The Graphical Representation of the Digital Astronaut Physiology Backbone

Demarcus Briers[1]

Nomenclature

DA	=	Digital Astronaut
DAP	=	Digital Astronaut Project
DH	=	Digital Human
HumMod	=	Human Model
NASA	=	National Aeronautics and Space Administration
PEMDAS	=	(order of operations) Parenthesis, exponents, multiplication and division, addition and subtraction
UMMC	=	University of Mississippi Medical Center
XML	=	Extensible Markup Language

I. Abstract

This report summarizes my internship project with the NASA Digital Astronaut Project to analyze the Digital Astronaut (DA) physiology backbone model. The Digital Astronaut Project (DAP) applies integrated physiology models to support space biomedical operations, and to assist NASA researchers in closing knowledge gaps related to human physiologic responses to space flight. The DA physiology backbone is a set of integrated physiological equations and functions that model the interacting systems of the human body. The current release of the model is HumMod (Human Model) version 1.5 and was developed over forty years at the University of Mississippi Medical Center (UMMC). The physiology equations and functions are scripted in an XML schema specifically designed for physiology modeling by Dr. Thomas G. Coleman at UMMC. Currently it is difficult to examine the physiology backbone without being knowledgeable of the XML schema. While investigating and documenting the tags and algorithms used in the XML schema, I proposed a standard methodology for a graphical representation. This standard methodology may be used to transcribe graphical representations from the DA physiology backbone. In turn, the graphical representations can allow examination of the physiological functions and equations without the need to be familiar with the computer programming languages or markup languages used by DA modeling software.

II. Introduction

From over forty years of research, an integrative physiology model known as HumMod (Human Model) has been developed to model the interacting systems of the human body. HumMod is composed of a user interface, and a physiology backbone. The physiology backbone is composed of thousands of variables, functions, and algorithms written in an XML schema developed by Dr. Thomas G. Coleman at the University of Mississippi Medical Center (Coleman, Hester & Summers, 2008). The physiology backbone is being increasingly expanded on by researchers at NASA and University of Mississippi Medical Center (UMMC). With the expanding complexity of HumMod and its physiology backbone it has become difficult for researchers to completely grasp the interacting systems of physiology backbone. To add to the complexity of the physiology backbone, there is no visual representation of how the systems are interconnected through the XML schema. This concern has been expressed by researchers at NASA and UMMC. If HumMod is to be a truly effective backbone, researchers must be able to foresee how it can connect to and supplement the model in a broad spectrum. In order to address these problems, a method of creating standard visual representations of the model from the XML schema needs to be developed.

Through the examination of NASA documents concerning computational biomedical research, it is evident that there is a lack of standardized graphical representation of physiologic system models as a whole (White, 1973; 1974; Fitzjerrell, Grounds, & Leonard, 1975; Kay, 1973). By examining these documents, methods used for the whole body algorithm, Guyton’s model, and other applicable references we propose to develop a standard methodology for creating graphical representations of the Digital Astronaut (DA) physiology backbone model (White, 1973; Van de

[1] Demarcus Briers is majoring in biology at Prairie View A&M University, Department of Biology

Vegte, 1994). In addition to examining previously used methods, the entire physiology backbone, consisting of over five thousand variables, was analyzed for mathematical functions and algorithms that are unique to the DAP.

The Digital Astronaut (DA) physiology simulation backbone is currently the HumMod version 1.5. The XML schema allows flexibility while maintaining organization inside of the model itself. The disadvantage of using the XML schema is the need to learn an extensive markup language in order to effectively analyze the DA physiology backbone.

The proposed graphical representation concerns transcribing from the HumMod code to a standard set of guidelines and symbols. This standard will allow transcribing a control flow diagram from the DA physiology backbone by hand or by means of automated program. With that in mind, the creation of the standardized graphical representation must be flexible enough to clearly portray variables whether in the midst of a few variables or thousands of variables. It may not be feasible to view thousands of variables at once, but the graphical representation must still consider this remote possibility.

While creating a standard for the graphical representation it was also necessary to document the functions of the tags in the XML schema. The XML schema will not be discussed in this summary report.

