtain detailed information about: data space for each data group, volumes of each data group, volumes of relationships, variance of volumes over time etc. According to Ashworth,

'*The basic approach to timing is to calculate the disk access time and CPU utilization time for each transaction. The actual elapsed time taken by a transaction (or response time for an on-line transaction) will probably be several times larger than that calculated.*'

And furthermore,

'*The prediction of overall system performance is a difficult area and there are several simulation programs and experts system programs available which attempt to predict and improve system performance either generally or for specific hardware and software.*' [2]

Other than this there are no simple tools or techniques that can be used for the evaluation and selection of hardware. Furthermore, SSADM employ a range of different, heterogeneous performance metrics that include: MPS, CPU time, disk access time, number of instructions per database call etc. Such benchmark metrics are in themselves problematic. Arguably a new technique is therefore needed based on the basic principles of modeling theory.

Models and Modeling

Models are used not only as a means of communication and controlling detail but may also forming the basis of a conceptual understanding of a system. According to Cooling there are two main types of diagram: high level and low level. High-level diagrams are task oriented and show the overall system structure with its major sub-units. Such diagrams describe the overall function of the design and interactions between both the sub-systems and the environment. The main emphasis is 'what does the system do' and the resultant design is therefore task oriented. According to Cooling, '*Good high-level diagrams are simple and clear, bringing out the essential major features of a system*' [8]. By contrast, low-level diagrams are solution oriented and must be able to handle considerable detail. The main emphasis is 'how does the system work'. However, all models should have the following characteristics: diagrammatic, self-documenting, easy to use, control detail and allow hierarchical top down decomposition. In effect models provide abstraction to control complexity and change. There have been, and may continue to be, many technical developments that impact on computer and network technology education. Clements makes the point that, '*For example, the generation of students studying electronics in the 1950's leaned about the behavior of electrons in magnetic fields. The next generation studied transistor circuits, and the one after that studied integrated circuits. The traditional logic course changes rapidly.*' [6]. Digital techniques and modeling provide an abstraction that is independent of the underlying details of transistor theory. Such combinational or sequential circuits can be described without the complexity of their implementation in different switching technologies e.g. Transistor-Transistor Logic (TTL), Complementary Metal Oxide Semiconductors (CMOS) etc. Similarly details of semiconductor switching may be modeled using abstractions independent of the underlying details of quantum mechanics. Computer and network technology can therefore be described using a progressive range of

models based on different levels of detail e.g. semiconductors, transistors, digital circuits, registers, device controllers, clients, servers etc. Each model is valid in the context in which it is used. All such models are designed to progressively hid and hence control detail and yet still provide sufficient information to be useful for communication, design and documentation. This is in keeping with the ACM/IEEE Computing Curricula 1991 in which abstraction is a recurring concept fundamental to computer science [24]. A PC can be described as a collection of sub-modules (microprocessor, electronic memory, hard disc drive etc) arranged in a memory hierarchy [12]. The performance of a PC depends therefore on the performance of individual sub-modules. However, the sub-modules employ different technologies (electronic, electromechanical etc) and there exists therefore a wide range of performance metrics that include: MHz, nanoseconds, rpm, seek time, latency etc. The direct comparison and evaluation of these heterogeneous modules is therefore problematic. Furthermore, there appears to be no simple model that can be used to describe these sub-modules. A new higher-level abstract model of computer and network technology and associated metric is needed not only as a commercial tool but also as a new pedagogical tool.

B-Nodes

Individual modules in a PC (microprocessor, hard disc drive etc), and the PC itself, may be modelled using B-Nodes [15]. Each B-Node can be treated as a data source/sink capable of, to various degrees, data storage, processing and transmission. The performance of each B-Node may be calculated, to a first approximation, by Bandwidth = Clock Speed x Data Path Width (B = C x D) with units in either MBytes/s or Frames/s. A frame is defined as 1024x1024 pixels with a color depth of 3 bytes per pixel i.e. 3MBytes. This simple, high-level, task oriented model may provide a suitable conceptual map and hence the framework for an introduction to computer technology. Even though technical detail is lost, this model is conceptually simple, controls detail by abstraction and may allow students to easily make viable constructs of knowledge based on their own experience. The units Frames/s may be more meaningful to a typical user because it relates directly to their perception of performance. To a first approximation, smooth animation requires approximately 30 Frames/s (90MBytes/s). According to Barney, '*A measurement is the process of empirical objective assignment of numbers to properties of objects or events in the real world in a way such as to describe them*' [3]. History has many examples of measures in the search for useful standards. Early Egyptians defined one finger-width as a zebo and established an associated simple, reproducible and denary scale of standard measurements. It is significant that human dimensions were used as the basis of one of the first standards. If B-Nodes are not used hardware selection is based on a wide range of units (Microprocessor -MHz, Electronic memory - nanosecond, Hard Disc Drive - rpm, CDROM – speed etc). Evaluation of these heterogeneous modules, each using different units of measurement, is therefore difficult. B-Nodes can be used to model heterogeneous sub-modules within a PC (microprocessor, hard disc drive, bus structures, network etc) using simple, meaningful, derived units with a denary scale. We have therefore a common unit of measurement, relevant to common human perception, with decimal based units, that can be applied to different nodes and identify performance bottlenecks. The use of simple, fundamental units allows other units such as frame transfer time to be easily calculated. This allows the performance of heterogeneous units to be directly compared (Table 1) using the same units.

