# System Dynamics Calculator

## *A new approach to "visual" computing*

ABSTRACT: This document describes a System Dynamics Calculator – an entirely new "visual" approach to computing for ordinary people. "Systems Thinking" and System Dynamics (SD) are traditional analysis and model simulation methods to gain insight into complex problems with a number of interacting dynamic factors. SD is used to simulate commodity market behavior that might result from global conflict for example. While SD provides insight into which factors have the greatest impact in a model, it does not solve problems. Also, its application is currently limited to a relatively small group of specially trained professionals. SD has the potential for many more to accomplish much more – but not with the current methods and tools.

The System Dynamics Calculator is a proposed software-based tool that will enable people who are unfamiliar with SD to solve a variety of complex problems and get real answers instead of just insights. The calculator will run on a broad range of computers from a desktop PC to a handheld touch screen device. This calculator represents a significant improvement over SD methods in use today with visual cues and an array of analysis tools that augment conventional SD techniques to give real answers. The calculator does not require comprehensive knowledge from the user to describe a problem or explore a solution. Unlike current SD tools, it can directly access and incorporate web-based information on the fly. An iPhone, G Phone or Pre owner running the System Dynamics Calculator application would be able to calculate the change to their 401K deduction necessary to be able to afford improved medical insurance and a new car, and then immediately see what the likely impact to their planned retirement date would be. In order to realize the greatest benefits of the System Dynamics Calculator, the adoption of model methods by an international standards body will be required in order to utilize models from multiple sources that are not proprietary or parochial. Easy access and use, a variety of readily available calculator models and data sets online, and compatibility with already purchased hardware platforms will bring this sophisticated problem solving tool to a significant percentage of the population to get answers to real problems from the trivial to the life-impacting.

Brian Crowe
President, TELE–WORX
2432 Kingsley Dr.
Grand Prairie, TX 75050
USA

+1-972-602-7922
brian_crowe@tele-worx.com

# TABLE OF CONTENTS

# THE SYSTEM DYNAMICS CALCULATOR

The System Dynamics Calculator has a user interface that runs on a variety of platforms including smart phones and PDAs, laptop computers and desktop computers. The calculator operates on a model of a dynamic system with methods and techniques that are to be defined by a global standard body to insure compatibility with models that are created by other compliant model builders around the world. The calculator presents a rich, visually appealing interactive environment for the user to explore the behavior of a dynamic system, project future system behavior and compute optimal solutions to complex real-life problems. The System Dynamics Calculator includes a Model Builder process and a Model Simulator process.

The System Dynamics Calculator immediately prompts the user to select and load a model to simulate. The calculator presents at least one of several options to the user to begin a simulation:

- select a model from a library of locally installed models
- select a model from local media such as a memory card or a USB storage device
- select a model from remote libraries available over the Internet
- build a new model from scratch
- use a conventional calculator

Once a model is loaded, the calculator's Model Simulator identifies and loads essential initial data for the model:

- user-input values for constants and for initial stock levels
- local datasets such as TAL files exported from personal finance software
- remote datasets from websites such as US Census Bureau or the New York Stock Exchange

Next, the Model Simulator prompts the user to specify other simulation criterion and options that are desired. The model's simulation strategy allows for statistically shaped random number generation, multiple trials and a host of other tools to produce meaning information and solutions for the user. Typical simulation options that the model supports including the following:

- beginning and ending simulation time and trial time increment
- speed of the simulation such as fastest, or one trial per second, for example
- goal value for a stock or flow level by adjusting other designated stocks or flows
- optimizer trial values for constants and other variables to be adjusted through a range
- number of simulations if something other than the default is desired
- confidence or statistical deviation specification such as 3-sigma for a goal value that is the result of one or more random number input values in the model

The results of the simulation can be presented to the user in customizable graphs and charts for any stock, flow or transformer value in the model

The user is able to query the Model Simulator to analyze the simulation results further to determine something of the significance from the results. For example, the user is able to:

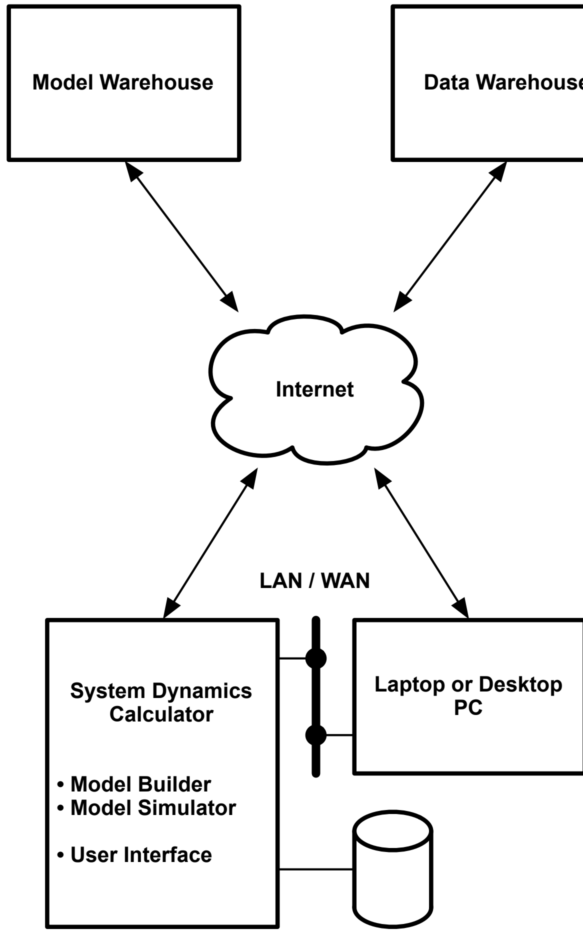
- calculate the min, max, mean and standard deviation for any result level
- calculate the correlation between any two stocks or flows
- graph velocity and acceleration of stocks and flows
- graph the sensitivity of stocks and flows relative to a specified stock or flow
- designate variables to be random values shaped to a specified probability distribution
- solve for a level = a target by adjusting a specified variable in a specified range for a solution
- solve for a level = a target and a second criterion such as a minimum or maximum for another level value by adjusting a set of variables in a specified range for a solution

## Basic Calculator Architectures

Calculator resources may limit the calculator's feature set. The following basic considerations are essential for calculator design and development:

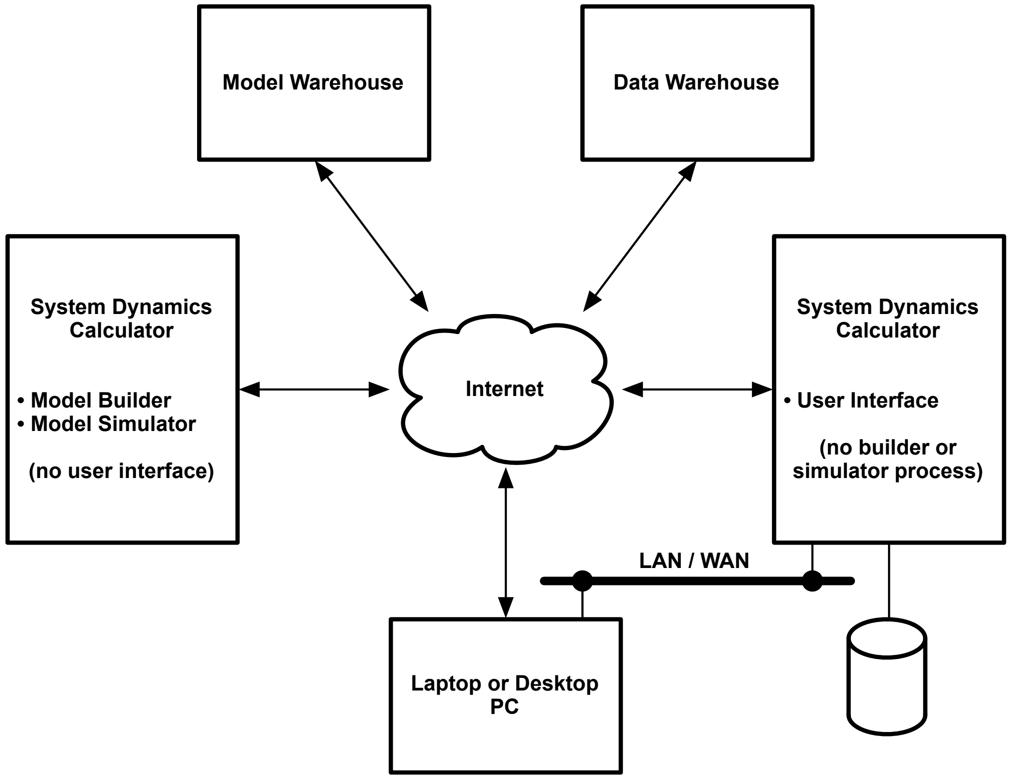
- ample display area — the display area will dictate whether the user can easily build a model, create an abstraction layer and manage simulation scenarios of any significant complexity
- mouse or touch screen — supports the requirement to interact with menus, to display significant model detail on demand for a portion of the model, and to select and move icons on the screen; this hardware feature will impact the required operating system feature support to incorporate into the application
- internal memory — supports the requirement to store a model and the simulation results; internal nonvolatile storage also allows for an inventory of models and results retained for future use
- external storage interface — supports the requirement to transfer models and results between the calculator and another device that relies upon calculator features such as a memory card slot or a USB interface
- network interface — supports the requirement to load models and data sets from remote "warehouses" accessible over the Internet or from local PCs

Several calculator architectures are possible. The calculator may be a locally stored and run application as depicted in the following drawing:




In the above implementation example, the calculator is autonomous and fully functional without an Internet interface.

A “Cloud Computing” architecture with a local web-based user interface and a remotely executed application is illustrated in the following drawing:




Both of the preceding drawings illustrate the potential use of the Internet by SD Calculator users to access libraries of standard compliant models, libraries of factories and scripts, and raw data in convenient formats from a variety of sources. Warehousing and serving libraries and data sets creates a new business opportunity for Internet-based computing services and remotely served software that supports the SD calculator with tremendous resources for the individual user. This opportunity adds significantly to the business of remotely hosting the Model Builder and Model Simulator applications in the “cloud computing” arrangement.

## Calculator User Interface

The user interface for the System Dynamics Calculator is color-oriented, graphic-intensive, user configurable and “computer-like” in many respects. The following are the fundamental components of the calculator’s user interface:

- application menu
- help menu
- model builder palette
- model simulator control strip
- canvas

The calculator application is configured and controlled with a single application menu. The following basic functions and operations are controlled through the calculator's primary application menu interface:

- create and name a new model, a new associated abstraction layer, or a new simulation
- save a model with abstraction layers and simulation definitions in local memory, to a storage device or to a network location
- save the current "state" for a model of anchored and torn-off palettes and their locations, the control strip visibility and location, and every page of the canvas - models, graphs, tables and logs
- select, load and open a model with abstraction layers and simulation definitions from local memory, a storage device or network locations
- select, load and open a model with abstraction layers and simulation definitions in its "saved state" from local memory, a storage device or network locations
- select, load and open a conventional calculator
- create, name and save a new factory from a defined model fragment
- create, name and save a new script
- create, view, save, print and export a report – any one or more of a model drawing, custom graph, custom table, simulation log (the value log, threshold crossing log and error log)
- clear the canvas
- administer calculator application preferences
- purchase and save a model; purchase and install a factory package or a script
- import and export a model with its associated abstraction layers and simulation definitions in a variety of convenient file formats
- quit the calculator application

The calculator application preferences are set from a window that is commanded from the application menu. The calculator preferences include the following:

- define and manage user accounts
- define and manage trusted network services
- define and manage file sharing policy
- define and manage other security policy
- set the user turn-on preferences to 1) restore the previous saved "state" (to pick up where you last left off), or 2) restore a default starting "state" (a clear canvas with a default anchored palette and a defined set of placed torn-off palettes), or 3) a clear canvas with no palette displayed
- set the default styles for text, text boxes, symbol size and line width, canvas color and background
- set the user localization preferences including currency system, units of measure system and number system (may be an operating system level function)
- set the calculator location "time and place" (may be a device level GPS function)

The user can perform the following operations on the palette:

- create, edit and delete a palette
- create, edit and delete a hierarchy of palettes
- select a palette from a list of palettes
- add to, remove from and reorganize a palette
- tear off a palette
- reposition the current palette
- rotate the current palette to accommodate horizontal or vertical orientation of the calculator
- resize the current palette
- anchor the current palette to any display edge and to the application menu
- show/hide current palette

The user can perform the following operations on the control strip:

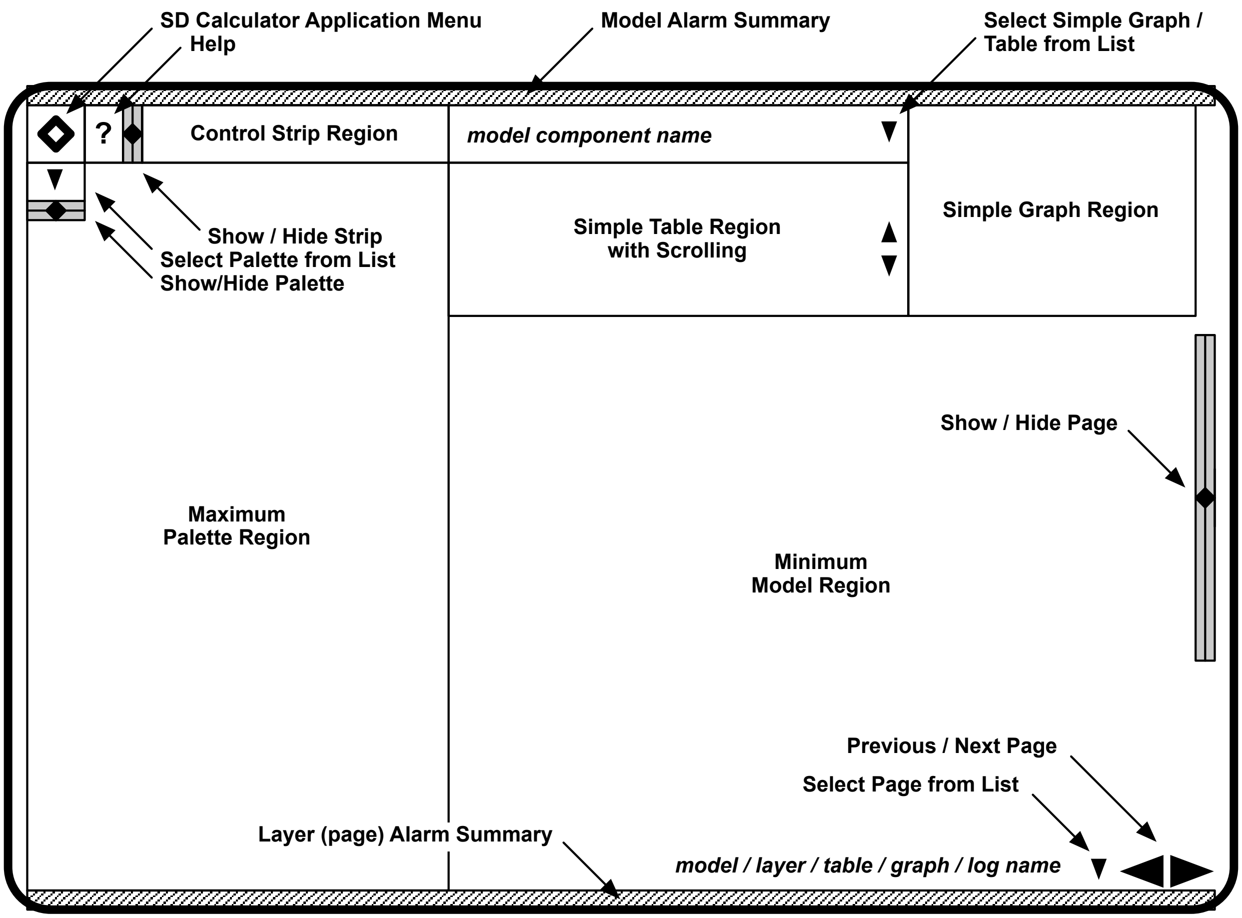
- tear off a control strip

- reposition the control strip
- rotate the control strip to accommodate horizontal or vertical orientation of the calculator
- anchor the control strip to any display edge and to the application menu
- show/hide the control strip

The canvas holds the primary model, its associated abstraction layers, simple graphs and tables, custom graphs and tables, and logs. The canvas has a paging system, a coordinate system, a default point of origin and default scale. The user can perform the following operations on the canvas:

- move the canvas relative to the physical display area
- zoom or scale the canvas
- return to the default point of origin and scale
- page the canvas – display the primary model layer, an abstraction layer, a custom graph, a custom table and any of the simulation logs
- select a canvas page – select the primary model layer, an abstraction layer, a custom graph, a custom table and any of the simulation logs from a list of canvas pages available

The following drawing illustrates basic elements of a possible user interface implementation:



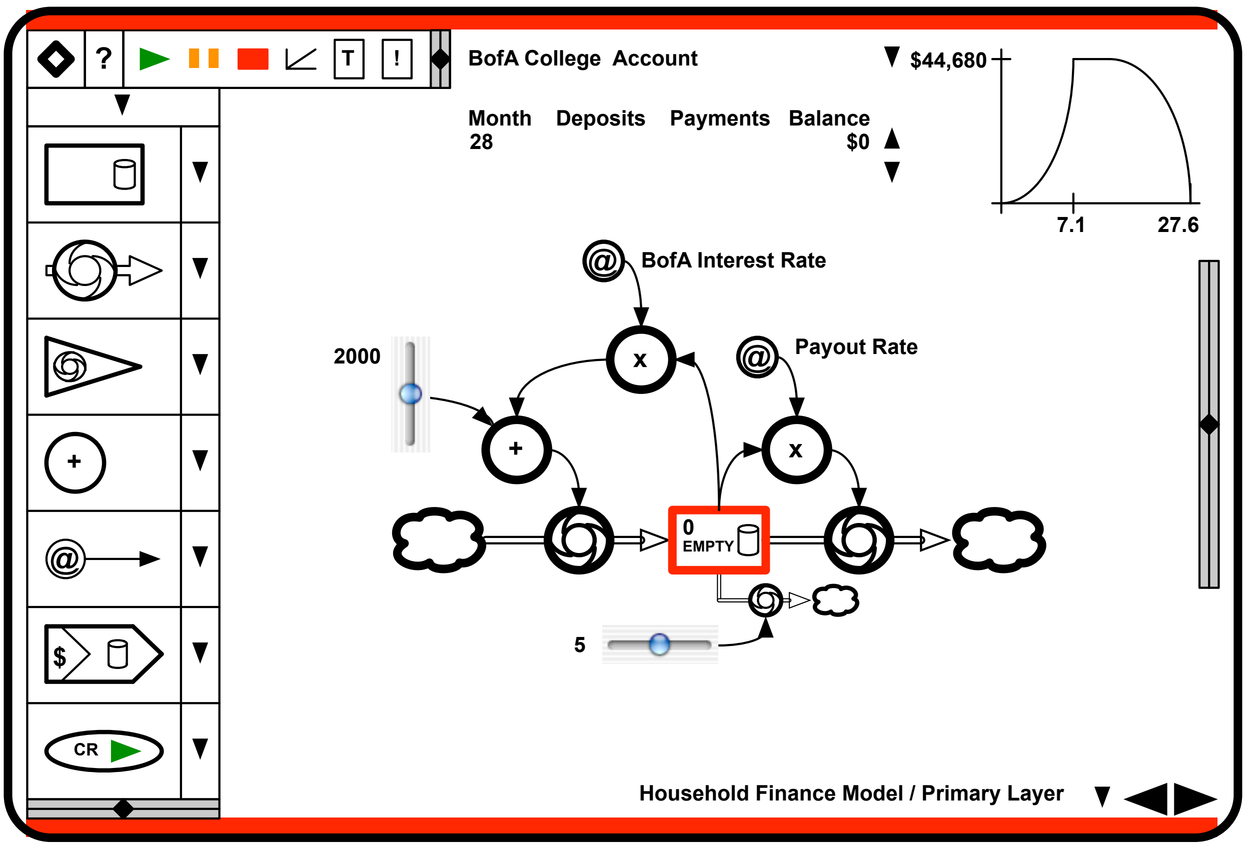

The above drawing illustrates the following user interface characteristics:

- an application control (the large diamond) to select the secondary actions of the calculator (after building a model and running a simulation) such as setting preferences
- regions potentially occupied by a control strip, palette, simple graph and simple table and the model

- controls to browse the canvas page by page
- controls to show / hide the control strip, palette and the current page on the canvas
- controls to select canvas pages, palettes, simple graphs and simple tables from lists

Elements of the previous drawing are more tangible when a model is displayed and a simulation has been run. In the following drawing, these user interface elements are illustrated:

- a model builder tool palette for stocks, flows, transformers, factories, converters, and variables
- a model simulator control strip to run and stop the simulator, build a graph, examine the simulation log
- a model that is animated, editable, movable and resizable
- a simple graph associated with a designated model component
- a simple scrollable table associated with the same designated model component
- model, layer and component names
- model, layer and component alarm summaries