III. Organization of DA Physiology Backbone

When HumMod is launched and simulations are executed, a series of XML files containing equations and functions are parsed in order for the modeler to simulate the physiology of the interacting systems in the human body. The XML files are organized into folders with names relevant to the different physiologic systems in the human body. Although the XML code inside each file is separate, a file can relate to other files when a variable in its code defines an external variable or is defined by an external variable. An external variable can be differentiated from regular variables by a “dot” notation. For example, a reference to “Kidney.02Flow” in the XML code refers to the “02Flow” variable located in the “Kidney” file.

IV. Proposed Graphical Representation of the DA Physiology Backbone

All images depicted in this report were created to guide the graphical transcription of the DA physiology backbone. Some images were created based on the examination of control flow diagrams of engineering and physiology models (Van de Vegte, 1994), whereas other images were uniquely created for the purposes of this project.

.

A. Color Scheme

The graphical representation of the DA physiology backbone follows the form of a control flow diagram as represented in control flow systems (Van de Vegte, 1994).When depicting a physiologic model in the form of a flow diagram with many variables it can be challenging to differentiate between a value that arises from subroutine(s), a local variable, or a predefined variable. To alleviate this problem, the following color scheme is established as a standard. Figure 1 illustrates how this standard is implemented to depict the flow diagrams.

- A **red line** is used to represent a calculated input value sent from a preceding file(s)
- A **blue line** is used to represent a calculated output value that will be sent to another file.
- A **green line** is used to represent a constant that is predefined in the code. In other words, it is not calculated or manipulated by files in the model.
- A **black line** represents (local) values that are used in the scope of the current file.

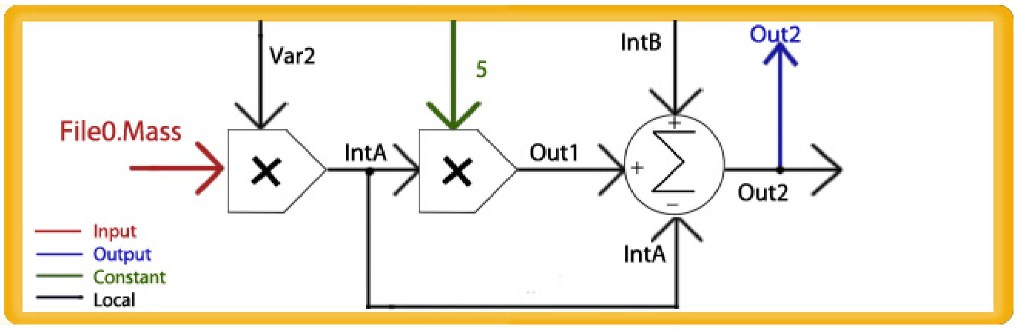

Figure 1: Proposed color scheme to distinguish different types of variables.

B. Mathematical Operators

The principle method of manipulating the functions, equations, and variables in the physiology backbone involves performing sequences of mathematical operations. These sequences of mathematical operations are used to define variables by ordering them in the structure of the XML schema. In order to represent the mathematical operations of addition, subtraction, multiplication, and division in a graphical manner, a combination of widely recognizable symbols and forms were adopted.

The *product operator* is represented by a pentagon symbol with an “x” in the center as illustrated in Figure 2. The purpose of the product operator is to multiply all inputted values with each other and output their result. In the example illustrated by Fig. 2, *Out1= IntA x 5*. Also, the product symbol always points in the direction of information flow. Declaring the product symbol to point in the direction of the flow of information promotes strict uniformity in the graphical representation and leaves little room for ambiguity.

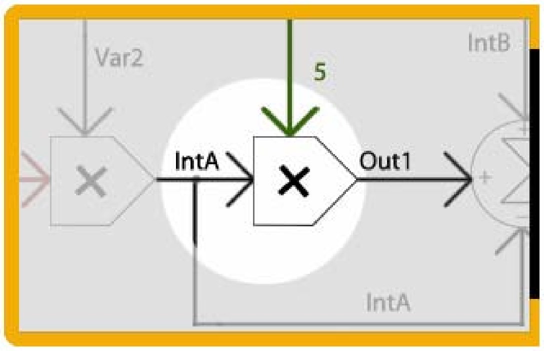

Figure 2: Product Operator multiplies 5 by IntA in this example.