Table 1: Bandwidth

Device	Clock Speed (MHz)	Data Width (Bytes)	Bandwidth (MBytes/s)
Processor	400	8	3200
DRAM	16 (60ns)	8	128
Hard Disc	60rps	90Kb	5.4
CROM	(30 speed)		4.6
ISA Bus	8	2	16
Ethernet	100	1/8	12.5

The characteristics of the Frame may be changed to more directly suit different applications. By example medical images are often stored digitally. A single ultrasound image represents approximately 0.26MBytes of data [9]. The performance of each B-Node may be calculated using this metric. The use of B-Nodes has been confirmed experimentally [14]. The B-Node model has been successfully applied to a wide range of PC architectures allowing a direct comparison not only between different B-Nodes within a given PC but also comparisons between different PC's. Using B-Nodes it was possible to analyze PC's with different Intel microprocessors (8088/6, 286, 386, 486 etc.) and various associated bus structures (Micro Channel Architecture, Extended Industry Standard Architecture, Video Electronic Standards (VESA) Local Bus).

Sub-optimal Operation

B-Nodes typically operate sub-optimally due to their operational limitations and also the interaction

between other slower nodes. For example, a microprocessor may need two or more clock cycles to execute an instruction. Similarly a data bus may need multiple clock cycles to transfer a single data word. The simple bandwidth equation can be modified to take this into account i.e. Bandwidth = Clock x Data Path Width x Efficiency (B = C x D x E). The early Intel 8088/86 required a memory cycle time of 4 clocks cycles (Efficiency = ¼) however, for the Intel 80x86 series, including the Pentium, the memory cycle time consists of only 2 clocks (Efficiency = ½) for external DRAM. Efficiencies can be calculated for each device and the performance calculated accordingly (Table 2) [16]. However, other factors not considered include the effects of compression, operating system overheads etc. The effect of these is currently being examined.

Table 2: Bandwidth with efficiency

Device	Clock Speed (MHz)	Data Width (Bytes)	E	Bandwidth (MBytes/s)
Processor	400	8	0.5	1600
DRAM	16 (60ns)	8	0.5	64
Hard Disc	60rps	90Kb	0.5	2.7
CROM	(30 speed)		0.5	2.3
ISA Bus	8	2	0.25	4
Ethernet	100	1/8	0.9	11.25

Web Server Modeling

If a web server is modeled as a B-Node then the performance metric is bandwidth with units of Mbytes/s. The sub-modules of a server (microprocessor, hard disc, electronic memory etc) and also be modeled as B-Nodes, again using the same performance metric. The use of fundamental units (Mbytes/s) allow other units to be derived and used e.g. transactions per second (T/s). Assuming the messages in a client/server interaction are 10kbytes each, the performance of each B-Node can be evaluated using the units of transactions/s (Table 3).

Table 3: Bandwidth (Transactions/s)

Device	Bandwidth (MBytes/s)	Bandwidth, (T/s)	Load, (T/s)	U
Processor	1600	160k	250	<1%
DRAM	64	6.4k	250	4%
Hard Disc	2.7	270	250	93%
CROM	2.3	230	250	>100%
ISA Bus	4	400	250	63%
Ethernet	11.25	1.1k	250	23%

If the demand on this server is 250 Transactions/s it is a simple matter to determine both performance bottlenecks and also the expected performance of the equipment upgrades. From table 3 it is possible to determine that for this web server, the hard disc drive, CDROM and ISA bus are inadequate – the utilization (U) exceeds 50%. The metric of transactions/s can easily be converted to the fundamental unit of Mbytes/s, which can then be used to determine the required performance specification of alternative bus structures, CDROM devices and hard discs. A PCI (32 bit) bus structure is capable of 44Mbytes/s. A 40-speed CDROM device has a bandwidth of approximately 6Mbytes/s. Similarly replacing the single hard disc drive by one with a higher performance specification (rpm and higher track capacity) results is a new server capable of meeting the required workload (Table 4).