It is important to note some details that are illustrated in the above drawing:

- color cues assist the user with controls (green to go, red to stop)
- color cues assist the user with simulation interpretation (red is a critical simulation alarm)
- alarm summary information is present for the model, the layer, fragment and the component
- graphic pictures are prominent and text is sparse
- the model is in the center and all other tools and information are on the display perimeter

## Conventional Calculator Functionality

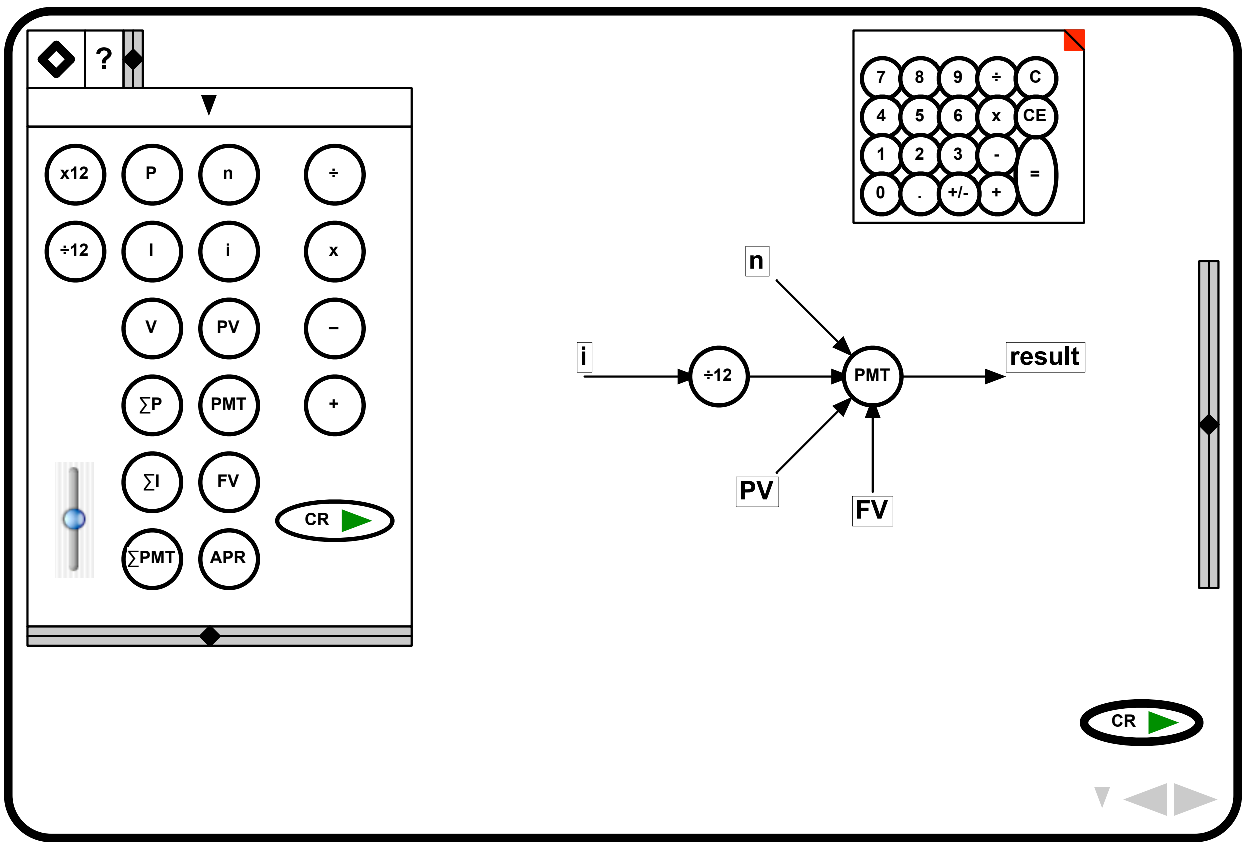
The SD Calculator includes a set of conventional calculator implementations available through a set of custom palettes that gives the user convenient and extensive calculator functionality on the canvas. The following conventional calculators are provided:

- Basic Arithmetic Calculator
- Comprehensive Financial Calculator
- Comprehensive Scientific Calculator
- Comprehensive Programming Calculator

The set of custom palettes could include the following custom function collections to place converters in computation-intensive models more conveniently:

- Time Value of Money (loans and annuities)
- Time Value of Capital Assets (depreciation)
- Localization (time conversions, units of measure conversions and currency conversions)

In the following example of a possible user interface, two calculators are displayed:




The two calculators illustrated above also demonstrate several behaviors that are necessary for efficient calculator operation by the user. One calculator is a simple, conventional arithmetic calculator with a number pad of buttons and a column of function buttons. The other calculator is a loan payment model calculator. The loan payment calculator requires the placement of two converters from the custom palette – one is the “PMT” loan payment function converter, and the other is the “÷12” converter that converts the annual loan interest rate to a monthly interest rate.

The model builder automatically draws the variable lines and labels them consistently with the converter behavior that is defined for the custom palette converters. The model builder also automatically connects the “÷12” converter when that converter comes within a threshold distance to the “i” interest variable line as a standard behavior. At this point, the user enters the numbers that define the annual interest rate “i”, the number of monthly payments “n” and the loan value “PV”; if the loan is to be completely paid over time, the final value (future value) “FV” is zero, and it is implicitly zero if the user does not enter a value for this variable.

A number entry method is automatically presented when the user clicks on or gestures on a variable label; the variables are defined as numbers, so only the number entry method is presented. If a number entry method is already in place, it is utilized. In the above example, the user has a arithmetic calculator displayed, and numeric variable entry can be made from the other calculator’s number pad.

Finally, the user runs the loan payment calculator by clicking or gesturing on the “PMT” converter’s result variable line. In the above example, the user has also placed a script to run the model every time an entry is completed – every time the user clicks or gestures on a variable value – or every time the user clicks of gestures on the “=” button on the arithmetic calculator.

The user dismisses the basic arithmetic calculator with a “go-away” button that is the red rectangle in this illustration.

## Help

The SD Calculator has a rich set of help features including tool tips, topical help, wizards and a variety of tutorials. These help features are unobtrusive, and they are easily available to the user on demand. The help features are supplemented with web links to additional resources.

The SD Calculator provides the following help features:

- a help tool at the top level of the user interface to select:
  - the help topic index
  - an objective-based wizard from a list (to calculate a loan payment, for example)
  - a tutorial from a list
- a mouse-up tool-tip for every control and palette symbol
- a list of related help resources including additional topics from the index, wizards and tutorials, and web url’s for supplemental information from the manufacturer when this information is available

## Security, Localization and Personalization

The SD Calculator is secured, personalized and localized. The calculator is configured with a calculator name and credential, a user name and credential, and a user location that sets the time zone, default language, default currency and default units of measure systems and some other attributes.

The SD Calculator must be a secure device because of the valuable nature of personal data that the calculator potentially stores and utilizes. To facilitate casual usage, an effortless login method is highly desirable such as an RFID user token, a biometric method or some other automatic mechanism that logs in the user when the user handles the calculator. To guard against theft of the user's token, the user may elect to require authentication for certain operations or for the use of certain data.

The SD Calculator provides the following security features, functions and policies

- root user account
- normal secure user account
- multiple simultaneous users
- security policy for secure user accounts
- security policy for unregistered users and the functionality they have access to
- security policy for multiple simultaneous logins (logging in a second user without logging out the first user or securing each user's data from other users)
- security policy for user switching (logging in a second user without logging out the first user, but securing each user's data from other users)
- security policy for user login, logout, sleep, wake and automatic shutdown
- security policy to create and delete a user ID and to create and edit a password
- security policy to terminate file sharing and to auto-logout the user after an idle time
- security policy to encrypt user data files and network data transactions
- data encryption method for securing user data files and network data transactions
- credential "key ring" for the calculator and for each registered user
- file sharing credential for authorized computers and authorized users
- file sharing credentials from authorized computers and authorized users

The SD Calculator maintains the following user data related to security:

- user ID and password
- user name, address, birthday
- user security policy for access to user data by the calculator, by explicitly authorized computers, by other computers, by explicitly authorized users, and by other users (simultaneous or switched users)

The SD Calculator maintains the following user data related to localization:

- calculator location, automatically set by GPS
- user location; default is the user address by user address
- user alternative location, manually set by user (persistently overrides the user address if set)
- language preference; default is the national language at the user address
- time of day preferences and daylight savings time rules; default is the local custom at the user address
- currency preference; default to national currency at the user address
- system of units preferences for distance, mass, volume and temperature; default to local custom at the user's address, and the user may persistently override any of these default preferences

The user can personalize the canvas:

- revise the current default canvas color and canvas texture
- revert the current default color and texture back to the previous default
- reset the current default color and texture back to the manufactured default
- display a background picture with a configurable transparency

## Software Versioning and Maintenance

The SD Calculator presents the user with a large number of potential features. A number of functional separations are essential for the manufacturer of the software to differentiate the product. A number of considerations are essential for version control and keeping the calculator up to date.

Privilege management for categories of functionality is necessary to differentiate the product and preserve opportunities to upgrade. Privilege is managed for the following SD Calculator versions at the time of manufacture:

- "Modeler" product enables everything
- "Calculator" product denies the ability to model a dynamic system
  – deny the ability to pick and place stocks, flows and transformers from the palette
- "Modeler Trial" product runs for a preset number of days, and then assumes Limited Calculator Trial version functionality
- "Limited Calculator Trial" product denies the ability to model a dynamic system and the ability to output in any way other than to display
  – deny the ability to pick and place stocks, flows and transformers from the palette
  – deny the ability to pick and place deny the ability to "Save", "Save As…", "Share", "Export" or "Print"

Every version of the SD Calculator software will be able to engage in socialization and partner commerce as a consumer by being able to load or import models, factories and scripts developed by other users, by running simulations of the models, and by using the factories and scripts in their own models and simulations. For this workflow, each version of the SD Calculator is fully capable.

SD Calculator application software and factory packs are installed, uninstalled, updated and upgraded in an administered process. Software is installed from local media or downloaded over the Internet from a remote software volume.

Updates may be manually requested or scheduled. If an update is available, it will be delivered and installed in an administered process that requires the user's approval. Updates will typically be downloaded from the manufacturer over the Internet. The calculator will recover from network failures and installation failures. Once downloaded, the update software will remain ready for installation until the user approves the installation process.

Upgrades from trial software to featured software or from the Calculator to the Modeler are accomplished with a manual serial number entry or a software token that is stored in the calculator, and no further software download is necessary unless a maintenance update is available.

# THE MODEL BUILDER

The Model Builder is one of two basic components of the System Dynamics Calculator. The Model Builder offers the user a convenient graphical user interface to construct a model. Some users will not want to build a model or modify an existing model, and some calculator platforms will not afford the user interface capabilities necessary to make the model building process possible except fr the simplest of models.

The Model Builder immediately presents the user with a blank "canvas" and a "palette" of model components. The Model Builder allows the user to pick model components from the palette, place them on the canvas, connect them to other model components, define rules and relationships, and describe the nuances of behavior that will make the model meaningful. The Model Builder preserves graphical connections between components to allow the user to easily clean up and rearrange the model to keep it visually organized and uncluttered. The Model Builder also ensures that the model is logically rigorous and that the user describes all the essential details as the model is built.

## An Introduction to the Model

A System Dynamics model is fundamentally based upon stocks of "goods" and flows of a "good" between stocks over time. Flows move a calculated quantity of a "good" between stocks in each unit of time during the model simulation. Stocks and flows are typically connected in a "causal loop" to model cycles of cause and effect relationships and dominant constructive and destructive forces in the system. Stocks and flows have limitations, and flows have controls. This modeling technique also employs feedback in the model to characterize the dynamic behavior of a system and capture real-world behaviors that are relevant to the model objectives.

A system dynamics model consists of a small set of model components:

- Stocks — goods storage with flows of the good in and out
- Flows — goods transport between stocks
- Transformers — goods transformation from one or more good to another
- Converters — a numerical calculator function chosen from a comprehensive set
- Variables — internal, external and user provided model values
- Factories — functional building blocks – macros – of many model components

A model component has the following characteristics:

- Names — a user-assigned label and also a computer-assigned label
- Type — the type and subtype of model component
- Symbol — a basic shape with visual cues for subtype, value, status and activity
- Location — the symbol's coordinates on the model image
- Interfaces — the set of connections to other model components that are allowed
- Connections — direct associations with other model components
- Indicators — values, summary, animation, mouse overs, graphs and tables
- Attributes — the set of characteristics that are configured by the user
- Formulas — the mathematical formulas that operate on the connected components
- Value — the attribute that is calculated by the component from formulas
- Cost — the per interval / per unit cost
- Alarms — threshold crossings, illegal operations or model malfunctions

| | |
|---|---|
| • Security | the permission for the user to edit configurable characteristics |
| • Notes | simple text generated by the user and placed anywhere to explain things |
| • Other Behaviors | characteristics in addition to indicators, interfaces and errors detected |

The model displays the summary, or the most severe status or alarm level for the entire model, for each layer, for each fragment, and for each component:

| | |
|---|---|
| • Summary Status | static (ST – the model simulator is idle)<br>not alarmed (NL – the model, fragment or component status is normal)<br>alarmed (a MN minor, MJ major or CR critical – a threshold crossing or an error has been detected) |
| • Threshold Alarms | threshold crossings between normal, minor, major and critical ranges |
| • Error Alarms | errors that produce misleading simulation results or reflect model malfunctions or calculation errors |

The model can be layered into “abstractions”. The user can create an abstraction layer to facilitate analysis of a model without the distraction of the primary model in the application foreground. The following abstraction layer is defined, and the user can define other abstraction layers:

| | |
|---|---|
| • General Abstraction Layers | separate model layers that are useful for focused analysis of the model |
| • Cost Abstraction Layer | all modeled stocks, modeled flows, transformers and factories have a cost attribute that can be defined and accumulated by a cost account finance factory in a separate model layer |

The model can be augmented with components that simplify its display and simulation

| | |
|---|---|
| • Fragments | convenient subsets of the model represented by a single symbol |
| • Scripts | a small program to perform specified actions in a sequence |

## Basic Drawing Functions

Many model-building operations resemble the actions performed with a typical computer 2D drawing application. A suite of basic drawing tools and features are required to build understandable and visually appealing models include the following features:

- a palette of model component shapes to select from and place on the canvas – the standard set of system dynamics model component types and their derivative subtypes
- a text tool for user input to define and place formatted text in a text box on the canvas
- select / unselect / select another / select all by each and by a region
- copy / paste / delete
- multiple undo / redo
- group / ungroup
- move to front / move to back
- units of measure such as pixels, metric (millimeters) or English (inches)
- grid feature that is configurable to display or hide major and minor gradients and establish their ratio
- guides for alignment of a number of model components
- snap to grid / snap to guide
- align selected to guide feature for edges and centers
- attach / detach lines (flow lines and variable lines) to shapes (stocks, flows, transformers, factories and fragments)
- scale shape on canvas by corners and edge relative to opposite or center
- scale canvas by percent / scale canvas to fit for display and to print / reset scale to default
- report template with a title, header, footer, multiple page support, page numbering, and date and time stamping

## Basic Model Building Functions

Basic model building tasks begin with selecting a model component from the component palette and dragging or gesturing it onto the canvas. The symbols displayed on the palette are the last used; if the desired symbol is not displayed, it is available with a pop-up or pull-down list to select alternatives from. When the desired component is identified, it is selected and dragged onto the model canvas with a mouse movement or gesture, and it is placed with another mouse movement or button click or gesture.

Several techniques are essential for rapid model building such as shortcuts for component connection and expansion. The following user interface techniques are incorporated into the model builder:

- click or gesture on the canvas for the default text tool to present an alpha-numeric input method and locate the point of the text
- click or gesture on a model component on the canvas to select the component
- shift-click or gesture on a component to unselect a component or select another component
- drag or gesture on the canvas to select an area of the model
- drag or gesture on one or more selected components to move them on the canvas
- drag or gesture from the palette to the canvas to add a model component to the canvas
- drag or gesture a modeled stock, modeled flow or transformer from the palette and drop on an existing flow on the canvas to bisect the existing flow and insert the new component into the model
- drag or gesture a converter from the palette and drop on a variable line on the canvas to add an intermediate computation to the model
- click or gesture on the palate to select a flow or variable model component, then option-click or gesture on a component on the canvas to select a "from" component, and finally drag to the "to" component to connect two components with the flow or variable that is displayed on the palette
- option click or gesture on one end of a flow on the canvas and drag it to detach it from the model component it is attached to and drag it to another model component to attach it
- double-click or gesture on a converter operand variable line to add another operand to the converter; this technique is convenient with a custom palette that supports conventional calculator applications or when SHOWUSERINPUT is enabled for a converter that supports operand expansion
- click or gesture on a converter variable line label to enter a constant value
- click or gesture a model component on the canvas and drag off the edge to delete it
- click or gesture on the canvas (but not on a model component) and drag past the display edge to view the next canvas – to view another model layer

Several techniques are essential for efficient editing of a model component's attributes:

- double-click or gesture on a component on the canvas to edit the component – to change the component subtype or modify any of its other attributes
- double-click or gesture on a character string in a formula or other attribute in a model component definition window to highlight the string, and then type to replace what was highlighted
- option-click or gesture on a function name or function place indicator to expose a function browser

Several style specification options are required features to present significant model characteristics and to maintain model clarity:

- symbol size style of small, medium, large and custom with a default
- minimum component spacing style specification including a default
- semicircular or right-angled variable lines style specification including a default
- semicircular or right-angled flow lines style specification including a default
- the ability to revert back to the current default style preferences for the entire model
- the ability to revert back to the current default style preferences for a selected subset of the model
- the ability to edit the current default for any style preference

- the ability to reset the current default to the manufactured default for any style preference

Several techniques are required to accommodate making and changing connections between model components. In particular, techniques supporting the following flow connectivity behaviors are required for flow lines and variable lines:

- drag an end of a line within a certain close distance of a modeled stock, custom modeled flow or transformer to connect the line to the model component – at a threshold distance, the line will snap to the edge of the model component and a modeled connection will be created
- drag and end of a line from the edge of a modeled stock, custom modeled flow or transformer up to a certain distance without disconnecting the line – up to a threshold distance, the line will snap back to the edge of the model component
- drag and end of a line along the edge of a modeled stock, modeled flow, transformer or converter to the other side of an adjacent line to reverse their order on the model component's edge
- drag an end of a line more than a certain distance from the edge of a modeled stock, custom modeled flow or transformer to disconnect the line – past a threshold distance, the line will disconnect from the edge of the other model component and move freely according to the user's mouse movement or gesture and a modeled connection will be deleted
- the user can release the end of a line without connecting it to another model component – in this instance, the Model Builder will establish a model builder alarm condition for the line until it is either deleted or connected to another model component

Moving shapes (stocks, flows, transformers, factories and fragments) also moves attached lines (flow lines and variable lines) and respects the right-angled or semicircular style specifications for the lines. There are several nuances to moving model components (shapes and lines) to preserve the model appearance:

- flow line edges and variable line edges on a modeled stock are initially orthogonal to each other
- flow line hemispheres and variable line hemispheres on a flow control are initially orthogonal to each other
- flow lines pass through the center of modeled flow controls
- flow lines and variable lines are distributed on an edge of a component from the center to the corners or on a hemisphere from the 90 degree angle to the 0 and 180 degree angles
- moving one or more attached lines from an edge of a model component to another component redistributes the lines on both edges affected

Several features are essential to facilitate cleanup, changes and rearrangement of the model drawing. For example, it may be necessary to reposition a stock to emphasize a reinforcement loop. The following methods are performed on the entire model or on collections of model components:

- compact the model
- expand (distribute) the model to an area
- remove overlap by rerouting flow and variable lines or repositioning the connected shapes
- cluster a group of model component on a horizontal and vertical basis for a grid arrangement
- cluster a group of model components on a circular basis for a loop arrangement
- create a fragment / destroy a fragment
- collapse to a fragment / expand from a fragment

## Defining Model Relationships

Several simple rules are necessary for a convenient and straightforward model building procedure:

- stocks, transformers and factories are placed before the flows that connect them
- a custom flow can be placed on the canvas without inflow or outflow attachments

- converters are placed before the variables that attach to them
- naming is essential for graphs, tables and reports

Once stocks and transformers have been placed and attached to flows, the user begins to define relationships with formulas. The formula for a modeled stock level is a trivial formula that is built automatically by the calculator for the user:

- $LEVEL_{t=i} = LEVEL_{t=i-1} + \sum_{t=i} inflows - \sum_{t=i} outflows$ where outflows include wastage

The formula for flow rates can be defined in either of two ways. The calculator begins to build the formula automatically. In the simplest case when only one variable is connected to the flow, the formula that the calculator automatically builds is the following:

- RATE = (variable)

The user can choose to insert a function

- RATE = $f$(variable) where $f$ is a function placeholder to be defined by the user

If more than one variable is connected to the flow, the formula that the calculator builds automatically is the following:

- RATE = $f$($variable_1$, $variable_2$…)

A method is required for the user to choose an attribute to define with a formula. A method is required for the user to build a formula from functions supported by the calculator applied to the variables that are connected. The simplest approach to build a formula is to have the user act to insert a function, select the function from a list, and then choose the associated function variables required from a list of those that are connected to the flow. This method is applicable to any instance that the user has to build a formula for any definable attribute of any model component.