The *summations operator* is represented by a circle with the summation symbol. Values that approach the summation symbol will be either added or subtracted based on the sign it comes in contact with. In the example illustrated by Fig. 3a, *Out2 = IntB + Out1 - IntA*. The summation symbol only allows three values to be added without disrupting the linear structure of the graphical representation. Knowing this limitation and the inherit complexity of physiologic functions and equations within HumMod, it is important to consider occurrences where more than three variables could be added or subtracted in one statement.

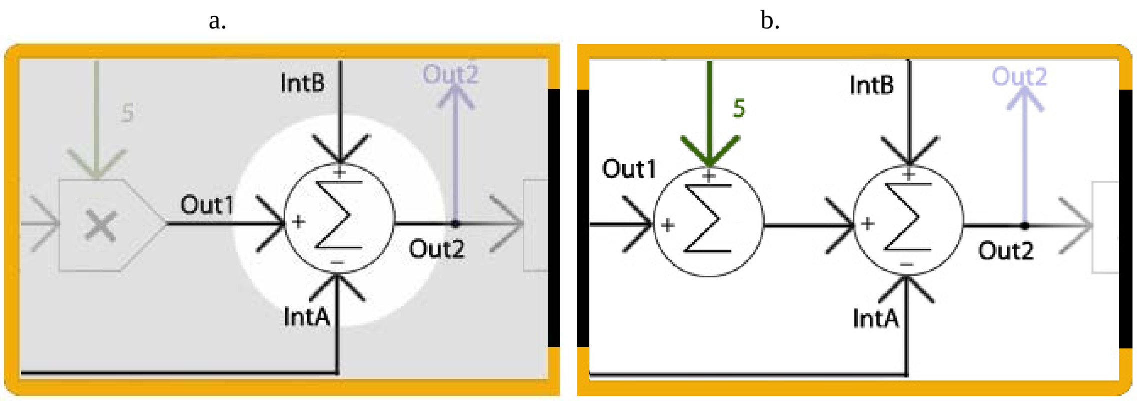

Figure 1: a.) Single Summation Operator is used when there are only 3 variables being summed.
b) Multiple summation operators are used in sequence when there are more than three variables being summed.

Multiple summation operators are used in occurrences where more than three values are being added to define a single variable. This is illustrated by Fig. 3b; *Out2 = 5 + Out1 + IntB - IntA*. Multiple summation operators are distinguished from two separate sequences of summation because the multiple summation operators are separated by unnamed intermediate variable(s)/arrow(s). Separate sequences of summations will be separated by named variable(s)/arrow(s).

The last of the basic mathematical operators is the division operator. Values that approach the *division symbol* divide one value by a second value. The value that is the denominator is the value that approaches from the bottom. This is illustrated by Fig. 4, where *Change= Float1/ File1.Out2*. The denominator was selected to approach from the bottom because when performing mathematical calculation the denominator is located at the bottom of the fraction. Maintaining this concept would be a simple and effective manner to implement universally understood procedures.

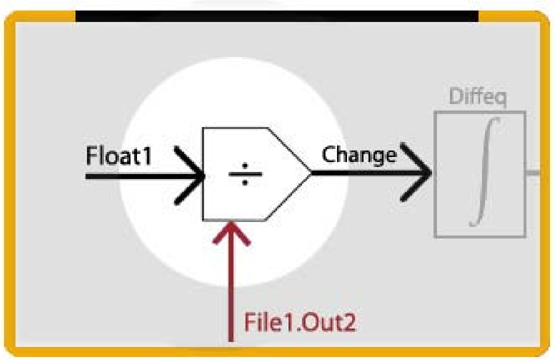

Figure 4: “File1.Out2” is the denominator because it approaches from the bottom

Conditional Statements are implemented widely throughout the XML of the DA physiology backbone in order to provide varying scenarios depending on the variables meeting a predefined condition or test. The XML schema defines several different tags to represent conditional statements that are unique to the DA modelers. They are unique in the sense that the nomenclatures used to associate properties to particular tags are not of the sort used by the web-based XHTML. The most frequently used conditional statements arise from *multiple IF statements*. To represent multiple “IF” statements in the graphical representation the conditions are placed inside of the brackets of the conditional operation placeholder illustrated in Fig. 5a. Different true conditions or *cases* are separated by a horizontal-curvilinear line as seen in Fig. 5a. For example, depending on the value of *PA1*, one of the three cases in Fig. 5a will define the variable *AUC*.

a. b.