Table 4: Upgraded server

Device	Clock Speed (MHz)	Data Width (Bytes)	B (T/s)	Load (T/s)	U
Processor	400	8	160k	250	<1%
DRAM	16 (60ns)	8	6.4k	250	4%
Hard Disc	100rps	250K	1.25k	250	20%
CROM	(40 speed)		0.6k	250	42%
PCI Bus	33	4	6.6k	250	4%
Ethernet	100	1/8	1.1k	250	23%

Capacity Planning

Capacity planning is the process of predicting future workloads and determining the most cost-effective way of postponing system overload and saturation. Assuming that the web traffic is anticipated to rise to 550 transactions/s – the current single server solution will be inadequate. To accommodate much higher web traffic a typical e-business configuration may consist of a front-end Web server, a Secure Web server, a Payments server, an Application server and a Database server. Assuming each server is a separate device connected by a 100Mbps Ethernet link it is possible to model this configuration using B-Nodes. Each server represents a B-Node. The communication link may be represented as a directed arc (arrow) annotated by its bandwidth performance (units MBytes/s or Transactions/s). Using Customer Behavior Model Graphs (CBMG's) it is possible to evaluate the relative frequency that each dedicated server is used [19]. Assuming probability based on relative frequency our performance equation is now Bandwidth = Clock Speed x Data Path Width x Efficiency x Frequency (B = C x D x E x F). For each server the results can be tabulated (table 5). The load data obtained from table 5 can then be used to evaluate the performance of the individual components in each server. In the case of the Web server the actual load is 5.625Mbyte/s (0.55kTransactions/s) from which the utilization of each module can be evaluated (table 6)

Table 5: E-Commerce servers

Server	Network bandwidth (T/s)	F	Actual server load (Mbytes/s)	Actual Server load (T/s)
Web	1.1k	0.5	5.625	0.55k
Secure	1.1k	0.05	0.5625	0.055k
Payment	1.1k	0.05	0.5625	0.055k
Database	1.1k	0.2	2.25	0.22k
Apps	1.1k	0.1	1.125	0.11k

The CMBG's clearly indicate that the majority of the traffic is to the Web server. Assuming that it is necessary to plan for an expected load of 1,000 transactions/s. From table 6 it is evident that the web server would not perform satisfactorily.

Table 6: Evaluation of E-Commerce Web server

Web Server	Bandwidth (MBytes/s)	B (T/s)	Load (T/s)	U
Processor	1600	160k	550	<1%
DRAM	64	6.4k	550	9%
Hard Disc	12.5	1.25k	550	44%
CROM	6	0.6k	550	92%
PCI Bus	66	6.6k	550	8%
Ethernet	11.25	1.1k	550	50%

Traffic characterization may be used. Assuming that such an analysis indicates that 60% of the traffic is for static JPEG images and 40% for dynamic HTML pages. A possible solution may be to use a caching proxy to serve the static web pages. Assuming that the traffic analysis further finds that this brings down the average message size to the server from 10Kbytes to 5kBytes – the web server can easily be modeled using B-Nodes accordingly. For the web server 40% of 1,000 transactions/s is 400 transactions/s, furthermore each transaction is on average 5kBytes. This results in a different, and much lower, utilization for the Web server (Table 7).

Table 7: Evaluation of E-Commerce Web server

Web Server	B (T/s)	Load (T/s)	U
Processor	320k	400	<1%
DRAM	12.8k	400	3%
Hard Disc	2.5k	400	16%
CROM	1.2k	400	33%
PCI Bus	13.2k	400	3%
Ethernet	2.2k	400	18%

It is simple to modify the message size to accommodate the type of transaction. Certainly further work is needed to experimentally verify the errors and possibly absolute performance associated with this approach. However to a first approximation it is possible to determine the relative performance of different architectures.

Tools and Techniques – a new tool in the tool kit. Even though there are a wide range of methods there are a relatively few tools and techniques. Many techniques are common to more than one method. However this does not necessarily mean they are directly interchangeable as they could address different parts of the development process and have different objectives. There are two broad categories of technique – data objects and processes. Entity modeling and normalization are used for data analysis and design. Data flow diagrams, entity life cycles, decision trees/tables etc are used for process analysis and design. Many of these methods have the common characteristics of being self documenting and diagrammatic. Furthermore they allow top-down decomposition hence allowing detail to be controlled by means of abstraction. However an analysis of a wide range of different methods and associated tools and techniques failed to find a simple tool or technique for modeling hardware. Work to date indicates that B-Nodes may be used to model hardware and have the following characteristics:

- Easy to use.
- Self-documenting.
- Diagrammatic.
- Top-down decomposition (allows the detail complex systems to be controlled by means of abstraction).
- Fundamental units (Mbytes/s) allows comparison of heterogeneous devices (e.g. hard disc drive, microprocessor, web server etc).
- Fundamental units allow other, more meaningful units to be derived (e.g. Transactions/s).
- Derived units (e.g. Transactions/s) can easily be converted to fundamental units (Mbytes/s).
- Independent of architectural detail hence B-Nodes are valid not only for old and current technologies but may also be valid for future digital architectures.
- Recursive decomposition .