## File Import / Export Functions

Several import and export methods are required for socializing models.

- import and export the model builder part in SD Calculator XML format
- import and export the model metadata part in SD Calculator XML format
- import and export the model builder file in Stella STM format
- import and export the model builder file in Vensim MDL format
- export of the model diagram in JPG, PNG and PDF formats

# THE MODEL SIMULATOR

The Model Simulator loads a model and runs the model simulation. The model simulator accepts user inputs, calculates variable and component values, detects threshold crossings and component errors, animates symbols, draws simple graphs and logs all pertinent information for the user. The logs are sufficient for the user to play back the simulation at a later time if network connectivity is lost or computing resources are insufficient to perform the simulation calculations at a later time or in the required timeframe. The model simulator can produce a movie from a simulation. The model simulator also produces finely formatted detailed graphs and reports suitable for publication on the web or in print media.

## Loading a Model to Simulate

The Model Simulator begins the user session with a prompt for the user to select and load a model to run a simulation on. The user has several options:

- select a model from a library of models stored locally in the system dynamics calculator
- select a model from local media such as a memory card, USB storage device or a SAN or NAS device attached or bonded to the system dynamics calculator
- select a model from remote libraries in model warehouses available over the Internet
- optionally build a new model from scratch with the system dynamics calculator model builder

These options may require the following actions:

- accessing the Internet, browsing for a model and downloading it into the SD Calculator memory
- verifying a credential of the source server to confirm the integrity of the source server
- verifying a checksum or a hash to confirm the integrity of the model file
- checking the model file out of a version control system and returning it revised to the control system
- encrypting or decrypting the model file

Once a model is loaded, the model simulator identifies and loads the data that is necessary to define variable values and in the model. This initial data includes the following:

- user-input values of constants in model formulas and for initial stock levels
- local datasets such as TAL files exported from personal finance software
- remote datasets from websites such as US Census Bureau or the New York Stock Exchange

To accomplish the import of data from files and websites, the simulator may be required to generate queries and parse responses with a set of rules that the user may have to define.

## Defining the Model Simulation

The Model Simulator operates each model component according to a set of characteristics that the user defined when the model was built. The following Model Simulation Base attributes are defined in the Model Simulator process for the model:

- SIM NAME STRING the simulation scenario's user assigned plain language label
- SIM NAME REG the label assigned to the simulation scenario by the Model Builder
- SIM OWNER the Model Simulator user who defined the simulation method
- SIM VERSION NUMBER the simulation base reference version number
- SIM VERSION OWNER the name of a person or organization name, a URL
- SIM VERSION DATE the published date

- SIM REVISION NUMBER an appended simulation revision number
- SIM REVISION OWNER the user who last revised the simulation revision
- SIM REVISION DATE the date of simulation revision changes
- SIM COMPLIANCE the version of the global system dynamics modeling standard followed by the Model Simulator process
- SIM SCRIPT LIST (optional) a list of simulation initialization and execution scripts
- SIM NUMBER SYSTEM the number system set for the model – typically base 10 (default)
- SIM CURRENCY SYSTEM the currency system set for the model – typically dollars (default)
- SIM UNIT SYSTEM the unit system set for the model – typically Metric (CGS default)
- SIM MODE (optional) a value of n (the number of simulations to run useful with random variables), “ACCURATE” (default) or “QUICK”
- SIM PAUSE the amount of time to pause between intervals (optional);a value in seconds of 0 (default) or a number
- TIME START the starting model time for the simulation with a value of “now” or a specified date and time or “null” (not a time-dependent model)
- TIME INTERVAL the basic time increment for each trial of the model such as a number of years, months, days, hours, minutes, seconds, or “null” (not a time-dependent model)
- TIME PERIOD the number of time intervals in a report period, or alternatively
- TRIAL PERIOD the number trials in a report period
- TIME FIRST PERIOD the number of time intervals in the first period, or alternatively
- TRIAL FIRST PERIOD the number of trials in the first period
- TIME END the ending model time for the simulation such as a specified date and time, or alternatively
- TRIAL LAST the number of trials (time intervals) simulated
- LOCK SIM a value of “NO” (default) or “YES” to prevent Model Builder changes by a user who is not the model owner, model version owner, model revision owner, simulation owner, simulation version owner, simulation revision owner or authorized by a password
- SIM DESCRIPTION an abstract of the simulation’s scope and use – a “Read Me”

The default currency system is associated with the user location.

The model simulator checks the MODEL MODE attribute to determine if simulation attributes for the model are calendar / clock-based or unitless interval trials for model calculations.

## Basic Simulator Control Strip Functions

The simulator control strip performs the following functions:
- run the simulation
- pause and resume the simulation
- stop the simulation
- build a custom graph
- build a custom table
- examine the simulation logs

# Calculating the Simulation

The model simulator performs a trial in five fundamental steps. First, the model is checked for structural model errors. Next, static calculations are performed. Static calculations are primarily performed on variable values by converters; static calculations may also be performed in modeled stocks, modeled flows and transformers. Dynamic flows are then made in calculated amounts to and from modeled stocks, transformers and fragments. Corrections are applied to stock and flow values to account for configured constraints. Finally, threshold crossings and errors are detected, summarized and reported.

### *Structural Model Errors*

The model is checked for several structural errors such as those that result in unintended permanent infinite or zero stock levels. The simulator searches for the following errors, and instances are reported to the user as errors and summarized to the symbol, fragment, layer and the model:

- an unrestricted flow connecting an unbounded stock and a modeled stock (in either direction)
- an unrestricted flow connecting an unbounded stock and a transformer (in either direction)
- unattached flow lines or variable lines
- a queued stock, custom stock, modeled flow, transformer or factory with no formula (note that the formula for a contained stock can be implicit);
- a converter with an incorrect number of operands (a sum requires at least two operands)
- an external variable from a networked source when no network connection can be established

If structural errors are detected, the model simulator will abort the simulation trial.

### *Calculations*

The following attributes are calculated by the Model Simulator process for each model component for each simulation trial except for unbounded stocks and unrestricted flows:

- VALUE — the result of the primary formula calculation at the current simulation time
  - LEVEL is the primary stock value;
  - OUTFLOW RATE is the primary flow value and transformer value;
  - RESULT is the primary converter value
  - OUTPUT is the primary factory value
- VALUE LOG — the log of all VALUE calculations for all trials of the simulation; this log is maintained if GRAPH is enabled (YES) for the model component; optionally, this log is maintained for all model components for playback and export
- THRESHOLD ALARM — threshold crossings that are detected for each threshold attribute defined for the component
- ERROR ALARM — errors that are detected for the component
- SUMMARY STATUS — the most severe status or alarm for the component, fragment, layer and model

Attributes for modeled stocks calculated by the Model Simulator process in addition to the common set:

- NET FLOW — net flow value at the current time = $\Sigma_{t=i}$ outflows - $\Sigma_{t=i}$ inflows

NET FLOW is also logged in the VALUE LOG.

Static calculations are performed in order of dependency:

- external variables are retrieved and interpreted; where external variables are retrieved as an array of values to support a number of trial times, each value is assigned to a trial time
- calculations for converters that operate on external variables and adjusted variables exclusively are processed first
- calculations for converters that operate on internal variables from stocks, flows or transformers exclusively are processed next
- calculations for converters that operate on internal variables from other converters are processed next as those other converters are processed until all converter calculations have been performed

Circular dependencies are identified, and all model components that are circularly dependent declare a CR-ERR error alarm. A circular dependency is detected when a converter is processed twice for the same trial interval and the model simulation ends prematurely.

An explicit order of precedence for static calculations is saved so that subsequent simulations for unaltered models are processed more efficiently.

### *Corrections*

If the user elects for a “Quick” model simulation, CR-INSUF errors are declared when calculated flows exceed flow capacity or stock levels become negative where these constraints have been configured. In this circumstance for “quick” simulations, no corrections are made to the simulation results. The error alarms inform the user that the simulation results are misleading, and summary indications show the user where the model misbehavior is located.

If the user elects for an accurate simulation, a number of corrections are made, and the simulation process may take significantly more time to perform as iterative corrections are performed:

- once all static calculations have been made, a correction is performed for custom flows that have a maximum capacity constraint configured that is less than the calculated flow, and these flow values are revised down to the flow capacity constraint.
- following the correction for insufficient flows, the simulator applies the calculated flow changes to the model and updates the value for each modeled stock connected to a revised flow.
- following the stock updates, a second correction is performed for custom stocks with negative values that have a zero minimum value constraint configured, and these stock values are revised to zero.

This cycle of corrections may be repeated a number of times. The modeled flow fed by a corrected modeled stock is correspondingly adjusted to account for the insufficient stock capacity, and the connected stock fed by the corrected flow is also adjusted in a corresponding manner. This correction cycle is repeated as necessary if a stock value is subsequently adjusted to negative values and the stock is constrained to non-negative values until no further corrections are necessary or no further corrections are possible.

After all corrections have been made, threshold crossings and errors are detected and the corresponding alarms are declared. At this point, the trial is complete, all necessary results are logged, and the next trial commences.

### *Thresholding*

Thresholding is a valuable tool to alert the user to conditions that impact decision-making, model simulation interpretation and simulation error analysis. The user is able to configure thresholds for any flow outflow or stock level during the model building process or at any time during the model simulation process. Custom flows, custom stocks and factories allow thresholding for other attributes as well.

Threshold crossings are logged for the model. The following log is maintained:

- THRESHOLD ALARM LOG

As many as three levels of severity can be assigned to thresholds crossing alarms according to the user's model design:

- MN (minor)
- MJ (major)
- CR (critical)

Up to six opportunities to threshold are defined:

- MN-range; VALUE ≤ MN-LOW limit, VALUE ≥ MN-HIGH limit (but not MJ or CR)
- MJ range; VALUE ≤ MJ-LOW limit, VALUE ≥ MJ-HIGH limit (but not CR)
- CR range; VALUE ≤ CR-LOW limit, VALUE ≥ CR-HIGH limit

A threshold crossing alarm is declared in each simulation interval that a data value falls in a thresholded region of values that has been defined. The threshold crossing alarm stands (remains) until a new threshold region for data values is entered or until the data value falls in a normal range of values, at which time the threshold crossing alarm is updated or cleared respectively.

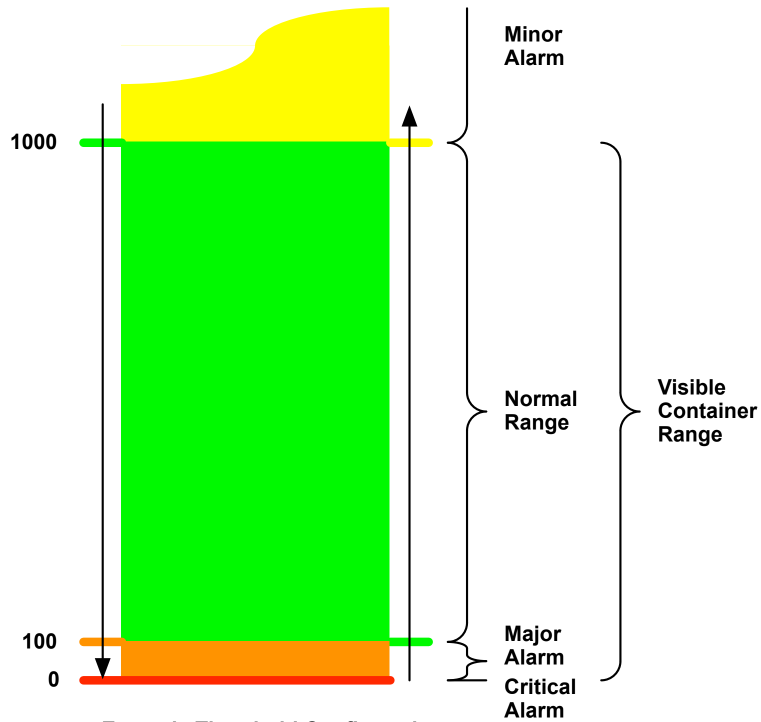
The user is not required to utilize thresholds in their model. Further, the user may choose to utilize just several thresholds (but not all). And finally, the user is free to use only the level of urgency that is logically appropriate for the model for each threshold configured.

It is important to note several rules when applying thresholds:

- the order of magnitude is CR-LOW < MJ-LOW < MN-LOW < MN-HIGH < MJ-HIGH < CR-HIGH; this relationship is mandatory; the severity of threshold crossings graduates to more urgent levels (MN to MJ to CR for example) as a flow rate or stock level rises or falls
- a flow's or a stock's symbol animation is bounded by EMPTY and FULL and may coincide with a threshold value

An example will help to illustrate the necessary flexibility for thresholds. A petty cash box has a balance on hand as much as $1000, but a balance of $100 or less must be immediately replenished. In this example, a balance more than $1000 is excessive and not correctly represented by the stock animation, and the user is alerted to this circumstance with a minor threshold crossing alarm. A balance of $100 triggers action – a major threshold crossing alarm to the user. The "down- arrow" and "up-arrow" illustrate threshold crossing declarations for the petty cash balance as it falls and rises. Note the deliberate avoidance of a minor threshold defined less that the normal range (MN-LOW), and no major or critical thresholds above the normal range (MJ-HIGH or CR-HIGH) are defined in this example.

The following drawing illustrates a complex assignment of threshold crossing values:




**Example Threshold Configuration**

| | Bound | Value |
|---|---|---|
| CR-LOW | EMPTY | 0 |
| MJ-LOW | | 100 |
| MN-LOW | | |
| MN-HIGH | FULL | 1000 |
| MJ-HIGH | | |
| CR-HIGH | | |

**Animate from EMPTY to FULL**
**Constrain Empty = YES**
**Constrain Full = NO**

## *Error Detection and Reporting*

An error detection and reporting process advises the user when the model simulator results are misleading or simply incorrect. Symbol border color reflects the severity of an error. Severity is assigned to each error condition as minor, major or critical; the default severity assignments represent the typical relative impact to meaningful simulation results. All error reports are logged for later reference by the user.

Errors are logged for the entire model. The following log is maintained:

- ERROR ALARM LOG

As many as three levels of severity can be assigned to simulation errors:

- MN (minor)
- MJ (major)
- CR (critical)

Errors are reported as “minor”, “major” or “critical”. The criterion for alarm classification is as follows:

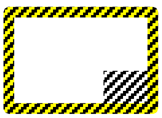
- minor errors are animation behavior that might not reflect simulation results
- major errors are simulation results that may be misleading because of the effects of model component constraints
- critical errors are model component malfunction such as a computational error

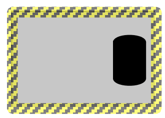
The simulation is stopped in the event of a critical error.

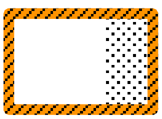
The following computational errors are detected for all model components:

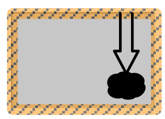
- invalid operation (*e.g.*, square root of a negative number)
- division by zero
- overflow (a result is too large to be represented correctly)
- underflow (a result is very small (outside the normal range), non-zero, and inexact)
- inexact (the resolution of the result is inadequate to represent the precision of the calsulation)

The following symbols represent stock error conditions; a pictoral addition may also be applied to represent the error condition that is relevant to the configured behavior of the stock.


(MN-NEG)


(MN-OVER)


(MJ-INSUF

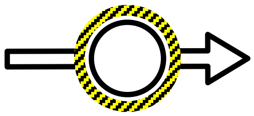

(MJ-SPILL)

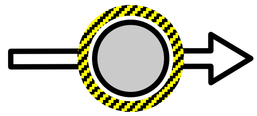
(CR-ERR)

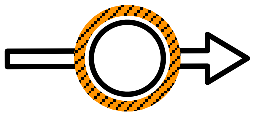
Stock error indicators include the following conditions

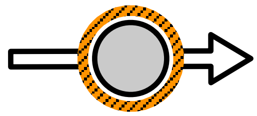
- MN-NEG — minor error (default) if a stock has a negative level (under water)<br>LEVEL < EMPTY and DISPLAY AREA is “YES”
- MN-OVER — minor error (default) if stock animation area is insufficient (full level)<br>LEVEL > FULL and DISPLAY AREA = “YES”
- MJ-INSUF — major error (default) if the stock level is insufficient (sucking air)<br>LEVEL < ZERO and CONSTRAIN ZERO = “YES”, or<br>LEVEL < CONSTRAINT MIN and CONSTRAIN MIN = “YES”
- MJ-SPILL — major error (default) if a full stock is overflowing (spilling the excess)<br>LEVEL > CONSTRAINT MAX and CONSTRAIN MAX = YES
- CR-ERR — critical error (default) if an erroneous behavior or condition is detected such as a computational error; the “bang” demands the user’s attention to the error log

The following symbols represent flow error conditions:


(MN-NEG)


(MN-OVER)


(MJ-INSUF)

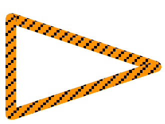

(MJ-SPILL)

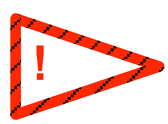

(CR ERR)

Flow error indicators include the following conditions:

- MN-NEG — minor error (default) if a stock has a negative level (under water)
  OUTFLOW < EMPTY and DISPLAY AREA is “YES”
- MN-OVER — minor error (default) if stock animation area is insufficient (full level)
  OUTFLOW > FULL FWD or OUTFLOW < FULL RVS, and
  DISPLAY AREA = “YES”
- MJ-INSUF — major error (default) if the stock level is insufficient (sucking air)
  OUTFLOW < ZERO and CONSTRAIN ZERO = “YES”, or
  OUTFLOW < CONSTRAINT MIN and CONSTRAIN MIN = “YES”
- MJ-SPILL — major error (default) if a full stock is overflowing (spilling the excess)
  OUTFLOW > CONSTRAINT MAX and CONSTRAIN MAX = YES
- CR-ERR — critical error (default) if an erroneous behavior or condition is detected such as a computational error; the “bang” demands the user’s attention to the error log

The following symbol represents factory error conditions:


(MJ-INSUF IN / OUT)


(CR-ERR)

Transformer error indicators include the following three conditions:

- MJ-INSUF IN — major error (default) if inflow capacity is insufficient to deliver the required outflow;
- MJ-INSUF OUT — major error (default) if the outflow capacity is insufficient (sucking air)
  OUTFLOW > CONSTRAINT POS and CONSTRAIN POS = “YES”
- CR-ERR — critical error (default) if an erroneous behavior or condition is detected such as a computational error; the “bang” demands the user’s attention to the error log

The following symbol represents factory error conditions:


(CR-ERR)

Factory error indicators include the following condition:

- CR-ERR — critical error (default) if an erroneous behavior or condition is detected such as a computational error; the “bang” demands the user’s attention to the error log

A factory may declare other error conditions according to its design.

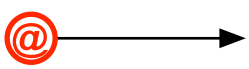
The following symbol represents converter error conditions:


(CR-ERR)

Converter error indicators include the following condition:

- CR-ERR — critical error (default) if an erroneous behavior or condition is detected such as a computational error; the “bang” demands the user’s attention to the error log

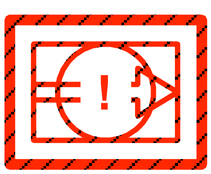
The following symbol represents variable error conditions:


(CR-ERR)

External Variable error indicators include the following condition:

- CR-ERR — critical error (default) if an erroneous behavior or condition is detected such as a computational error; “file not found” or “404 not found” are unique external variable errors; the “bang” demands the user’s attention to the error log

A fragment allows errors from the modeled constituent components to “bubble up” and be declared. The fragment symbol color reflects the summary state of the fragment’s subset of the model. The following symbol, for example, represents a critical error condition detected in the model subset that the fragment represents:


(CR-ERR)

Custom graphs and tables present the following two error conditions:

- MN-FIT — the graph or table does not fit in the print area as formatted
- MN-OOB — the data plot lies entirely outside of the bounds of the graph format, or the tabular data order of magnitude exceeds the formatted number of

digits displayed

### *Summarization*

When a threshold crossing is detected, the color of a model component's symbol border may change according to the model simulation preferences to convey the severity of the threshold crossing alarm:

- green represents a value in a normal range (NL) – no threshold crossed or a return across a threshold to the normal range of data values
- yellow represents a threshold crossing into the minor alarm range of data values (MN)
- orange represents a threshold crossing into the major alarm range of data values (MJ)
- red represents a threshold crossing into the critical range of data values (CR)

When an error is detected, the color of a model component's symbol border may change according to the model simulation preferences to convey the severity of the error alarm:

- green represents a normal status (NL) – no errors
- yellow represents minor status (MN) – minor errors
- orange represents major status (MJ)– major errors
- red represents critical status (CR) – critical errors

However, only one border color is displayed, and this color corresponds to the most severe of all threshold crossing alarms and error alarms declared for the model component or fragment. This most severe alarm represents the summary status for the model component or fragment.

The summary status is also calculated for each layer of the model according to the most severe alarm declared by the model components on that layer.