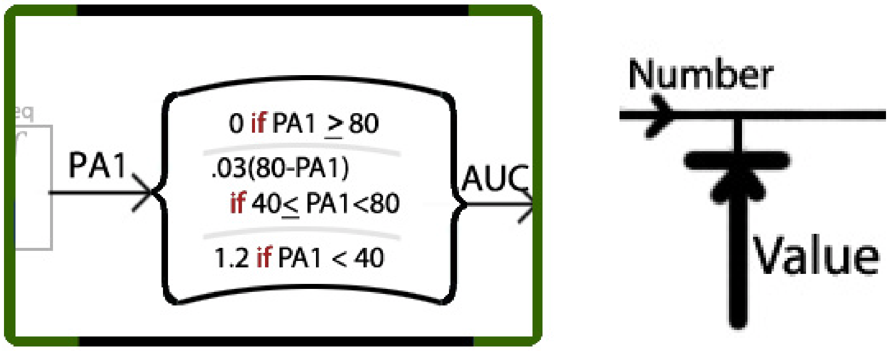

Figure 5: a.)The different cases are separated by a grey curvilinear line. b.) In the threshold symbol "Number" must be greater than "value"

Some simple conditional statements may also appear in the graphical representation as a *threshold limit*. When a variable encounters a threshold limit, the variable must be greater than the threshold to have any effect on the subroutine/system. If the variable being examined is greater than the predefined value of the threshold limit, then the value of the variable is passed for the next operation. However, if the value of the variable being examined is less than the threshold limit, then a value of *zero* is passed to the next operation instead of the calculated value. Fig. 5b shows how a threshold symbol is represented graphically.

The physiology backbone does not always perform actions that involve simple mathematical operations. Many subroutines contain algorithms that utilize differential equations. The only part of the algorithm controlled by the user of the HumMod modeling software is the integration interval (Dx). These algorithms are differentiated in the flow diagram by the name of the algorithm performing the integration. The names of the algorithms that solve for variables utilizing differential equations are *diffeq*, *stablediffeq*, and *backwardeuler*. Some algorithms are prefixed with the word "stable" because the mathematical functions they monitor and manipulate are known to be unstable at large integration intervals.

Since the algorithms perform multiple unseen mathematical operations the C++ source code for the HumMod modeler had to be examined. Without examining the true function of the algorithms, the graphical representation would not accurately portray any particular function. The user would know what value was inputted and what value was outputted, but not what process was used to get to that output Through the analysis of the source code, it was determined that the *diffeq(1),stablediffeq(2), and backwardeuler(3)* algorithms perform the following operations:

$$Y(n+1) = Y(n) + Y'(n) \times Dx \quad (1)$$

$$Y(n+1) = Y(n) + Y'(n) \times Dx \quad (2)$$

$$Y\,(n+1) = (e^{-F2*Dx})*Y(n) + (1 - e^{-F2*Dx})*K \quad (3)$$

Y(n+1) is the value of the variable at the next integration interval. The algorithms listed use their respective equations and predefined values in the physiology backbone to solve for changing values of a variable over time.

The methodology for incorporating the algorithms into the graphical representation involved creating a symbol that can be easily associated with an algorithm and its function, while conserving space in the event that several other variables are in focus. The symbol that represents the *diffeq*, *stablediffeq*, and *backwardeuler* algorithms is a circle with an integration symbol in the center as illustrated by Fig. 7. The variable that approaches the integration symbol in the flow diagram is the derivative (Y'n) of the variable being monitored by the algorithm. Fig 7 portrays how the integration "icon" is to appear in the zoomed-in and zoomed out view of a flow diagram.