Scalable i.e. can be used to describe and define a micro system (e.g. hard disc drive), mini system (e.g. PC or LAN) or even a macro system (e.g. global e-commerce system).

Conclusions

This paper proposes a new modeling technique called B-Nodes. B-Nodes represent a new, high-level abstraction that allows technical detail to be controlled using top-down recursive decomposition. This abstraction is independent of architectural detail and can therefore accommodate rapid changes in technology. The use of recursive decomposition allows B-Nodes to be used not only for entire E-Commerce systems but also sub-modules within this system. The use of fundamental units allows the performance of heterogeneous technologies to be compared and other units to be derived. Results to date indicate no comparable model exists. Should further work validate this technique the authors recommend its use as a standard technique in information systems analysis and design.

References

[1] D. Andrews and D. Ince, *Practical Formal Methods with VDM*, McGraw Hill, New York, 1991.

[2] C. Ashworth and M. Goodland, *SSADM: A Practical Approach*, McGraw-Hill, London, 1990.

[3] G. C. Barney, *Intelligent Instrumentation - Microprocessor Applications in Measurement and Control*, Prentice/Hall International (UK), Exeter, 1985.

[4] B. W. Boehm, *A software development environment for improving productivity*, Computer, 17 (1984), pp. 30-42.

[5] P. B. Checkland, *Systems Thinking, Systems Practice*, John Wiley, Chichester, 1981.

[6] A. Clements, *Computer Architecture Education,, IEEE Micro*, 2000, pp. 10-22.

[7] P. Coad and E. Yourdon, *Object-oriented Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1991.

[8] J. E. Cooling, *Software Design for Real-Time Systems*, Chapman and Hall, Padstow, Cornwall, 1991.

[9] S. Dwyer, *Teleradiology Using Switched Dialup Networks*, IEEE Journal on Selected Areas in Communication (1992).

[10] T. Gibb, *Principles of Software Engineering Management*, Wesley, Reading, MA, 1988.

[11] W. S. Humphrey, *Managing the Software process*, Addison-Wesley, Reading, MA, 1990.

[12] K. Hwang and F. A. Briggs, *Computer Architecture and Parallel Processing*, McGraw-Hill, New York, 1987.

[13] G. W. Jones, *Software Engineering*, Wiley, New York, 1990.

[14] S. P. Maj and D. Veal, *Architectural Abstraction as an aid to Computer Technology Education,, American Society for Engineering Education (ASEE)*, St Louis, Missouri, 2000.

[15] S. P. Maj and D. Veal, *Computer Technology Curriculum - A New Paradigm for a New Century*, Journal of Research and Practice in Information Technology, 32 (2000), pp. 200-214.

[16] S. P. Maj, D. Veal and P. Charlesworth, *Is Computer Technology Taught Upside Down?*, in T. J, ed., *5th Annual SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education*, ACM, Helskinki, Finland, 2000, pp. 140-143.

[17] J. Martin, *Rapid Application Development*, Macmillan, New York, 1991.

[18] Y. E. Matsumoto and Y. E. Ohno, *Japanese Perspectives in Software Engineering Practice*, Addison-Wesley, Reading, MA, 1989.

[19] D. A. Menasce, V. A. F. Almeida, R. C. Fonseca and M. A. Mendes, *A Methodology for Workload Characterization for E-Commerce Servers,, 1999 ACM Conference in Electronic Commerce*, ACM, Denver, CO, 1999, pp. 119-128.

[20] E. Mumford and M. Wier, *Computer Systems in Work Design - the ETHICS Method*, Associated Business Press, London, 1979.

[21] J. D. Nauumann and A. M. Jenkins, *Prototyping: the new paradigm for systems development,, MIS Quarterly*, 1982.

[22] T. W. Olle, H. G. Sol and A. A. E. Verrijin-Stuart, *Information Systems Design Methodologies: Improving the Practice*, North Holland, Amsterdam, 1986.

[23] W. W. Royce, *Managing the development of large software systems: concepts and techniques,, WESCON*, 1970.

[24] A. B. Tucker, B. H. Barnes, R. M. Aiken, K. Barker, K. B. Bruce, J. T. Cain, S. E. Conry, G. L. Engel, R. G. Epstein, D. K. Lidtke and M. C. Mulder, *A Summary of the ACM/IEEE-CS Joint Curriculum Task Force Report, Computing Curricula 1991*, Communications of the ACM, 34 (1991).

[25] P. White, *Towards a Repeatable Methodology*, Perspective, 1 (1982).