The summary status is also calculated for the entire model according to the most severe alarm declared by all the model components on all the layers of the model.

## Simulator Output

The Model Simulator provides comprehensive results to the user in several ways. The simulator presents values for model component levels for model components. The simulator presents threshold and error alarm summary status for model components, fragments, layers and for the model itself. Each component or fragment provides information on mousing over the symbol. The model stocks, flows and transformers are animated to illustrate changes in level and outflow rate. The simulator generates graphs and tables for levels, inflows and outflows for modeled stocks, modeled flows, transformers and factories; the graphs and tables can be simple and compact or they can expansive and custom formatted. The user is able to keep the output sparse and simple, or create rich information displays and finely polished reports through easily configured model simulator options.

### *Indicators*

Indicators provide a quantitative view of the model's simulation to the user.

Indicators are displayed to the user for each simulation according to the indicator preference represented by the DISPLAY VALUE attribute for each model component. Continuously displayed indicators are limited to the model component's value. A "mouse over" provides a comprehensive pop-up indicator fro the user when additional information is desired.

Indicators for modeled stocks displayed to the user by the Model Simulator process:

- DISPLAY VALUE — display the LEVEL; a value of "NO" or "YES" (default)
- MOUSE OVER — LEVEL, NET FLOW, THRESHOLD ALARM, ERROR ALARM

Indicators for modeled flows displayed to the user by the Model Simulator process:

- DISPLAY VALUE — display the OUTFLOW RATE; a value of "NO" (default) or "YES"
- MOUSE OVER — OUTFLOW RATE, THRESHOLD ALARM, ERROR ALARM

Indicators for converters displayed to the user by the Model Simulator process:

- DISPLAY VALUE — display the RESULT; a value of "NO" (default) or "YES"
- MOUSE OVER — RESULT, ERROR ALARM

Indicators for transformers displayed to the user by the Model Simulator process:

- DISPLAY VALUE — display the OUTFLOW RATE; a value of "NO" (default) or "YES"
- MOUSE OVER — OUTFLOW RATE, THRESHOLD ALARM, ERROR ALARM

Indicators for factories displayed to the user by the Model Simulator process:

- DISPLAY VALUE — display the OUTPUT; a value of "NO" or "YES" (default)
- MOUSE OVER — OUTPUT, THRESHOLD ALARMS, ERROR ALARMS

Indicators for fragments displayed to the user by the Model Simulator process:

- MOUSE OVER — nothing is displayed

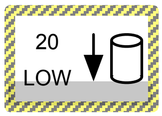
### *Animation*

Animation provides a qualitative view of the model's simulation to the user.

The user is able to configure stock animation for any stock during the model building process or at any time during the model simulation process. Stock animation displayed to the user by the Model Simulator process is configured with the following attributes:

- DISPLAY SUMMARY — display the most severe threshold crossing or error alarm severity for the component; a value of "NO" or "YES" (default) fragment, layer and model
- DISPLAY THRESH ALARM — display threshold alarm text; a value of "NO" (default) or "YES"
- DISPLAY ERROR ALARM — display error alarm text; a value of "NO" (default) or "YES"
- DISPLAY GRAPHTABLE — display the simplest possible graph and table of LEVEL; a value of "NO" (default) or "YES"
- DISPLAY AREA — display a calculated shaded region of the stock rectangle that represents the value; a value of "NO" (default) or "YES"; if FULL in the animation profile is "null", then AREA animation is automatically disabled
- FULL — the maximum value represented by the area animation

An example of stock animation is the following:



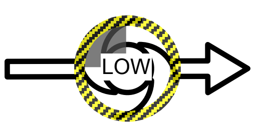

This example modeled, containerized stock symbol is animated. The stock's value is currently 20 (EMPTY = 0 and FULL = 100 in this example)) and falling. A corresponding volume of the stock is shaded and a down-arrow is displayed. The stock status is a minor alarm because the low threshold was crossed from the direction of the normal unthresholded range – the word "LOW" and a yellow border are displayed.

Some allowance must be made for the display of animation features in general if the processing resources or display resources are inadequate. In this case, the user should elect for "simple" animation that displays only the AREA animation feature. If "simple" animation has been elected, value and threshold alarm information is available on demand when the user "mouses over" the stock symbol. If changes of the calculated value are so rapid that VALUE is just a blur of pixels, VALUE should be updated every second and displayed when the model simulation is paused.

Flow animation is an optional Model Simulator feature. The user is able to configure flow animation for any flow outflow during the model building process or at any time during the model simulation process. Flow animation displayed to the user by the model simulator process is configured with the following attributes:

- DISPLAY SUMMARY — display the most severe threshold crossing or error alarm severity for the component; a value of "NO" or "YES" (default) fragment, layer and model
- DISPLAY THRESH ALARM — display threshold alarm text; a value of "NO" (default) or "YES"
- DISPLAY ERROR ALARM — display error alarm text; a value of "NO" (default) or "YES"
- DISPLAY GRAPHTABLE — display the simplest possible graph and table of OUTFLOW; a value of "NO" (default) or "YES"
- DISPLAY AREA — display a calculated shaded region of the flow circle that represents the value; a value of "NO" (default) or "YES"; if FULL in the animation profile is "null", then AREA animation is automatically disabled
- FULL FWD — the maximum value represented by the area animation for forward flow rates
- FULL REV — the maximum value represented by the area animation for reverse flow rates (default = 0 for a unidirectional flow)

An example of flow animation is the following:




This modeled flow symbol is animated. The flow is currently approximately 25% of capacity (25% full). The flow status is a minor alarm because the low level threshold was crossed from the direction of the normal unthresholded range of values – yellow is displayed.

Transforner animation is an optional Model Simulator feature. The user is able to configure transformer animation for any transformer during the model building process or at any time during the model simulation process. Transformer animation displayed to the user by the model simulator process is configured with the following attributes:

- DISPLAY SUMMARY — display the most severe threshold crossing or error alarm severity for the component; a value of "NO" or "YES" (default) fragment, layer and model
- DISPLAY THRESH ALARM — display threshold alarm text; a value of "NO" (default) or "YES"
- DISPLAY ERROR ALARM — display error alarm text; a value of "NO" (default) or "YES"
- DISPLAY GRAPHTABLE — display the simplest possible graph and table of OUTFLOW; a value of "NO" (default) or "YES"

### *Graphs and Tables*

Graphs are an optional Model Simulator feature. Graphs provide insight into the behavior of a model during simulation by providing a visual record of model behavior over time. The user is able to configure graphs for any flow outflow or stock level during the model building process or at any time during the model simulation process.

Graph types displayed to the user by the Model Simulator process include the following:

- SIMPLE (GRAPHTABLE) — a simple graph of a stock level or an outflow rate drawn over the course of a simulation that is not much larger than the model symbol; a simple graph is drawn on a pair of orthogonal linear axes without gradients, units, labels or customization features of any kind; nonzero minimum and maximum values are noted on a simple graph
- CUSTOM (GRAPH) — a custom graph is a large graph image drawn in its own window; a custom graph can be drawn completely with default formats, but it is entirely customizable by the user

Customizable graphs have the following attributes:

- graph label, font, style and color
- graph border line style, weight and color
- graph background color
- field line style, weight and color
- field background color
- horizontal axis scale
- horizontal axis line style, weight and color
- horizontal axis label, font, style and color
- horizontal axis units
- horizontal axis major divisions
- horizontal axis minor divisions
- horizontal second axis
- vertical axis scale
- vertical axis line style, weight and color
- vertical axis label, font, style and color
- vertical axis units
- vertical axis major divisions
- vertical axis minor divisions
- vertical second axis
- plot legend, label, font, style and color
- plot line style, weight, color and data point mark for each plot
- print header block label, font, style and color
- print footer block label, font, style and color
- data sets

A SIMPLE graph can be converted to a CUSTOM graph on demand.

Tables are an optional Model Simulator feature. Tables also provide insight into the behavior of a model during simulation by providing a tabular record of model behavior over time. The user is able to configure tables for any flow outflow or stock level during the model building process or at any time during the model simulation process. Custom flows, custom stocks and factories allow tables for other attributes as well.

Table types displayed to the user by the Model Simulator process include the following:

- SIMPLE (GRAPHTABLE) a simple compact table of a stock's level or a flow's or transformer's outflow value over the course of a simulation; a simple table is displayed in a scrollable window several columns wide without customization features of any kind;
- CUSTOM (TABLE) a custom table is a comprehensive table displayed in its own window; a custom table can be displayed completely with default formats, but it is entirely customizable by the user

Customizable tables have the following attributes:

- table label, font, style and color
- table border line style, weight and color
- table background color
- field line style, weight and color
- field background color
- horizontal legend line style, weight and color
- horizontal legend label, font, style and color
- vertical legend line style, weight and color
- vertical legend label, font, style and color
- print header block label, font, style and color
- print footer block label, font, style and color
- data sets

Simple graphs and tables are displayed in a manner that consumes a minimum area on the calculator display. The following feature is required to navigate a collection of simple graphs and tables that are produced by a simulation:

- a list of available simple graphs and tables are displayed on demand for the user to choose from, and the user should be able to choose one to view with a click or gesture, or more than one to view with the shift key or a gesture

A SIMPLE table can be converted to a CUSTOM table on demand.

## *Pictures and Movies*

The model simulator is also optionally able to create pictures and movies of the simulation results. The picture of a model simulation captures the animation at a point in time, and a simulation movie captures the animation over a period of time.

Pictures have the following attributes:

- picture label, font, style and color
- picture border line style, weight and color
- picture background color

- picture background image
- print header block label, font, style and color
- print footer block label, font, style and color
- trial time or trial number

Movies have the following attributes:

- movie label, font, style and color
- movie border line style, weight and color
- movie background color
- header block label, font, style and color
- footer block label, font, style and color
- beginning and either ending time or number of trials
- export file format such as MP4, MOV or AVI

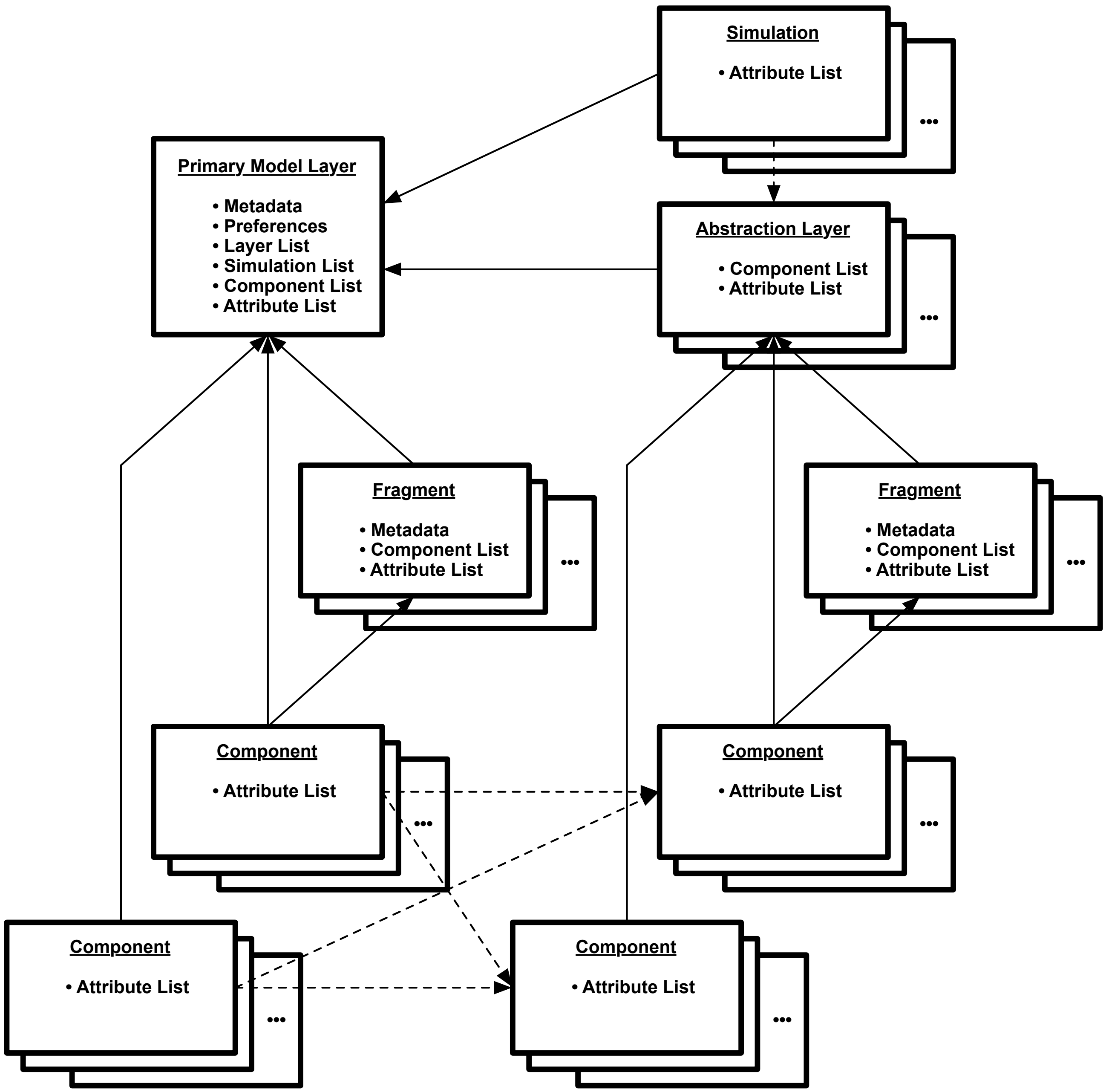
### *File Import / Export Features*

Several import / export features are required for socializing simulation results.

- import and export the model simulator part in SD Calculator XML format
- export the custom graph in JPG, PNG and PDF formats
- export the custom table in TXT, RTF and PDF
- export pictures in JPG, PNG and PDF formats
- export movies in FLV, MOV and MP4 (M4V) formats

# THE MODEL IN DEPTH

The bulk of the model file is defined with the Model Builder process. The primary focus of this section is on the variety of components that can be included in the model itself by working with the model builder process. Significant detail is provided for each different model component that the user can use. The concept of layer abstraction is also addressed. The following diagram illustrates the relationship between the various elements of the System Dynamics Calculator model and the model's data structure:




In this drawing, it is important to note several details. All model elements have a relationship with the primary model layer either directly or indirectly. A fragment has many of the same characteristics as the primary model layer such as extensive metadata. Components that belong to a fragment also belong to the model layer. Fragments are treated like components; fragments may contain fragments, and any layer may contain fragments.

## The Model

The model has the following basic file structure:

- METEDATA — the minimum metadata to accomplish ownership, version control, security, traceability, and searchability
- PREFERENCES — a set of settings that affects appearance, abstraction and other model-specific qualities
- LAYER LIST — the primary model registered name and a list of layer registered names
- SIMULATION LIST — the list of named simulation scenarios
- COMPONENT LIST — the unique set of registered names of model components and fragments contained in the model

The model has the following minimum metadata:

- MODEL NAME STRING — the plain language name of the model
- MODEL NAME REG — the label assigned to the model by the Model Builder
- MODEL MODE — a value of “INTERVAL” for a unitless model or “DATETIME” for a calendar / clock-based model
- MODEL OWNER — the Model Builder user who defined the model
- MODEL VERSION NUMBER — the model base reference version number
- MODEL VERSION OWNER — the modeler information such as organization name and model URL
- MODEL VERSION DATE — the published date
- MODEL REVISION NUMBER — an appended model revision number
- MODEL REVISION OWNER — the user who last revised the model revision
- MODEL REVISION DATE — the date of model version changes
- MODEL COMPLIANCE — the version of the global system dynamics modeling standard followed by the Model Builder process used to build the model
- LOCK MODEL — a value of “NO” (default) or “YES” to prevent Model Builder changes to any element of the model by a user who is not the model owner, model version owner, model revision owner or authorized by a password
- LOCK PRIMARY LAYER — a value of “NO” (default) or “YES” to prevent Model Builder changes to any element of the model by a user who is not the model owner, model version owner, model revision owner or authorized by a password
- MODEL DESCRIPTION — an abstract of the model’s scope and use – a “Read Me”

The model’s metadata shall be compliant with StratML, an XML-like mark-up language that is searchable and allows the model to be registered in StratML-compliant model “warehouses”.

The following Model Definition Base attributes are defined in the Model Builder process for each model component or fragment in the model:

- COMP NAME STRING — the component’s or fragment’s user assigned plain language label
- COMP NAME REG — the label assigned to the component or fragment by the Model Builder
- COMP TYPE — the component type or subtype or fragment
- COMP LOCATION — the coordinates or “placement” of the component or fragment in the model image
- LABEL COMP — label the component with the NAME STRING; a value of “NO” or “YES” (default)
- COMP INTERFACE LIST — the names of the interfaces or “ports” that this model component is

| | |
|---|---|
| | equipped with; the standard names can be overridden by the user with a new user-assigned plain language COMP INTERFACE NAME STRING |
| • LABEL COMP INTERFACES | label each component interface with the COMP INTERFACE NAME STRING; a value of "NO" (default) or "YES" |
| • COMP CONNECTION LIST | the mapping from each COMP INTERFACE NAME STRING to COMP NAME REG / COMP INTERFACE NAME STRING on another model component in the model |
| • NOTE | simple text generated by the user and placed anywhere to explain things |
| • NOTE LOCATION | the coordinates or "placement" of the note in the model image |

Connections between model components are explicitly known from the defined formulas for each model component.

The following sections describe the unique characteristics of each component that the model may contain.

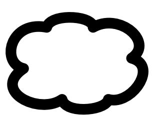
## Stocks

Stocks store a good in the model. Goods flow into a stock and out of a stock.

### *Unbounded Stocks*

An Unbounded Stock stores a tangible or intangible good in the model without any limitation to the volume of the good in the stock or the rate of flow of the good in or out of the stock. An unbounded stock typically sits on the model's boundary with the "rest of the world".

The unbounded stock symbol is a stylized "cloud" or "puddle". The following is the conventional unbounded stock symbol:


Unbounded Stock

Attributes configured by the Model Builder user for any unbounded stock:

| | |
|---|---|
| • GOOD | the good that the stock holds |

Component Interfaces on an unbounded stock allowed by the Model Builder process:

- from or to any type of flow

Error Conditions detected and prohibited by the Model Builder process:

- none

Other Behaviors and Characteristics:

- an unbounded stock implicitly accepts "spillage" from custom stocks and leakage from custom flows without an explicit connection to the custom stock or flow

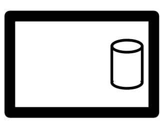
### *Modeled Stocks*

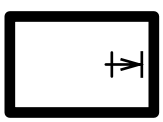
A Modeled Stock stores a tangible or intangible good in the model. A modeled stock is supplied the good over time by one or more inflows according to the type of modeled stock and optionally delivers the good with one or more outflows according to the type of modeled stock. The quantity of the good in the stock is continuously monitored and available to any converter or modeled process as a “variable” value. All the inflows and outflows connected to a modeled stock deliver the same good – a modeled stock stores only one good.