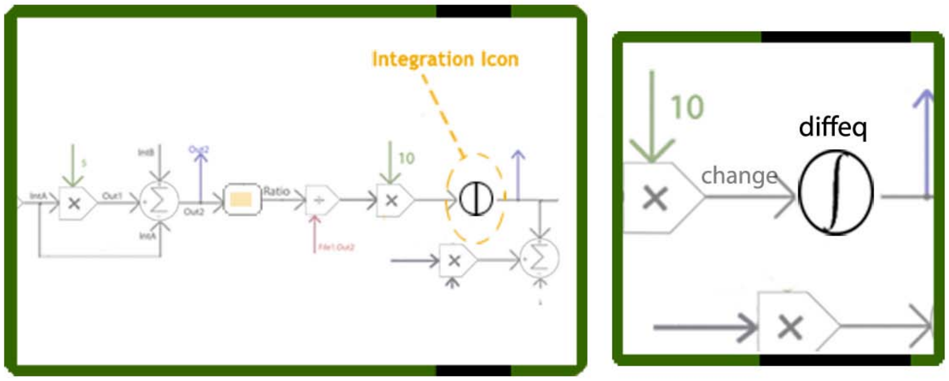

Figure 7: Integration symbol zoomed-out and zoomed-in.

Another function that does not involve simple mathematical operations is the *logarithmic* and *antilogarithmic* functions. The most common base of the logarithm function is log base 10. It appears in the XML code as "LOG10". In the flow diagram, it is recommended to represent it as "LOG" enclosed in a rectangular box as depicted in Fig.8a. When a logarithmic function defines a variable within the scope of a conditional statement, the true statement is simply written in the conditional statement placeholder. This is illustrated by Fig. 8b.

a. b.

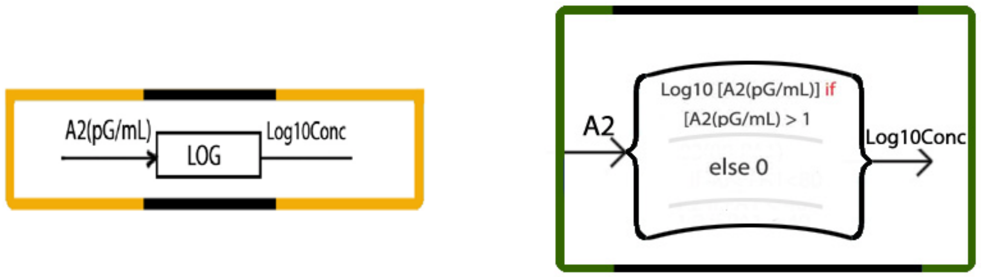

Figure 8: a) The log function appears in a rectangular box when used to define variables. b) The log function is written as it appears in the XML code when used in a conditional statement.

Exponents are represented in an A^B format. The value that approaches from the top is the power as illustrated in Figure 8a. The exponents symbol will always appear as A^B even if the variables are not called A and B respectively. The only exception to this rule is raising *e* to a power. As illustrated by Fig9b the e^x function is represented by EXP enclosed in a rectangular box. In Fig. 9b the output value would be equal to e^{AUX}.

a b.

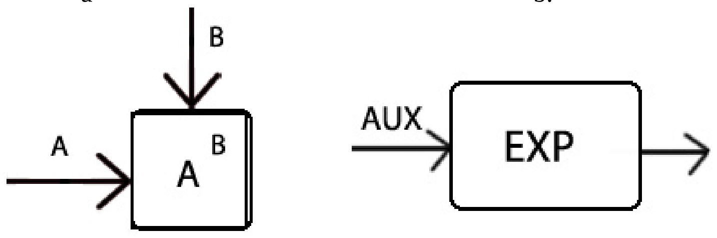

Figure 8: a) exponents symbol b) e^x symbol

C. Function/Curve Analysis

Through examination of the DA physiology backbone, I observed many occurrences where curves are estimated by the modeler. A curve is estimated by defining a series of x vales, y values and slopes at the corresponding xy coordinates. The DA modeling program then estimates a continuous curve based on the given information (Coleman, 2008). In order to represent this process in the graphical format effectively, only crucial information should be displayed. When considering the zoomed out view of a flow diagram, displaying all of the sets of coordinates and slopes that define the curve would not be practical in terms of space. Instead, the standard icon illustrated in Fig.10 is proposed to represent every curve function in the DA physiology backbone.