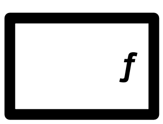
Three modeled stock types are defined for the Model Builder process:

- Containerized Modeled Stock is a simple stock type that accumulates a good from one or more inflows and delivers the good to zero or one outflow
- Queued Modeled Stock is a simple stock type that accumulates a good from one inflow and delivers the same quantity of good to one outflow after a configurable time delay
- Custom Modeled Stock is a complex stock type that incorporates the combined characteristics of both containerized and queued stocks, and adds other refinements to their models such as negative levels, level overflow, LIFO queues, level erosion, multiple outflows and thresholding in addition to the level for any inflow or outflow that the user configures

The modeled stock symbol is a stylized rectangle. The following are modeled stock symbols:


Containerized Modeled Stock


Queued Modeled Stock



Custom Modeled Stock

Attributes configured by the Model Builder user for any modeled stock:

| | |
|---|---|
| • GOOD | the good that the stock holds (optional) |
| • UNIT | the unit of measure for the good (optional) |
| • ANIMATE | a value of “NO” or “YES” (default) |
| • GRAPH | display a high quality formatted graph of LEVEL; a value of “NO” (default) or “YES” |
| • TABLE | display a high quality formatted table of LEVEL; a value of “NO” (default) or “YES” |
| • ABSTRACT | publish LEVEL to abstraction layers; a value of “NO” (default) or “YES” |
| • COST | publish COST attributes to abstraction layers; a value of “NO” (default) or “YES” |

Attributes optionally configured by the Model Builder user for any modeled stock to describe the corresponding Cost Abstraction Layer basis:

| | |
|---|---|
| • COST ACCUM PERIODIC` | accumulate cost for the period described in the simulation specification; a value of “NO” (default) or “YES” |

| | |
|---|---|
| • COST FIRST | a one-time “first cost” |
| • COST TIME | a cost per time interval |
| • COST GOOD | a cost per unit of good stored |

Attributes configured by the Model Builder user for a containerized modeled stock:

| | |
|---|---|
| • LEVEL INITIAL | the stock’s initial level value |
| • LEVEL THRESHOLDS | “NONE” (default) or values for MN range, MJ range, CR range |

Attributes configured by the Model Builder user for a queued modeled stock:

| | |
|---|---|
| • DELAY INITIAL | the stock’s initial delay |
| • DELAY FORMULA | the formula to calculate the stock’s delay value |
| • DELAY THRESHOLDS | “NONE” (default) or values for MN range, MJ range, CR range |

The attributes of the custom modeled stock are a “superset” of attributes including those of the containerized modeled stock and the queued modeled stock defined above. In addition, the following attributes are configured by the Model Builder user for a custom modeled stock:

| | |
|---|---|
| • STALE | a value of “NO” (default) or “YES” to age the value of the stock’s good |
| • STALE INITIAL | the stock’s initial level age |
| • STALE THRESHOLDS | “NONE” (default) or values for MN range, MJ range, CR range |
| • WASTE | a value of “NO” (default) or “YES – BEGIN” to erode the level of the stock’s level at the beginning of the interval or “YES –END” to erode the stock’s level at the end of the interval |
| • WASTE FORMULA | the formula to calculate the erosion of the stock’s level |
| • WASTE THRESHOLDS | “NONE” (default) or values for MN range, MJ range, CR range |
| • NETFLOW THRESHOLDS | “NONE” (default) or values for MN range, MJ range, CR range |
| • CONSTRAIN MAX (default) | stock LEVEL > CONSTRAINT MAX not allowed; a value of “NO” or “YES” |
| • CONSTRAINT MAX | the maximum allowed positive stock LEVEL value; mandatory if CONSTRAIN MAX = “YES” |
| • CONSTRAIN ZERO | stock LEVEL < 0 not allowed; a value of “NO” or “YES” (default) |
| • CONSTRAIN MIN | stock LEVEL < CONSTRAINT MIN not allowed; a value of “NO” (default) or “YES”; |
| • CONSTRAINT MIN | the minimum allowed stock LEVEL value; mandatory if CONSTRAIN MIN = “YES” |
| • OUTFLOW RATION | the rule to provide a good to multiple outflows when the level is insufficient; a value of “SMALLEST-FIRST” or “LARGEST-FIRST” or “PROPORTIONAL” (default) |
| • QUEUE ORDER | FIFO (default) or LIFO |

Component Interfaces on a modeled stock allowed by the Model Builder process:

- from or to any type of modeled flow
- from or to a factory according to its unique rules
- from or to a model component as an internal variable value
- from an external variable or an adjusted variable

Error Conditions detected and prohibited by the Model Builder process:

- none

Other Behaviors and Characteristics:

- a containerized stock formula is trivial and built automatically as the user connects flows to the stock
- a queued stock is strictly FIFO by definition
- a queued stock is interchangeable with a delayed flow in the model

- a modeled stock may be added to the model by “dropping” it onto any flow line to bisect the flow
- the GOOD and UNITS of a stock are implicit if the stock is connected through a flow to another stock where GOOD and UNITS have been previously defined

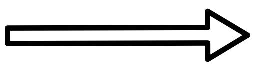
## Flows

Flows transport goods in the model. Goods are delivered between stocks in a flow. Goods are delivered to a transformer in a flow.

### *Unrestricted Flows*

An unrestricted flow delivers a tangible or intangible good in unlimited quantities at unlimited rates of flow.

The Unrestricted Flow’s symbol is a stylized “hollow arrow”. The Unrestricted flow symbol may optionally be drawn with enforced 90° transitions or as a semicircle. The following is an unrestricted flow symbol:


Unrestricted Flow

Attributes configured by the Model Builder user for any unrestricted flow:

- none

Unrestricted flows typically connect from a transformer or from a production factory to another model component. The following component Interfaces on an unrestricted flow are allowed by the Model Builder process:

- from or to any type of stock
- from or to any type of transformer
- from or a factory according to the unique rules of the factory

Error Conditions detected and prohibited by the Model Builder process:

- ILLEGAL UNRESTRICTED FLOW CONNECTION
  – between an unbounded stock and any type of stock
  – between any two stocks unless one (or both) are queued stocks
  – from an unbounded stock to any type of transformer
  – from a factory or to a factory that violates its unique rules
- INCONSISTENT GOOD OR UNITS
  – an attempt to connect from and to stocks, transformers and factories when GOOD or UNIT has already been defined on both sides of a flow and they are inconsistent

Other Behaviors and Characteristics:

- none

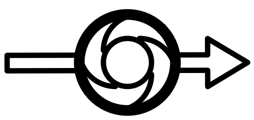
### *Modeled Flows*

A Modeled Flow is a homogeneous flow of a tangible or intangible good in the model with a control to manage the flow rate. A modeled flow incorporates the computational elements of a converter.

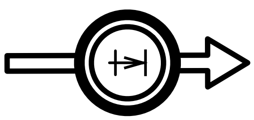
Three modeled flow types are defined for the Model Builder process:

- Adjusted Flow is a simple flow type that flows a calculated quantity of a good from the inflow to the outflow at every time increment
- Delayed Flow is a simple flow type that flows a good from the inflow to the outflow after a configurable time delay
- Custom Flow is a complex flow type that incorporates the combined characteristics of both adjusted and delayed flows, and adds other refinements to their models such as flow leakage and flow direction reversal

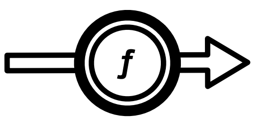
The modeled flow's symbol is a stylized pair of concentric circles overlaid upon the unrestricted flow symbol. The arrow part of the modeled flow symbol may optionally be drawn with enforced 90° transitions or as a semicircle. The following are modeled flow symbols:



Adjusted Flow



Delayed Flow



Custom Flow

Attributes configured by the Model Builder user for any modeled flow:

- ANIMATE — a value of "NO" (default) or "YES"
- GRAPH — display a high quality formatted graph of OUTFLOW; a value of "NO" (default) or "YES"
- TABLE — display a high quality formatted table of OUTFLOW; a value of "NO" (default) or "YES"
- ABSTRACT — publish OUTFLOW to abstraction layers; a value of "NO" (default) or "YES"
- COST — publish COST attributes to abstraction layers; a value of "NO" (default) or "YES"

Attributes optionally configured by the Model Builder user for any modeled flow to describe the corresponding Cost Abstraction Layer basis:

- COST ACCUM PERIODIC` — accumulate cost for the period described in the simulation specification; a value of "NO" (default) or "YES"
- COST FIRST — a one-time "first cost"
- COST TIME — a cost per time interval
- COST GOOD — a cost per unit of good outflowed

Attributes configured by the Model Builder user for an adjusted modeled flow:

- OUTFLOW FORMULA — the formula to calculate the flow's outflow rate
- OUTFLOW THRESHOLDS — "NONE" (default) or values for MN range, MJ range, CR range

Attributes configured by the Model Builder user for an delayed modeled flow:

- DELAY INITIAL — the outflow's initial delay
- DELAY FORMULA — the formula to calculate the flow's delay value
- DELAY THRESHOLDS — "NONE" (default) or values for MN range, MJ range, CR range

The attributes of the custom modeled flow are a "superset" of attributes including those of the adjusted modeled flow and the delayed modeled flow defined above. In addition, the following attributes are configured by the Model Builder user for a custom modeled flow:

- WASTE — a value of "NO" (default) or "YES-IN" to leak a portion of the inflow or "YES-OUT" to leak a portion of the outflow
- WASTE FORMULA — the formula to calculate the leakage of the flow's outflow
- WASTE THRESHOLDS — "NONE" (default) or values for MN range, MJ range, CR range
- CONSTRAIN MAX — flow OUTFLOW > CONSTRAINT MAX not allowed; a value of "NO" (default) or "YES"
- CONSTRAINT MAX — the maximum allowed flow OUTFLOW value; mandatory if CONSTRAIN MAX = "YES"
- CONSTRAIN ZERO — flow OUTFLOW < 0 not allowed; a value of "NO" or "YES" (default)
- CONSTRAIN MIN — flow OUTFLOW < CONSTRAINT MIN not allowed; a value of "NO" (default) or "YES"
- CONSTRAINT MIN — the minimum allowed flow OUTFLOW value; mandatory if CONSTRAIN MIN = "YES"
- OUTFLOW INCREMENT — the incremental number of units outflowed at a time; a value of "NONE" or a specific numeric value

Component Interfaces on a modeled flow allowed by the Model Builder process:

- from or to any type of stock
- from or to any type of transformer
- from or to a factory according to its unique rules
- from or to a model component as an internal variable value
- from an external variable or an adjusted variable

Error Conditions detected and prohibited by the Model Builder process:

- INCONSISTENT GOOD OR UNITS – an attempt to connect from and to stocks, transformers and factories when GOOD or UNIT has already been defined on both sides of a flow and they are inconsistent

Other Behaviors and Characteristics:

- a delayed flow is strictly FIFO by definition
- a delayed flow is interchangeable with a queued stock in the model
- the GOOD and UNIT of a flow are implicit if the flow is connected between stocks where GOOD and UNIT have been previously defined
- the flow arrow shows the direction of the flow; the arrow for a custom flow must change directions when the value of the flow rate is negative and CONSTRAIN LOWER is "NO"

## Transformers

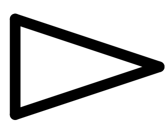
A transformer is a new basic system dynamics function that eliminates a significant source of modeling errors and simplifies many models for the user. Transformers perform a "transformation" function to change one or more different goods in the model into a new good. The outflow is related to one or more dissimilar inflows by a simple "exchange rate" relationship or a ratio.

A Transformer is analogous to a "macro". A transformer can be modeled from a collection of conventional stocks, flows, converters and variables. Transformers present the appearance of a stock to an inflow and behave like a flow to deliver the resulting good to stocks, other transformers or factories in the model.

Four transformer types are defined for the Model Builder process:

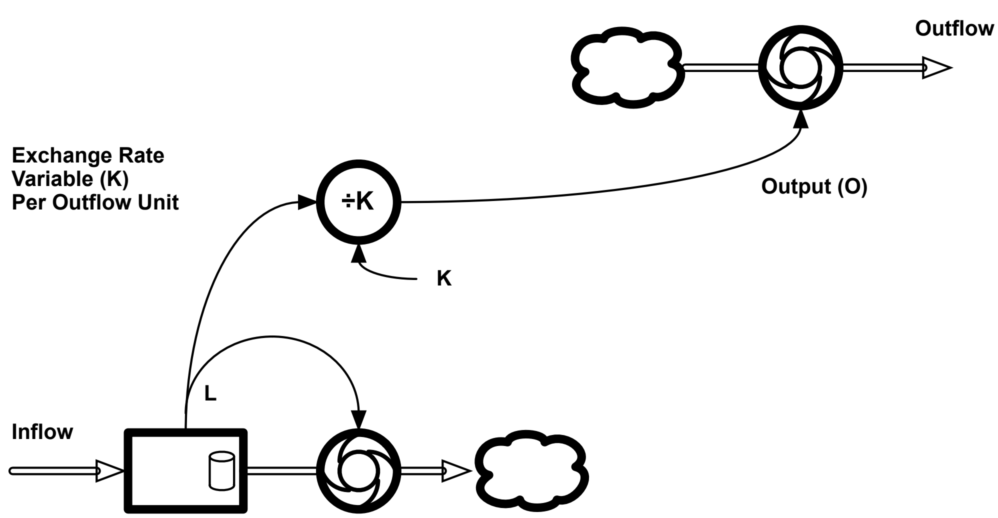
- Unconstrained Transformer changes all of an incoming good sent from a modeled flow into a different good according to a configured "exchange rate".
- Adjusted Transformer changes a calculated amount of an incoming good sent from a modeled flow into a specified amount of different good according to a configured "exchange rate".
- Delayed Transformer changes all of an incoming good sent from a modeled flow into a different good according to a configured "exchange rate". The outflow is delayed for a configured period of time.
- Custom Transformer incorporates the combined characteristics of both adjusted and delayed transformers, and adds refinements to their models such as multiple inflows, transformer wastage, inflow accumulation thresholding for any inflow that the user configures, and outflow constraint.

The transformer symbol is a stylized right-pointing isosceles triangle. The following is an unconstrained transformer symbol:

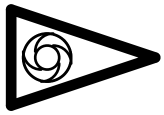


Unconstrained (All Available) Transformer

An unconstrained transformer may be modeled with conventional system dynamics functions in the following manner:

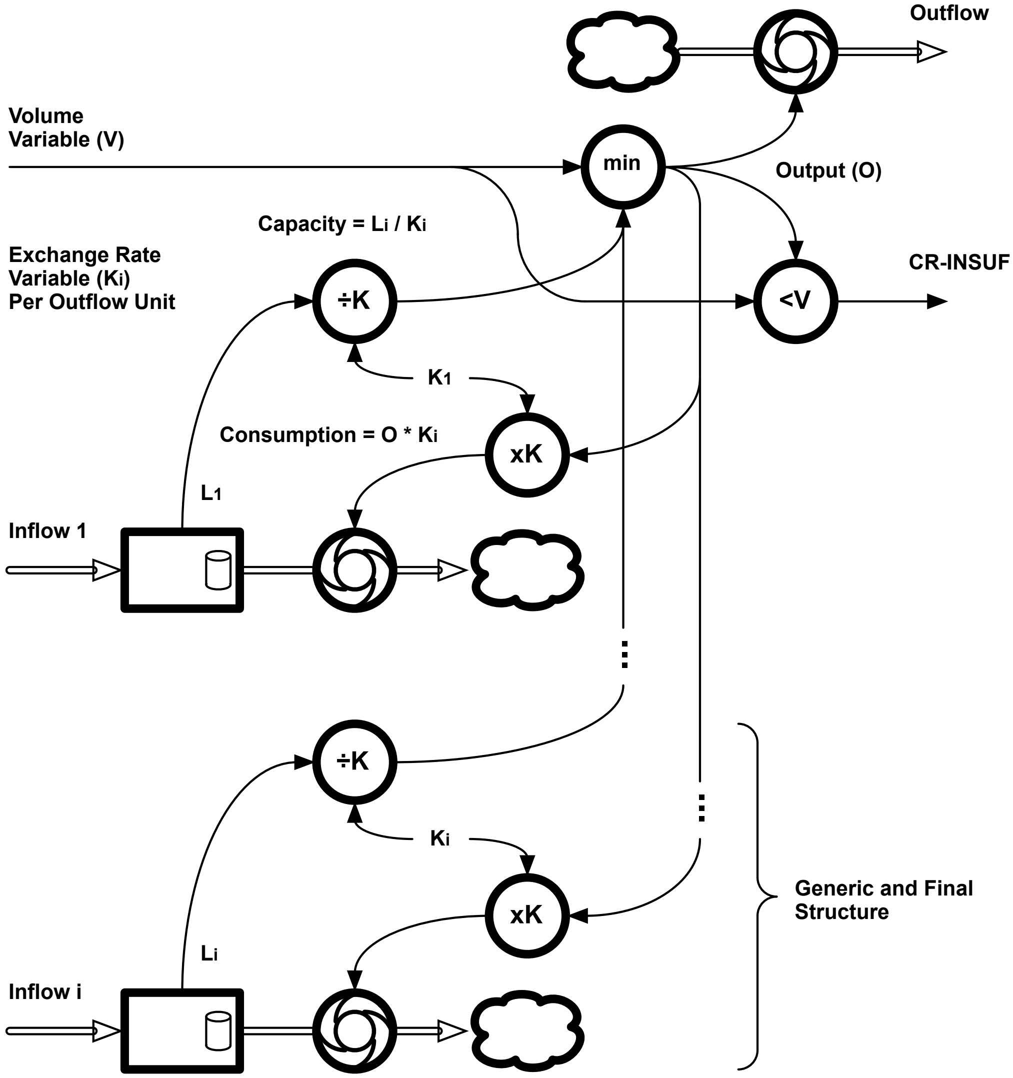


The following is an adjusted transformer symbol:

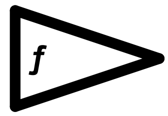


Adjusted Transformer

An adjusted transformer may be modeled in the following manner:



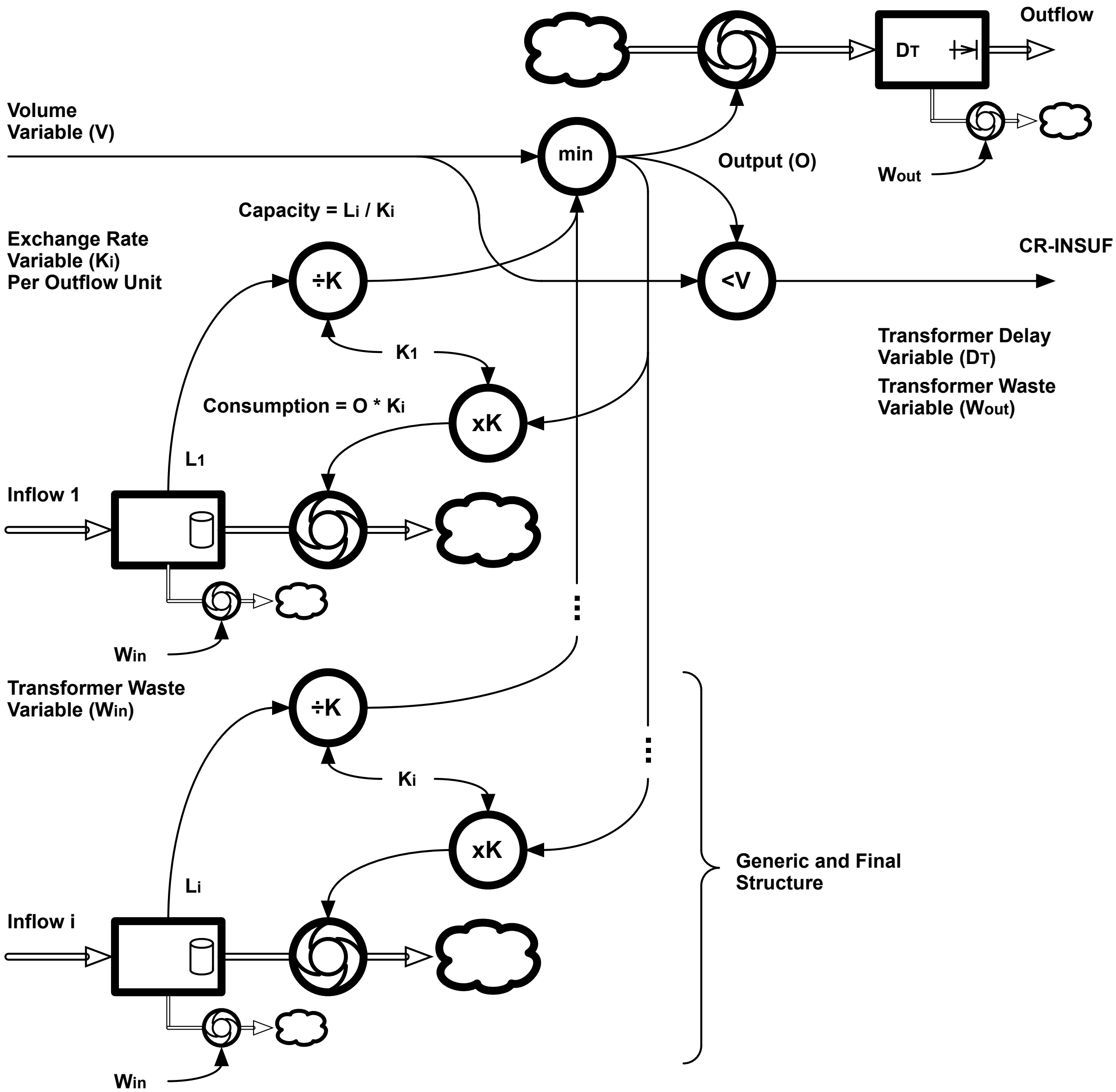
The following is a Custom Transformer symbol:




Customized Transformer

A customized transformer is an expandable model component with an core structure, and a generic structure. Expanding a customized transformer requires an additional generic structure in the underlying model of the transformer.

A customized transformer may be modeled in the following manner:




The customized transformer model above does not reflect several elements that may be added in the future. In particular, transformers may need to consume incoming flows or produce outgoing flows in integer increments. While the outflow case is reasonable handled as long as inflow capacity is sufficient, integer outflows less than the required outflow is not accommodated.