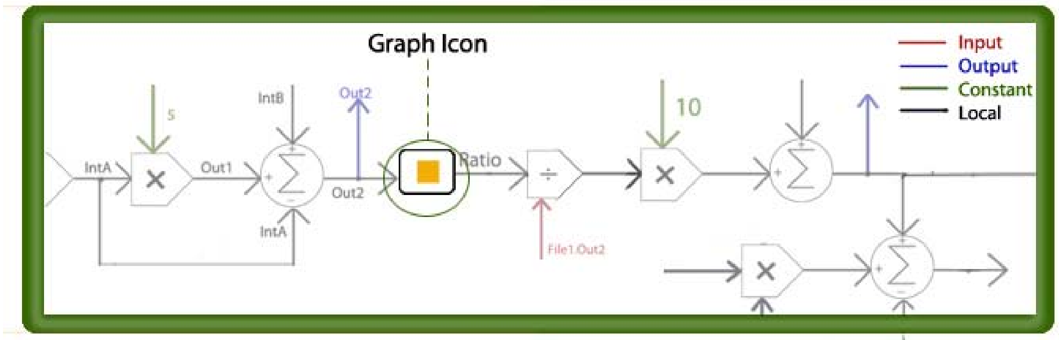

Figure 10: Graphical icon used when several variables are visible

Each curve can be identified by its predefined name in the physiology backbone. That name is positioned above the standard graph icon. As the range of variables being observed decreases the name of the function and a standard graph can be made, as illustrated in Fig, 11a. Initially it was determined that the graph in Fig.11a would be shown in all circumstances. However, after testing the image in a zoomed-out view of a flow diagram, the image disrupted the flow of variables perpendicular to the current flow of information.

Upon further investigation of the physiology backbone, I observed algorithms that the manipulate curves. These algorithms are *delay*, *stabledelay*, and *lag*. The operations carried out by these algorithms were determined through the examination of the QHP 2008 Schema (Coleman, Hester and Summers, 2008) and the HumMod source code. The *delay*(4), *stabledelay*(5), and *lag*(6) perform the following operations:

$$\text{Output}\ (n+1) = (e^{-K*Dt})\ *\text{Output}(n) + (1 - e^{-K*Dt})*\text{Input}(x) \tag{4}$$

$$\text{Output}\ (n+1) = (e^{-K*Dt})\ *\text{Output}(n) + (1 - e^{-K*Dt})*\text{Input}(x) \tag{5}$$

$$\text{Output}\ (n+1) = c * \text{output}(n) + (1 - c) * \text{Input}(x+1) \tag{6}$$

The *stabledelay* algorithm performs the same operations as the *delay* algorithm. The only difference is the *stabledelay* algorithm allows a maximum Dt (time step) to handle a known instability in the curve being examined. The *lag* algorithm adds complexity to the delay algorithm because its output(n) must also be modified through operations defined by math blocks inside of the physiology backbone.

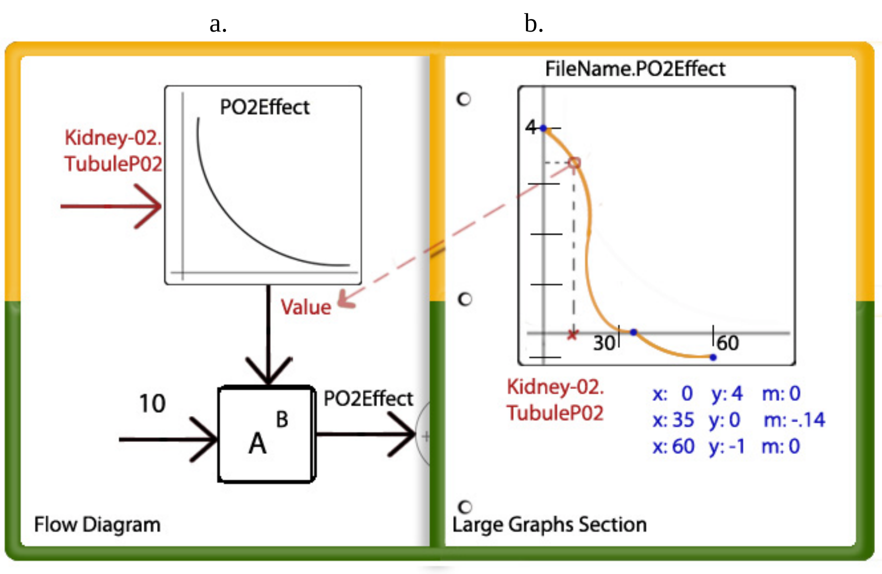

Figure 11: a) Standard graph symbol used for zoomed-out view of a flow diagram b) process of determining the value to return for the next operation.