Attributes configured by the Model Builder user for any transformer:

| | |
|---|---|
| • GOOD | the good for each transformer inflow and outflow |
| • UNIT | the unit of measure for the good in each transformer inflow and outflow |
| • ANIMATE | a value of "NO" (default) or "YES" |
| • GRAPH | display a high quality formatted graph of OUTFLOW; a value of "NO" (default) or "YES" |
| • TABLE | display a high quality formatted table of OUTFLOW; a value of "NO" (default) or "YES" |
| • ABSTRACT | publish OUTFLOW to abstraction layers; a value of "NO" (default) or "YES" |
| • COST | publish COST attributes to abstraction layers; a value of "NO" (default) or "YES" |

Attributes optionally configured by the Model Builder user for any transformer to describe the corresponding Cost Abstraction Layer basis:

| | |
|---|---|
| • COST ACCUM PERIODIC` | accumulate cost for the period described in the simulation specification; a value of "NO" (default) or "YES" |
| • COST FIRST | a one-time "first cost" assessed only in the first interval |
| • COST TIME | a cost per time interval |
| • COST GOOD | a cost per unit of good outflowed |

Attributes configured by the Model Builder user for an adjusted modeled transformer:

| | |
|---|---|
| • OUTFLOW FORMULA | the formula to calculate the transformer's target outflow rate "V" from the inflow exchange rate constants "$k_i$" |
| • OUTFLOW THRESHOLDS | "NONE" (default) or values for MN range, MJ range, CR range |

The attributes of the customized transformer are a "superset" of attributes including those of the adjusted transformer and the delayed transformer defined above. In addition, the following attributes are configured by the Model Builder user for a customized transformer:

| | |
|---|---|
| • DELAY INITIAL | the transformer outflow's initial delay |
| • DELAY FORMULA | the formula to calculate the transformer outflow's delay value |
| • DELAY THRESHOLDS | "NONE" (default) or values for MN range, MJ range, CR range |
| • WASTE | a value of "NO" (default) or "YES-IN" to leak a portion of the inflows or "YES-OUT" to leak a portion of the outflow |
| • WASTE FORMULA | the formula to calculate the leakage of the outflow |
| • WASTE THRESHOLDS | "NONE" (default) or values for MN range, MJ range, CR range |
| • LEVEL THRESHOLDS | "NONE" (default) or values for MN range, MJ range, CR range for each inflow stock |
| • CONSTRAIN POS | transformer OUTFLOW > CONSTRAINT not allowed; a value of "NO" (default) or "YES" |
| • CONSTRAINT POS | maximum transformer OUTFLOW value |
| • OUTFLOW INCREMENT | the incremental number of units outflowed at a time; a value of "NONE" or a specific numeric value |

Component Interfaces on a transformer allowed by the Model Builder process:

- to any type of stock
- from any type of modeled flow
- to any type of transformer
- from or to a factory according to its unique rules

- from or to a model component as an internal variable value
- from an external variable or an adjusted variable

Error Conditions detected and prohibited by the Model Builder process:

- INCONSISTENT GOOD OR UNITS – an attempt to connect to stocks, transformers and factories when GOOD or UNIT has already been defined on the other side of the outflow and they are inconsistent

Other Behaviors and Characteristics:

- a delayed modeled transformer is strictly FIFO by definition
- the GOOD and UNITS of a transformer are implicit if the transformer is connected through a flow to a stock where GOOD and UNITS have been previously defined

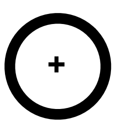
## Converters

A Converter is a computational model element that immediately processes constants, externally provided data, internal model values such as stock levels and flow rates, and other model variables in a user-defined formula. The result of a converter process is a floating point or an alphanumeric string.

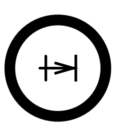
Three converters types are defined for the Model Builder process:

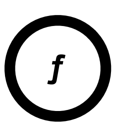
- Single Function Converter is a simple floating point converter type that performs a single mathematical operation on a set of input variables presented to it
- Delayed Converter is a converter type that delays the value of a variable for a time value or number of simulation intervals
- Customized Converter is a complex converter type that computes any configured multifunction calculation on any number of input variables and delays the result if configured to

The converter symbol is a stylized circle. The following are converter symbols:



Simple Converter (summed converter illustrated)


Delayed Converter



Customized Converter

Attributes configured by the Model Builder user for any converter:

- GRAPHTABLE display the simplest possible graph and table of RESULT; a value of "NO" (default) or "YES"
- GRAPH display a high quality formatted graph of RESULT; a value of "NO" (default) or "YES"
- TABLE display a high quality formatted table of RESULT; a value of "NO"

(default) or “YES”

- ABSTRACT — publish RESULT to abstraction layers; a value of “NO” (default) or “YES”

Attributes configured by the Model Builder user for a simple converter:

- FORMULA — the formula to calculate the converter’s result
- SHOW VARIABLES — show implicit user input variable and result variable lines and labels on the converter symbol; a value of “NO” (default) or “YES”

Attributes configured by the Model Builder user for an delayed converter:

- DELAY INITIAL — the converter’s initial delay value
- DELAY FORMULA — the formula to calculate the converter’s delay value

The attributes of the customized converter are a “superset” of attributes including those of the simple converter for one or more operations and the delayed converter defined above.

Component Interfaces on an converter allowed by the Model Builder process:

- from or to a modeled stock, a modeled flow, a transformer, a converter of a factory as an internal variable value
- from an external variable, an adjusted variable or a user-input variable

Error Conditions detected and prohibited by the Model Builder process:

- positive or negative infinity (largest number) or smallest number (not zero)

Other Behaviors and Characteristics:

- the simple converter symbol shows the operation performed by the converter
- the formula for a summed or multiplied converter is trivial and built automatically as operand variable connections are made to the converter

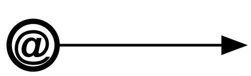
## Variables

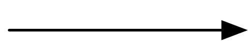
Variables provide a source of numeric information in the model. Variables are utilized by model components that perform a calculation.

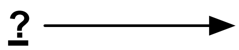
Four variable types are defined for the Model Builder process:

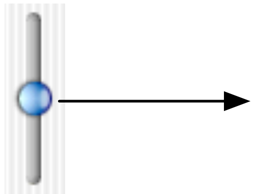
- external variable is a value, record, array or file that is stored on a locally or available from a remote data source such as a USB drive, another connected computer or a website on the internet
- internal variable is a value from another model component such as a stock level or flow rate or a converter result
- user input variable is a value that is input directly by the user
- adjusted variable is defined by the user with a movable control in the user interface

The variable symbol is a line with an arrow pointing to the component performing the calculation. The variable symbol may optionally be drawn with enforced 90° transitions or as a semicircle. The following are variable symbols:


from External Variable source


from Internal Variable source


from User Input Variable source


from Adjusted Variable source

Attributes configured by the Model Builder user for an external variable:
- EXTERNAL REF — the file location or URL for an external variable
- QUERY RULE — the rule to submit a query to the remote data source
- PARSE RULE — the rule to interpret the response to the query

Attributes configured by the Model Builder user for an adjusted variable:
- ADJUST RANGE — slider adjustment range from MIN to MAX
- ADJUST INCREMENT — the increment of the slide adjustment

Component Interfaces on an internal variable allowed by the Model Builder process:
- from or to a modeled stock, a modeled flow, a transformer, a factory of a converter

Component Interfaces on an external, user input or adjusted variable allowed by the Model Builder process:
- to a modeled stock, a modeled flow, a transformer, a factory of a converter

Error Conditions detected and prohibited by the Model Builder process:
- ILLEGAL VARIABLE CONNECTION
  – an attempt to connect a variable from or to an unbounded stock
  – an attempt to connect a variable from or to an unrestricted flow

Other Behaviors and Characteristics:
- none

## Factories

Like a transformer, a factory is analogous to a "macro". A factory is a complex function of any number of model components with a defined interface that can be reused in any model that the user builds. A factory can be warehoused, sold and distributed by third parties. A factory may have dissimilar inflows and outflows that are related to each other over a timeline and through a sequence of operations. A factory may accept an inflow over the timeline of one or more trials in a simulation and perform an analysis to produce a dimensionless statistical result.

A factory has the following basic file structure:
- METEDATA — the minimum metadata to accomplish naming, ownership, version control, traceability, and searchability
- ATTRIBUTE LIST — the attribute list including the common set defined for any component and the unique set defined for the factory

The factory has the following minimum metadata:
- FACT NAME STRING — the plain language name of the factory

- FACT NAME REG — the label assigned to the factory by the Model Builder
- FACT VERSION NUMBER — the factory base reference version number
- FACT VERSION OWNER — the modeler information such as organization name and model URL
- FACT VERSION DATE — the published date
- FACT REVISION NUMBER — an appended factory revision number
- FACT REVISION OWNER — the user who last revised the factory revision
- FACT REVISION DATE — the date of factory revision changes if changes are allowed
- FACT COMPLIANCE — the version of the global system dynamics modeling standard followed by the Model Builder process used to build the factory
- FACT CUSTOMIZED SYMBOL — customized tiff file for this factory
- LOCK MODEL — a value of "NO" (default) or "YES" to prevent Model Builder changes to the factory by a user who is not the factory owner, factory version owner, factory revision owner or authorized by a password
- FACT DESCRIPTION — an abstract of the factory's scope and use – a "Read Me"

Attributes configured by the Model Builder user for any factory:

- GRAPH — display a high quality formatted graph of an OUTPUT value; a value of "NO" (default) or "YES"
- TABLE — display a high quality formatted table of an OUTPUT value; a value of "NO" (default) or "YES"
- ABSTRACT — publish OUTFLOW to abstraction layers; a value of "NO" (default) or "YES"
- COST — publish COST attributes to abstraction layers; a value of "NO" (default) or "YES"

### *Production Factories*

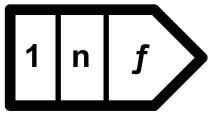
A production factory typically operates on tangible inflows. The output of a production fatory is a flow of a new good that is a consolidation of the goods in the inflows over a period of time. Two fundamentally different production fatories are defined to support a parallel process and a serial process.

A production factory is an expandable model component with an initial structure, a final structure and a generic structure. Expanding a production factory requires an additional generic structure in the underlying model of the factory.

Attributes optionally configured by the Model Builder user for any production factory to describe the corresponding Cost Abstraction Layer basis:

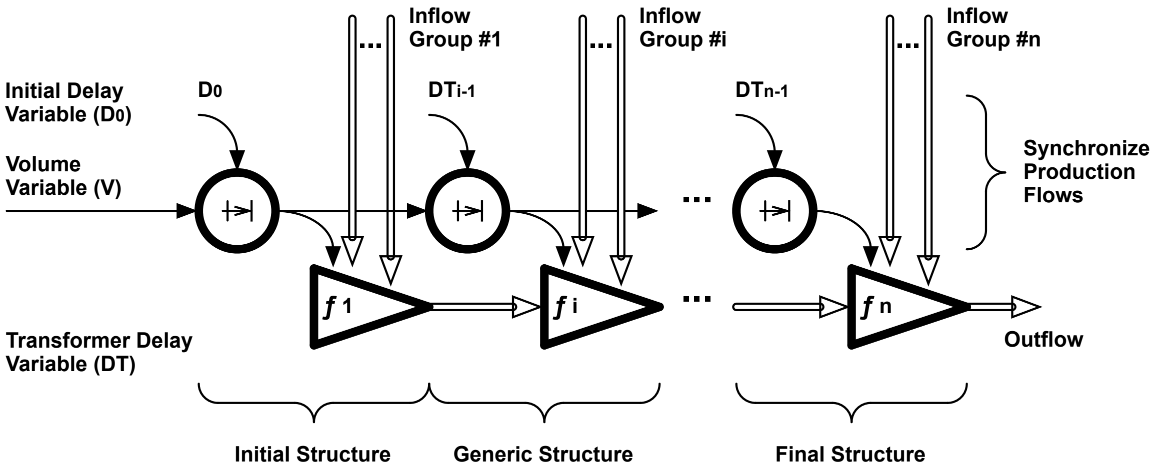
- COST ACCUM PERIODIC` — accumulate cost for the period described in the simulation specification; a value of "NO" (default) or "YES"
- COST FIRST INITIAL — a one-time "first cost" for the initial factory structure
- COST FIRST FINAL — a one-time "first cost" for the final factory structure
- COST FIRST GENERIC — a one-time "first cost" for the generic factory structure
- COST TIME INITIAL — a cost per time interval for the initial factory structure
- COST TIME FINAL — a cost per time interval for the final factory structure
- COST TIME GENERIC — a cost per time interval for the generic factory structure
- COST GOOD INITIAL — a cost per unit of good outflowed for the initial factory structure
- COST GOOD FINAL — a cost per unit of good outflowed for the final factory structure
- COST GOOD GENERIC — a cost per unit of good outflowed for the generic factory structure

Several generic production factories are defined:

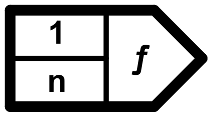



Generic Serial Production Factory

The serial production factory is an expandable model component that is represented by the following basic model fragment:

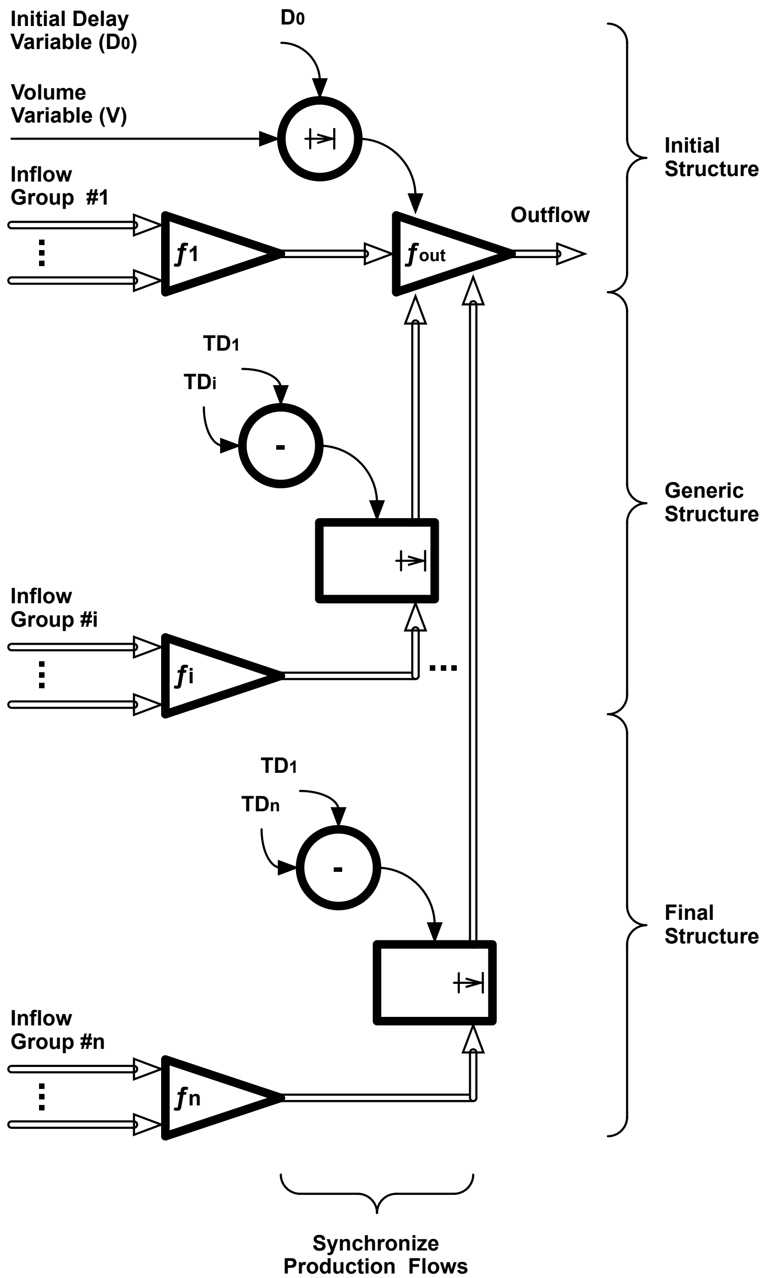



The orientation of the production factory structure diagrams deliberately mimics the factory symbol. The serial production factory features a production synchronization mechanism to account for the additive delay or each sequential transformer that is unnecessary for the parallel production factory case.




Generic Parallel Production Factory

The parallel production factory is an expandable model fragment that is represented by the following basic model:

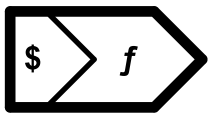



Significant work remains to describe production factories.

## *Finance Factories*

A finance factory is defined for accounts that are reported in Income Statements for sources and uses of cash and Balance Sheets for assets, liabilities and equity. This factory is typically applied on the cost abstraction layer of the model, although it may be applied on the primary model layer or any other model abstraction layer that is defined.

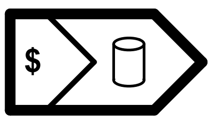
The following generic finance factory symbol is defined:



Generic Finance Factory – Dollars

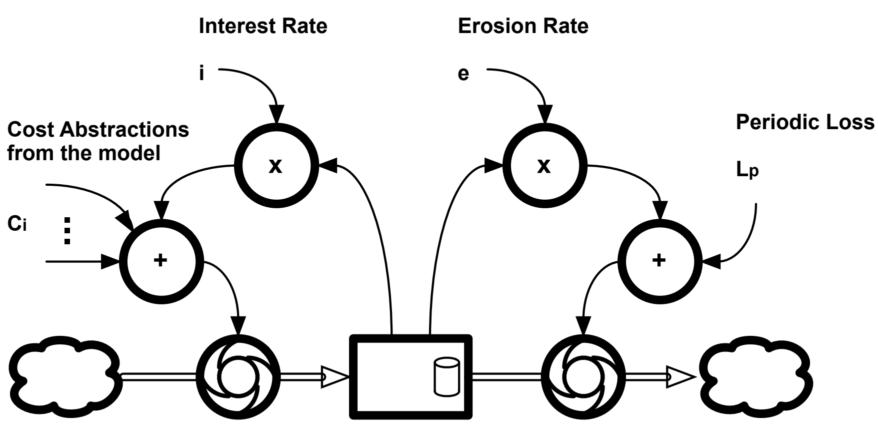
The currency symbol is displayed for US Dollars, Euros, Pounds, Rubles, Yen, Yuan, Won, Rupees or gold ounces. Other currencies may also be supported.

A finance factory is envisioned:



Cost Account Finance Factory

The cost layer finance factory is represented by the following basic model fragment:

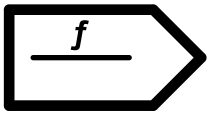



Significant work remains to describe finance factories.

## *Analysis Factories*

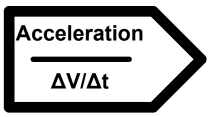
General analysis factories apply various methods to a single set of data over a trial of time intervals. The Model Simulator provides methods to analyze univariate data from a simulation for volatility, central tendency and cyclic characteristics with analysis factories.

The following generic analysis factory symbol is defined:

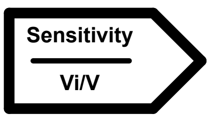



Generic Analysis Factory

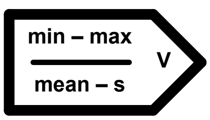
The following analysis factories are envisioned:




Acceleration Analysis Factory
(first derivative of a modeled stock's net flow, a modeled flow's outflow rate, or a transformer's outflow rate



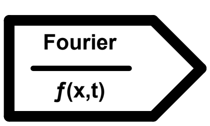

Sensitivity Analysis Factory




Descriptive Statistics Analysis Factory
(min, max, arithmetic mean, standard deviation and coefficient of variation; optionally, geometric mean, harmonic mean, median or mode instead of arithmetic mean)


Standard Moments Analysis Factory
(standard variance, skewness and kurtosis)


Normal Distribution Confidence Analysis Factory
(⅔σ = 50%, 1σ = 68%, 1½σ = 87%, 2σ = 95%, 3σ = 99.7% confident)



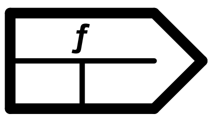

Fourier Transform Analysis Factory

Significant work remains to describe analysis factories.

### *Optimization Factories*

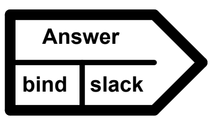
Optimization factories improve upon a guess for a solution by varying one or more variables according to a set of constraints. The Model Simulator provides methods to optimize model behavior and perform goal seeking with optimization factories.

The following generic optimization factory symbol is defined:

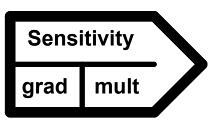



Generic Optimization Factory

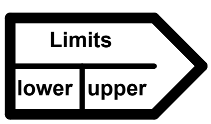
The following optimization factories are envisioned:




Answer Optimization Factory
(returns answer, binding variables and slack of nonbinding variables)




Sensitivity Optimization Factory
(returns reduced gradient and Lagrange multiplier of adjusted variables)




Limits Optimization Factory
(returns the lower and upper limits of adjusted variables)

Significant work remains to describe optimization factories.

## Abstraction Layers

An abstraction layer is analogous to a new worksheet in an Excel™ workbook that is linked to a main worksheet. Any value in the primary model can be "published" to the abstraction layers. When an abstraction layer is created, the published values in the primary model layer (or in any other abstraction layer) are made available to a separate model in the new abstraction layer. The abstraction layer model is built and modified in the same manner as the primary layer model with any additional model components and relationships that the user chooses to incorporate into the abstraction layer. Internal variables on an abstraction layer may be selected from the published or ABSTRACTED values in a pull-down or pop-up list.

An abstraction layer has the following basic file structure:

- METEDATA — the minimum metadata to accomplish ownership, version control, security, traceability, and searchability
- PREFERENCES — a set of settings that affects appearance, abstraction and other lsyer-

specific qualities
- COMPONENT LIST — the unique set of registered names of model components and fragments contained in the abstraction layer

An abstraction layer has the following minimum metadata:
- MODEL NAME REG — the label assigned to the model by the Model Builder
- LAYER NAME STRING — the plain language name of the fragment
- LAYER NAME REG — registered name; the label assigned to the fragment by the Model Builder
- LAYER OWNER — the Model Builder user who initially defined the layer in the model
- LAYER VERSION NUMBER — the fragment base reference version number
- LAYER VERSION OWNER — the modeler information such as organization name and model URL
- LAYER VERSION DATE — the published date
- LAYER REVISION NUMBER — an appended fragment revision number
- LAYER REVISION OWNER — the user who last revised the fragment revision
- LAYER REVISION DATE — the date of fragment revision changes if changes are allowed
- LAYER COMPLIANCE — the version of the global system dynamics modeling standard followed by the Model Builder process used to build the layer
- LOCK LAYER — a value of "NO" (default) or "YES" to prevent Model Builder changes by a user who is not the model owner, model version owner, model revision owner, layer owner, layer version owner, layer revision owner or authorized by a password
- LAYER DESCRIPTION — an abstract of the layer's scope and use – a "Read Me"

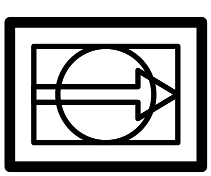
Support for an abstraction layer is built into the System Dynamics Calculator to augment the standard model. The abstraction layer operates on values that have been explicitly "published" or ABSTRACTED in the primary model layer. The ABSTRACTED values in the model's primary layer are available in a pick list for internal variables used in the abstraction layer primarily by factories.

The following functions are required in the abstraction layer:
- a list of ABSTRACTED values that are available to abstraction layers to pick from when defining formulas for any model component that is placed in the abstraction layer

## Fragments

A Fragment is a subset of a particular model that the user defines for convenience – a set of stocks, flows, converters, variables, transformers and factories in the model that are related to each other. A typical relationship would be a causal loop in the model.

The Fragment symbol is rectangle that contains a stock, flow and converter symbol superimposed. The following is the fragment symbol:


Fragment

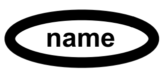
Attributes configured by the Model Builder user for any fragment:
- COMPONENT LIST — the unique set of registered names of model components and fragments contained in the fragment

The fragment component list is retained in a manner that allows the fragment's contained components to be easily distinguished from the rest of the model components. In this way, the fragment can be "exploded" to expose its constituent model pieces and reconstructed to conceal its constituent model pieces on demand by the user without loosing information that describes connectivity with the rest of the model.

## Scripts

Scripting is a system-level feature that provides control over system attributes and various calculator features. A script can be warehoused, sold and distributed by third parties. A script type is placed on a new palette when the first script is defined. To operate a script, it is simply placed on the canvas. A script on the canvas is executed every time the simulation is run. More than one script can be run with a simulation.

The script symbol is a stylized oval such as the following:



Script

The script has the following basic file structure:

- METEDATA — the minimum metadata to accomplish naming, ownership, version control, traceability, and searchability
- SCRIPT — the sequence of instructions

The script has the following minimum metadata:

- SCRIPT NAME STRING — the plain language name of the script
- SCRIPT NAME REG — the label assigned to the script by the Model Builder
- SCRIPT OWNER — the Model Builder user who initially defined the script
- SCRIPT VERSION NUMBER — the script base reference version number
- SCRIPT VERSION OWNER — the modeler information such as organization name and script URL
- SCRIPT VERSION DATE — the published date
- SCRIPT REVISION NUMBER — an appended script revision number
- SCRIPT REVISION OWNER — the user who last revised the script revision
- SCRIPT REVISION DATE — the date of script revision changes if changes are allowed
- SCRIPT CUSTOMIZED SYMBOL — customized tiff file for this script
- LOCK SCRIPT — a value of "NO" (default) or "YES" to prevent Model Builder changes to the script by a user who is not the script owner, script version owner, script revision owner or authorized by a password
- SCRIPT DESCRIPTION — an abstract of the script's scope and use – a "Read Me"

Many controls performed by a script are evident by "verbs" of "get" and "set" from the function list that the calculator supports. For example, scripting supports the following controls and functions:

- Get / Set an attribute (of a named model component in a named model)
- Get / Set Date and Time, Get / Set Date, Get / Set Time, Get / Set Time Zone
- Get User
- Get / Set Location, Language
- Get / Set Number System, Convert Number from / to
- Get / Set Currency System, Convert Currency from / to
- Get / Set Unit System, Convert Unit from / to

- Wake, Sleep, Shut Down

The following controls are functions of the model simulator that can be scripted:

- Load, Unload (a model)
- Select (a page, palette, simple table/graph)
- Start, Stop, Pause, Resume (a simulation, movie)
- Display, Clear Display (a result, text string, picture)
- Print (a model, graph or table)

Controls may be executed immediately upon or “On” the detection of system events such as the following events:

- Calculator Start-Up / Shut-Down
- User Login / Logout
- Simulator Idle / Running
- Simulation Start / Finish, Trial Start / Finish, Interval Start / Finish
- Character entry, String entry
- Network Services Established / Lost, Available / Unavailable
- External Power Connected / Disconnected, Available / Unavailable
- Internal Power Available / Unavailable, 100%, >25%, >5%, >0%

Controls may be executed “At” a specific date and time, or just at a specific time on every date, or after an interval of time from now, or “After” an interval of time from a specified date and time. Controls may be executed conditionally with an “If, Then, Else” logic. A script may perform or “Do” another named script.

# CUSTOMIZED FACTORY PACKAGES

Customized factory packages are shared or purchased and loaded on demand to provide the user with an application extension of factories and a “cookbook” to model and solve a particular problem with their System Dynamics Calculator. A comprehensive list of solutions that might be supported by customized factory packages is beyond the scope of this document. However, there are several packages of factories that are envisioned and discussed to illustrate the potential of this element of the System Dynamics Calculator.

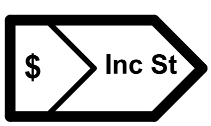
The intent of a customized factory package is to give the user a palette of familiar terms and concepts to simply pick from with a minimum level of understanding required. Embedded in each factory is a set of interfaces into and out of the factory, and a behavior that is configurable within some bounds. For some factories, the embedded interfaces and behaviors will be identical, but the label is distinguishing; this approach eliminates a step to label the factory, and limits errors by reducing the knowledge required to apply the factory correctly.

## Finance Factory Package

Finance Factories focus on the processing of money flows commonly found in households, business operations, banks and investment institutions.

Accumulation Account Finance Factories


Balance Sheet Account Finance Factory

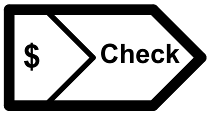



Income Statement Account Finance Factory

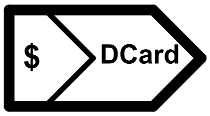
### *Transaction Account Finance Factories*

A number of factories are defined for common accounts that are maintained by institutions. Several finance factories are defined for conventional transaction accounts:

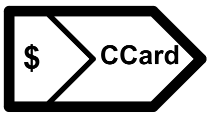

Cash Savings Account Finance Factory




Checking Account Finance Factory



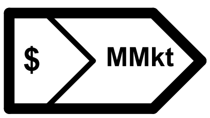

Debit Card Account Finance Factory




Credit Card Account Finance Factory

### *Investment Account Finance Factories*

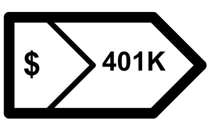
Several factories are defined for conventional investment accounts:



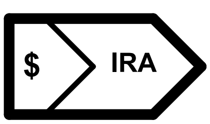

Illiquid (money market) Investment Account Finance Factory


Illiquid (non-cash) Investment Account Finance Factory
(note that the "Invest" badge may be replaced with a trading symbol)

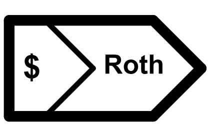
Several factories are defined for investment accounts that receive special tax treatment:




401K Account Finance Factory




IRA Account Finance Factory




Roth Account Finance Factory

### *Periodic Payment Finance Factories*

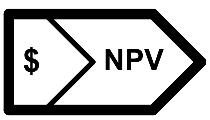
Several finance factories are defined for equal periodic payment accounts:


Annuity Account Finance Factory

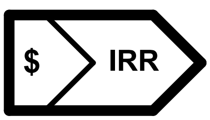

Loan Account Finance Factory

### *Past Flow and Future Flow Finance Factories*

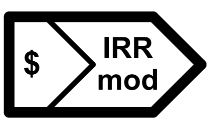
Several finance factories are defined that perform operations on past flows and future flows:



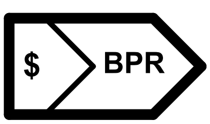

Net Present Value Finance Factory



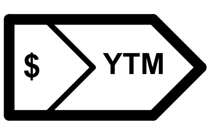

Internal rate of Return Finance Factory




Modified Internal Rate of Return Finance Factory




Bond Price Finance Factory



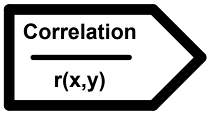

Bond Yield to Maturity Finance Factory

Significant work remains to describe finance factories.

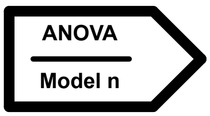
## Analysis Factory Package

Analysis factories apply a number of quantitative analysis techniques to the results of a simulation including inferential statistical techniques to see a pattern or a significant characteristic. In general, hypothesis testing and statistical tests on more than one set of data are covered below:

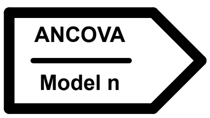
The following analysis factories are envisioned:



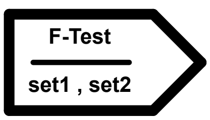

Correlation Analysis Factory



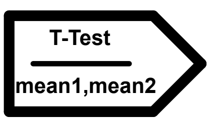

Analysis of Variance Analysis Factory
(Model 1 for fixed effects, Model 2 for random effects, Model 3 for mixed effects)




Analysis of Covariance Analysis Factory
(Model 1 for fixed effects, Model 2 for random effects, Model 3 for mixed effects)

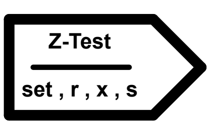



F-Test Probability Analysis Factory
(returns one-tailed P that two related data sets are statistically similar)




T-Test (Student's t-test) Probability Analysis Factory
(returns two-tailed P of statistical significance for a small sample)


Chi-squared Test ($X^2$-Test) Probability Analysis Factory
(returns the P that two independent data sets are statistically similar)




Z-Test Probability Analysis Factory
(returns two-tailed P of statistical significance for a population)

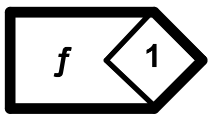
Significant work remains to describe analysis factories.

## Natural Science Factories

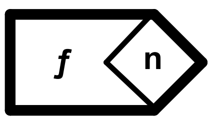
A Natural Science Factory describes characteristics and behavior of biological systems including people. Two types of factories are anticipated: "non-thinking" or evolution factories, and "thinking" or decision factories.

A Decision Factory typically operates on intangible inflows. The output of the decision factory is a Boolean indicator that is either "true" or "false". The decision factory represents decision making of an individual, a population sample or the population itself as simulation methods are applied differently for each case.

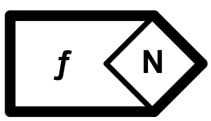
Several generic decision factories are envisioned:



Generic Individual Decision Factory



Generic Sample Decision Factory



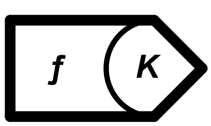
Generic Population Decision Factory

Significant work remains to describe natural science factories.

## Physical Science Factories

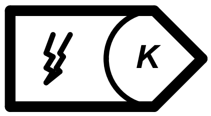
A Physical Science Factory describes characteristics and behavior of non-living systems. Two types of factories are anticipated: "micro-scale" or atomic factories for elemental particles, atoms or molecules, and "macro scale" or physical factories for objects.

The following generic atomic factory symbol is defined:

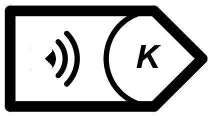


Generic Atomic Factory for an element or compound (*K*)

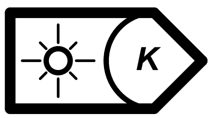
Several atomic factories are envisioned:



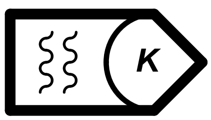

Electrolytic Reaction Factory



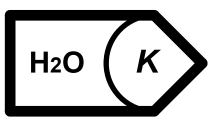

Radiolytic Reaction Factory



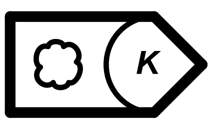

Photolytic Reaction Factory



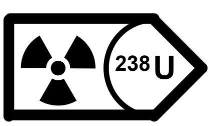

Thermolytic Reaction Factory




Hydrolytic Reaction Factory




Combustion Reaction Factory




Nuclear Reaction Factory (for Uranium, for example)

Significant work remains to describe physical science factories.

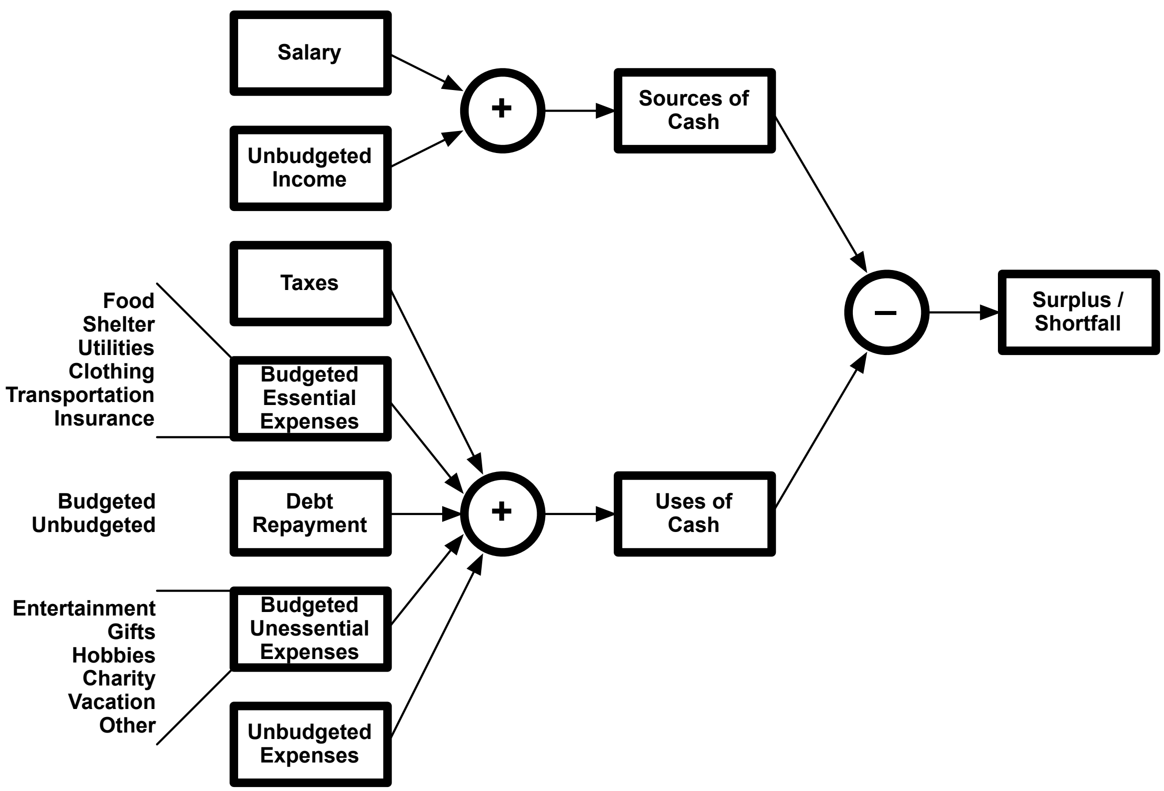
# THE MODELING PROCESS

## The Personal Finance Model

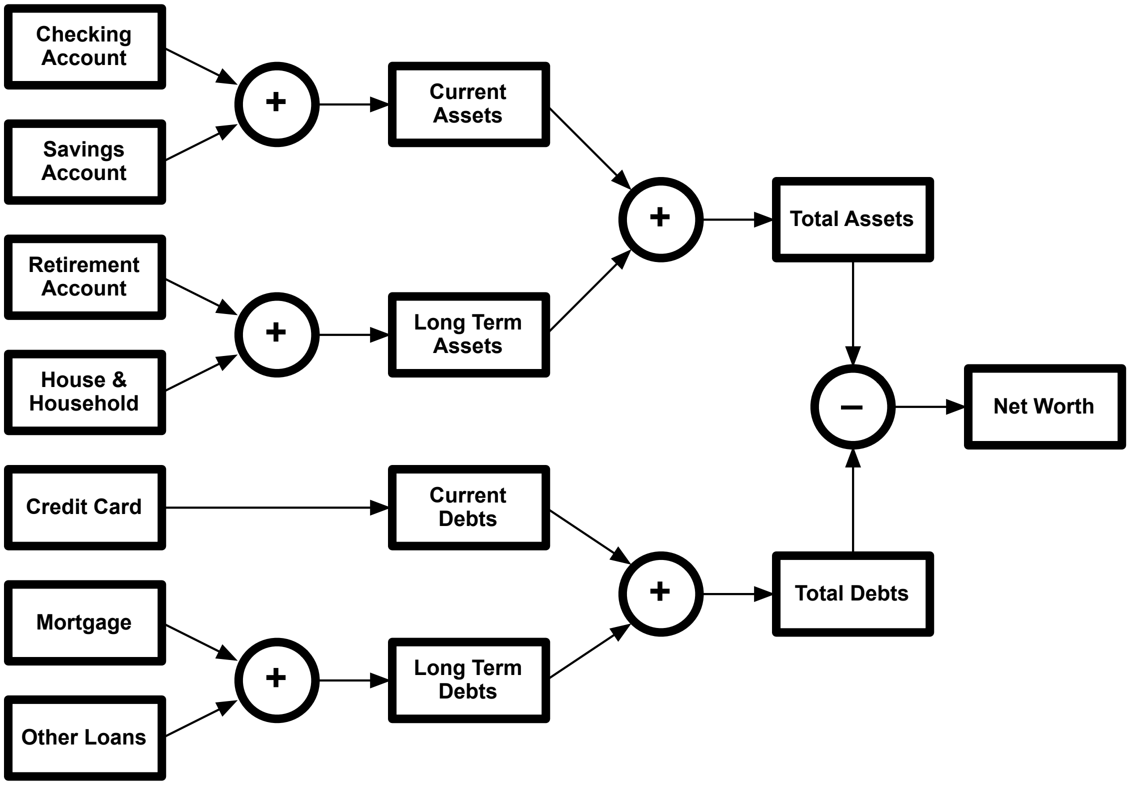
The modeling process begins with the identification of objectives and relationships that are relevant to the user. In the following example of a household finance and budget problem, the concept of a stock and a flow are explored. The user has an accurate “Income Statement” of sources and uses of their cash from their personal finance software such as Quicken. In their planning, the user has a budget, a contingent plan to survive a financial crisis or a “shortfall”, a wish list of uses for a windfall or a “surplus” and an objective to eliminate their credit card debt. While their plans are fairly well understood, the plans are relatively “static”, and the planning horizon is generally short term.

### *Entities Kept and Measured*

The following are two simple drawing of household expenses that the user has identified in their finances. The user demonstrated a poor understanding of finance by initially failing to distinguish between the transient nature of food expenses for example and the persistent nature of their mortgage loan. This naivety demonstrates a real need for templates and ready-built models.