V. Conclusions

In the current state of the model, researchers must learn the XML markup or consult with a programmer who is familiar with the markup language in order to effectively use or extend the capabilities of the model. A standard graphical representation is proposed to express the DA physiology backbone model into an easy to understand flow diagram. This standard can be integrated with a software application that can automatically transcribe the physiology backbone into more visual and interactive interface.

The transcription of HumMod into a graphical form can reveal the interworking of the physiology backbone and enhance the ability of researcher who are developing partial models to extend its capabilities. Also, the effectiveness of these researchers can be further increased due to the decryption of once hidden operations carried out by the algorithms in HumMod. Through the revealing of the operations carried out by the various algorithms, researchers can tie their work into and utilize the full capabilities of the modeling software at hand. On a broader note, our work is contributing to DA physiology backbone being a platform developed by researchers across the US, and potentially worldwide.

Acknowledgments

Lealem Mulugeta (Internship Mentor)
Robert Brock
Ramachandra Srinivasan
Nick Skytland
Galen Kreutzberg

References

Coleman, Tom. *Digital Astronaut (V1.0) C++ Source Code.* Mississippi, November 28, 2008.
Coleman, Tom, Robert Hester, and Richard Summers. *QHP 2008 Schema.* Vols. 1-5. Jackson, MS, 2008.
Fitzgerrell, D. G. *Design Specifications for the Whole Body Algorithm.* Houston,TX: General Electric Technical Information Release TIR -741-MED-4025 (NASA/JSC Technical Library), 1974.
Fitzjerrell, D. G., D. J. Grounds, and J. I. Leonard. *Study Report on the Major Physiological Subsytem Models: An Approach for Developing a Whole-Body-Algorithm.* Houston, TX: General Electric Technical Information Release TIR -741-MED-4021 (NASA/JSC Technical Library), 1975.

Kay, Franklin J. *Study Report: The Development of a Whole Body Algorithm.* Houston,TX: General Electric Technical Information Release TIR -741-MED-3058 (NASA/JSC Technical Library), 1973.

King, A. C., J. Billingham, and S. R. Otto. *Differential Equations.* Cambridge: Cambridge University Press, 2003.

Sedgewick, Robert. *Algorithms in C++.* 3rd. Vol. 1. 2 vols. Menlo Park, CA: Addison-Wesley, 1998.

Van de Vegte, John. *Feedback Control Systems.* Englewood Cliffs, NJ: Prentice-Hall, 1994.

White, Ronald J. *A Long Term Model of Circulation.* Houston, TX: General Electric Technical Information Release TIR -741-MED-4021 (NASA/JSC Technical Library), 1974.

White, Ronald J. *A Simple Model of Fluid Flow and Electrolyte Balance in the Body.* Houston,TX: General Electric Technical Information Release TIR -741-MED-3010 (NASA/JSC Technical Library), 1973.

White, Ronald J. *Acid-Base Homeostasis in the Human System.* Houston, TX: General Electric Technical Information Release TIR -741-MED-4017 (NASA/JSC Technical Library), 1974.

White, Ronald J. *Fluid and Electrolyte Control Systems in the Human Body.* Houston,TX: General Electric Technical Information Release TIR -741-MED-3032 (NASA/JSC Technical Library), 1973.

White, Ronald J. *Summary Report On a Basic Model of Circulatory, Fluid, and Electrolyte Regulation in the Human System Based Upon the Model of Guyton.* Houston, TX: General Electric Technical Information Release TIR-741-MED-3042 (NASA/JSC Technical Library), 1973.

White, Ronald J., and J. M. Nouchedehi. *Mathman a Users Manual.* Houston,TX: Managementand Technical Services Company Technical Information Release TIR 2114-MED-1007 (NASA/JSC Technical Library), 1981.