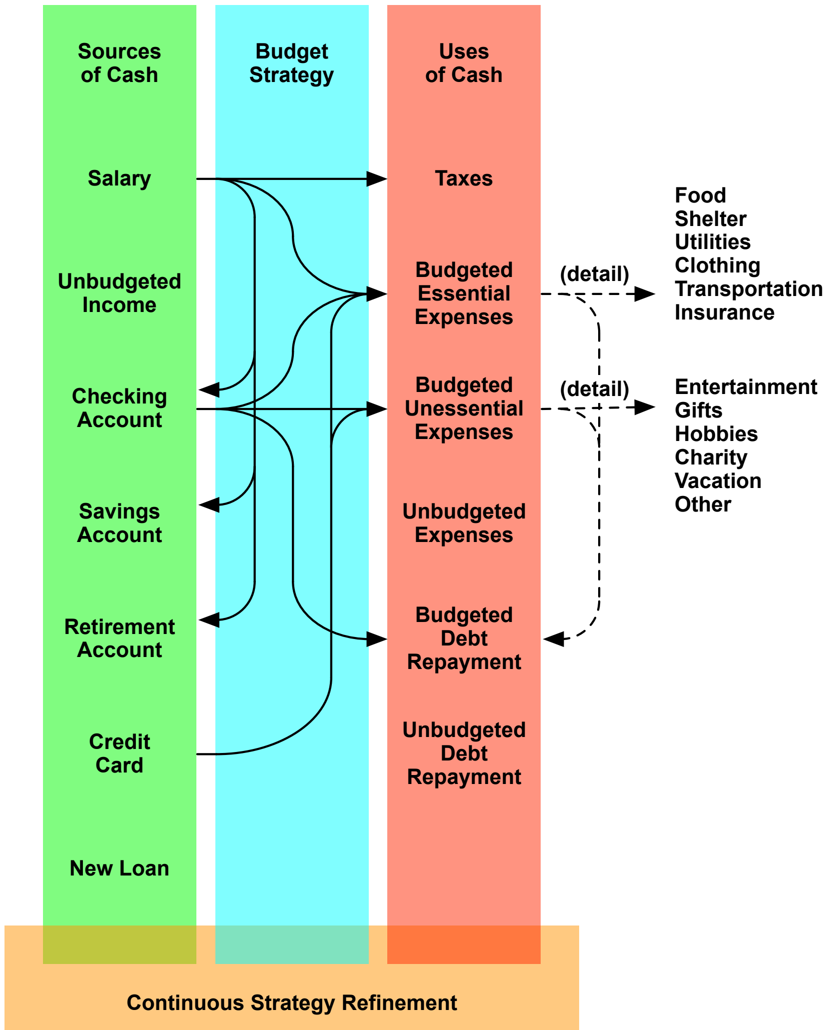

Typical Household Finances Mapped to an Income Statement




Typical Household Finances Mapped to a Balance Sheet

### *Flows Between the Entities*

The user describes their finances further by considering their budgeting process in some detail. The three drawings that follow illustrate the different flows that the family experiences as their finances experience perturbations from shortfalls and windfalls. The following drawing illustrates the normal budget allocation process and the flows that result:

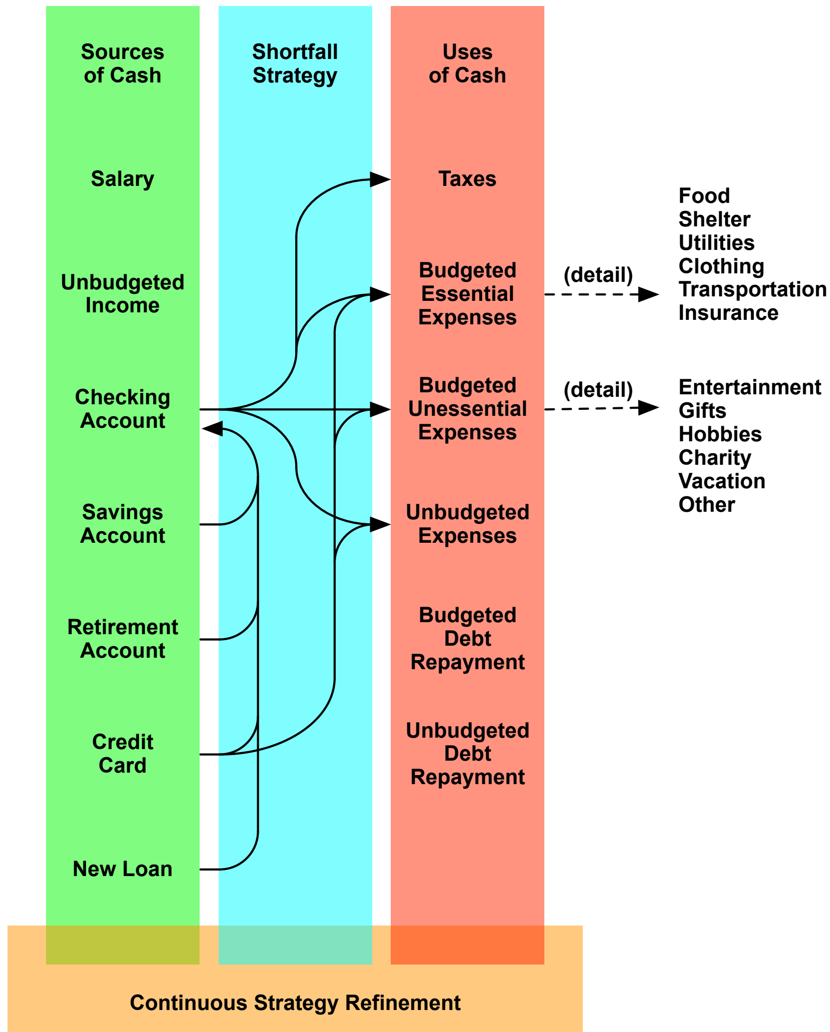



Typical Household Current Period Flows

The budgeting process is an allocation problem to suit a number of financial and lifestyle objectives:

- meet all essential living expenses and obligations through a sensible combination of cash and credit
- meet a reasonable level of unessential living expenses through a sensible combination of cash and credit
- save for a lapse of income, a vacation or a “rainy day”
- save for a comfortable retirement

The budgeting strategy is refined continuously to address changes in financial and lifestyle objectives, surpluses and shortfalls. The following drawing illustrates changes in budgeted flows necessary to manage a shortfall:



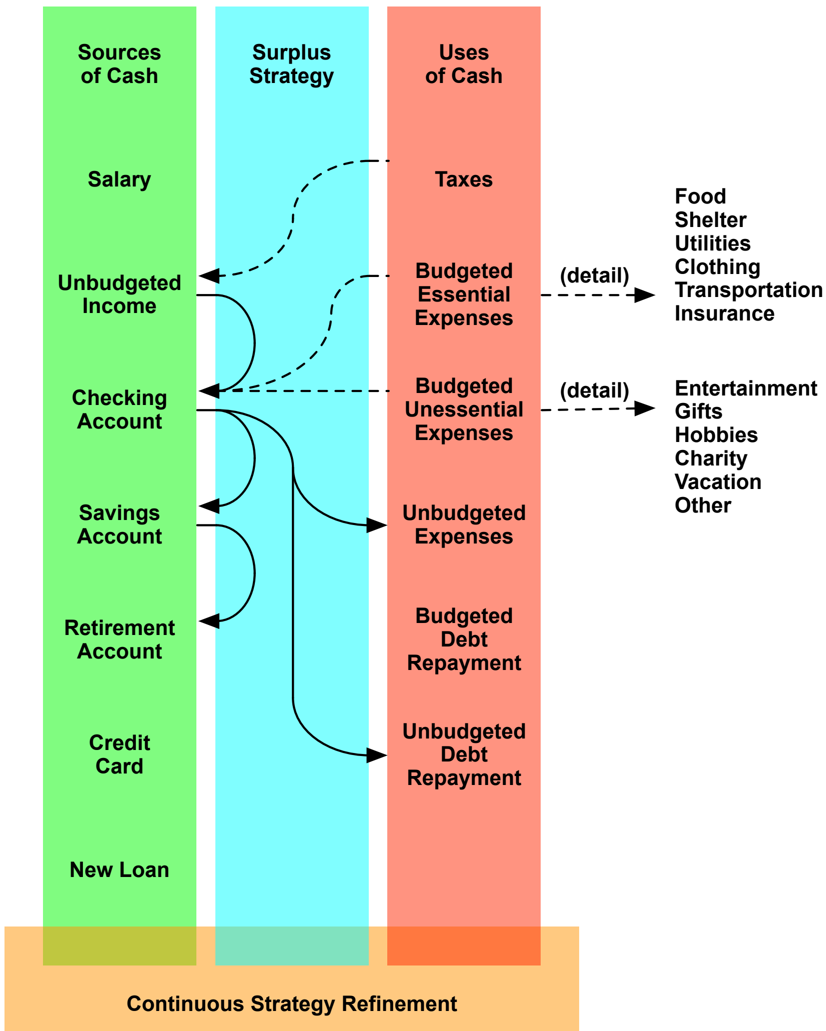

Potential Flows to Respond to Shortfalls

Shortfalls are the result of several scenarios:

- marginally over budget for current expenses
- unbudgeted current expenses such as automobile repairs or additional tax payments
- unanticipated permanent changes in budgeted expenses such as a “stair step” increase in insurance premiums or in mortgage interest rate
- unbudgeted long term expenses such as a new car or house
- loss of income

Shortfall management is a process of providing cash to meet the shortfall from available sources such as the savings account and by increasing the use of credit in a manner that minimizes the impact to a person’s longer term financial and lifestyle objectives. Insufficient levels of cash in the checking account or excess levels of credit card balances “trigger” the shortfall management process. If the shortfall management process is initiated for a number of consecutive periods of time, a rebudgeting process is triggered. Since the impact of a shortfall is immediate and detrimental, the rebudgeting process is usually triggered quickly after a small number of periods.

The following drawing illustrates changes in budgeted flows necessary to manage a windfall or surplus:




Potential Surplus Sources and Reallocation Flows

Surpluses are the result of several scenarios:

- marginally under budget for current expenses due to less consumption or improved “budget-mindedness”
- an unanticipated receipt of cash from a rebate or a tax refund
- a long term increase in credit available
- a long term increase in income such as a raise in salary

Surplus management is a process of allocating the surplus between savings vehicles and consumption in a manner that improves a person’s financial and lifestyle objectives. Excess levels of cash in the checking account or savings account “trigger” the surplus management process. If the surplus management process is initiated for a number of consecutive periods of time, a rebudgeting process is triggered. Since the impact of a surplus is not significant in the near term, there is no urgency to initiate a rebudgeting process.

### *Internal and External Data Sources*

To apply the System Dynamics Calculator, sources of essential data are identified. The following are sources of data:

**Personal Finance Model**
**Local Sources**

**Local Model Library:**

- **Household Budget Strategy**
- **Retirement Account Strategy**
- **Other Models**

**Local TAL Accounts Data Import:**

- **Checking Account**
- **Savings Account**
- **Retirement Accounts**
- **Charge Accounts**
- **Other Accounts**

**Individual Financial Model Preference Basis:**

- **Personality**
  - **Leader / Follower**
  - **Risk Aversion**
  - **Confidence Today / Future**
  - **QoL Present vs. Future**
  - **Other Factors**
- **Individual Account Thresholds**
  - **Cash Accounts Min / Max**
  - **Credit Accounts Max**
- **Rebudget Triggers**
  - **On Shortfall # of Periods**
  - **On Surplus # of Periods**
- **External Influences**
  - **CPI, CCI, DJIA**
  - **1yr, 10yr, 30yr TBill**

**Personal Finance Model**
**Remote Sources**

**Remote Model Libraries:**

- **Club of Rome World Models**
- **Thwink Models**
- **SD Society Models**
- **US Government Models**
- **Other Models**

**Financial Markets Data Import:**

- **DJIA**
- **Treasuries**
- **Futures**
- **Global Markets**
- **Other Indicies**

**US Census Bureau Data Import:**

- **Real GDP**
- **Unemployment**
- **CPI**
- **CCI**
- **Other Statistics**

# APPENDIX: FUNCTIONS

Functions are the basis for modeled flow and instantaneous converter definitions. Functions also define the basis for any customized model component or factory. Functions encompass mathematical, statistical, financial and date-time operations. Where a symbol's graphic cue is a stylized italic "***f***" as is the case with customized model components and factories, and only one operation is applied, the graphic cue will show the operation.

The following functions are implemented in the SD Calculator. They are essential to build sophisticated system dynamics models.

## Mathematical, JAVA Strict Math and fdLibM (IEEE 754) Functions

The following mathematical functions are implemented by a converter.

***Constants***

| | |
|---|---|
| • Zero, One | 0, 1, -1 |
| • Pi | π |
| • Euler's constant | e |
| • Largest Number | +/- ∞ or +/- $(2\text{-}2^{-23}) \times 2^{127}$ or approximately +/-3.4028 x $10^{38}$) |
| • Smallest Number | +/- 1/∞ or $2^{-126}$ or approximately +/-1.1755 x $10^{-38}$ |
| • nan | Not a number – a defined binary byte |
| • null | Null - nothing |

Note that extended and double precision calculations will have different "smallest" and "largest" numbers

***Arithmetic operations***

| | |
|---|---|
| • Sum | x + y |
| • Difference | x – y |
| • Product | x * y |
| • Quotient | x / y |

***Logical comparison operations***

| | |
|---|---|
| • Equal | = |
| • Not Equal | ≠ |
| • Less than | < |
| • Greater than | > |
| • Less than or equal | ≤ |
| • Greater than or equal | ≥ |

***Boolean comparison operations***

| | |
|---|---|
| • Is | ≡ |
| • Not | ¬ |
| • And | ∧ |
| • Or | ∨ |
| • Exclusive Or (XOR) | ⊻ |

Negation is also allowed such as Not And or Nand (¬∧ or ⊼)

***Conversions to integer in floating-point type***

| | |
|---|---|
| • Round | round(x) |
| • Round downward | floor(x) |
| • Round upward | ceil(x) |
| • Round toward zero | trunc(x) |

***Comparison operations***

| | |
|---|---|
| • Minimum | min(x,y) |
| • Maximum | max(x,y) |
| • Positive difference | dim(x,y) |

***Sign manipulation functions***

| | |
|---|---|
| • $\|x\|$ | abs(x) |

***Exponential functions***

| | |
|---|---|
| • $e^x$ | exp(x) |
| • $x^y$ | pow(x,y) |

***Logarithmic functions***

| | |
|---|---|
| • ln x | log(x) |
| • $\log_{10} x$ | log10(x) |

***Trigonometric functions***

| | |
|---|---|
| • cos x | cos(x) |
| • sin x | sin(x) |
| • tan x | tan(x) |
| • arccos x | acos(x) |
| • arcsin x | asin(x) |
| • arctan x | atan(x) |

***Hyperbolic functions***

| | |
|---|---|
| • cosh x | cosh(x) |
| • sinh x | sinh(x) |
| • tanh x | tanh(x) |
| • arccosh x | acosh(x) |
| • arcsinh x | asinh(x) |
| • arctanh x | atanh(x) |

***Error and gamma functions***

| | |
|---|---|
| • error x | erf(x) |
| • Γ(x) | gamma(x) |

***Miscellaneous functions***

| | |
|---|---|
| • Degrees to radians | toRadians |
| • Radians to degrees | toDegrees |
| • Square root | sqrt(x) |
| • Cube root | cbrt(x) |
| • Hypotenuse | hypot(x,y) |
| • Remainder | remainder(x,y) |
| • Sign of a number | signum(x) |
| • Random number | random(x) |

## Financial Functions

Financial functions are implemented either in a converter or a factory model component. The decision of which implementation is chosen rests on the time-wise nature of the function and the function library utilized by the developer.

***Time value of money operations***

Source data for time value of money calculations (solve for any one given the others):
- PV present value
- FV future value
- n number of periods
- I interest rate
- PMT payment (and payment timing at the beginning or end of the period)

Two period specifications are processed:
- N now
- m for a period number from 0 to n

- P principle part of payment now N, at a period m
- I interest part of payment now N, at a period m
- ΣP cumulative principle paid now N, at a period m
- ΣI cumulative interest paid now N, at a period m
- ΣPMT cumulative payments paid now N, at a period m
- V value now N, at a period m
- m period for a value V
- mPV periods since present value for a value V or now N
- mFV periods to future value for a value V or now N

***Depreciation operations***

Source data for asset depreciation calculations:
- Cost (basis)
- Salvage value at end of life
- Lifetime (months or years)
- Start of period (start of year, middle of year, a month of a year, or a date)

Four depreciation methods are processed:
- SL straight line
- DB declining balance
- DDB double declining balance
- SYD sum of years digits

Two period specifications are processed:
- N now
- m for a period number from 0 to n

Three results are calculated :
- D depreciation for a specified period
- ΣD accumulated depreciation at a specified period
- DV depreciated value at a specified period

- SL depreciation now N, for a period m
- DB depreciation now N, for a period m

- DDB depreciation now N, for a period m
- SYD depreciation now N, for a period m
- SL accumulated depreciation now N, at a period m
- DB accumulated depreciation now N, at a period m
- DDB accumulated depreciation now N, at a period m
- SYD accumulated depreciation now N, at a period m
- SL depreciated value now N, at a period m
- DB depreciated value now N, at a period m
- DDB depreciated value now N, at a period m
- SYD depreciated value now N, at a period m

***Ratios calculations***

- Current ratio
- Quick ratio
- Cash ratio
- Cash coverage ratio
- Times interest earned
- Asset turnover
- Inventory turnover
- Accounts payable turnover
- Accounts receivable turnover
- Average collection period
- Net profit margin (NP)
- Gross profit margin (GP)
- Return on assets (ROA)
- Return on equity (ROE)
- Return on investment (ROI)
- Dividend payout ratio (DPO)
- Earnings per share (EPS)
- Price earnings ratio (P/E)
- Equity multiplier ratio (A/E)
- Debt equity ratio (D/E)
- Debt assets ratio (D/A)

***Other financial operations***

- Annual Percentage Rate (APR)
- Net Present Value (NPV)
- Internal Rate of Return (IRR, modified IRR)
- Bond Yield to Maturity (YTM)
- Bond Price (BPR)
- Capital Asset Pricing Model (CAPM)
- Constant Growth Model (CGM)
- Option Pricing Model (OPM)

## Statistical Functions

Statistical functions are typically implemented by a factory model component as they operate on one or more data sets that might ordinarily result from a simulation. In conventional calculator mode, the statistical factories accept inputs in the form of an array or a matrix of data.

***Descriptive Statistics***

- Count
- Minimum
- Maximum
- Range
- Mean (arithmetic, geometric, harmonic)
- Median
- Mode
- Quantile (quartile, quintile, decile, percentile)
- Deviation (standard, mean absolute, median absolute, maximum absolute)
- Coefficient of variation
- Variance
- Semivariance
- Skewness
- Kurtosis

***Inferential Statistics***

- ANOVA analysis of variance
- ANCOVA analysis of covariance
- Normal probability density
- Normal cumulative distribution
- Binomial individual terms distribution
- Binomial cumulative terms distribution
- Poisson individual terms distribution
- Poisson cumulative terms distribution
- Beta distribution
- Gamma distribution
- Weibull distribution
- Z (normal) test
- T distribution
- T test
- Chi-squared distribution
- Chi-squared test
- F distribution
- F test

***Correlation Analysis***

- Pearson product-moment
- Spearman's rho
- Kendall's tau

***Regression Analysis*** (this section requires additional work)

## Physical and Chemical Functions (this section requires additional work)

Physical and chemical functions are implemented either in a converter of a factory model component. The decision of which implementation is chosen rests on the time-wise nature of the function and the function library utilized by the developer.

***Reactions of ions, atoms and molecules***

- acidity
- basicity
- oxidation
- reduction
- metal exchange
- pericyclic

***Reactions of structural change***

- addition
- elimination
- substitution
- rearrangement

***Reactions of functional groups***

- alkane
- alkene
- alkyne
- halide
- alchohol
- ether
- amine
- phosphene
- benzene
- nitro
- thiol
- sulfide
- nitrile
- aldehyde
- kytone
- carboxylic
- carboxylic derivative
- ester
- halide
- amide
- anhydride

## Date and Time Functions

***Types supported:***

- Structure of Date and Time record
  year;
  month;
  date;
  hour;
  minute;
  second;
  monthOfYear;
  dayOfWeek;
  timeZone

- Date and Time numeric string
  YYYY-MM-DD HH:MM:SS±HHMM

- Structure of Long Date and Time record
  era;
  year;
  month;
  date;
  hour;
  minute;
  second;
  millisecond;
  am/pm;
  timeZone

### *Basic date and time functions*
- Get date and time
- Get date
- Get time
- Get time zone
- Get dayNumberOfWeek
- Get dayNumberOfYear
- Get weekNumberOfMonth
- Get weekNumberOfYear
- Set date and time
- Set date
- Set time
- Set time zone

### *Convert date and time strings into numeric representations*
- String to date and time
- String to date
- String to time

### *Convert numeric representations into date and time strings*
- Date and time to string
- Date to string
- Time to string

### *Time interval operations*
- Date with time interval since now
- Date interval since now; Time interval since now
- Date with time interval since date
- Date interval since date; Time interval since date
- Add date interval
- Add time interval
- Subtract date interval
- Subtract time interval

### *Date and time comparison operations*
- Compare date and time
- Compare date
- Compare time

- IsEqualTo date
- Earlier date
- Later date

## Other Functions

### *Personalization and Localization*

- Get user
- Get / Set language
- Get/ Set location
- Get / Set default symbol size of small, medium or large
- Get / Set default symbol spacing of tight, moderate or loose

### *Number conversion*

- Get / Set number system
- Convert to Base2
- Convert to Base8
- Convert to Base10
- Convert to Base16
- Convert from / to

### *Notation conversion*

- Convert to fraction (to k/n or $k/2^n$ for $n \leq$ user specification)
- Convert to decimal

### *Currency conversion (ISO 4217, 10646)*

- Get / Set currency system
- Convert to USD ($) United States Dollar
- Convert to AUD (A$) Australian Dollar
- Convert to CAD (C$) Canadian Dollar
- Convert to MXN (M$) Mexican Peso
- Convert to EUR (€) Euro
- Convert to GBP (£) Pound
- Convert to RUB (Руб) Ruble
- Convert to JPY (¥) Yen
- Convert to CNY (元) Yuan Renminbi
- Convert to KRW (₩) Won
- Convert to INR (Rs) Rupees
- Convert to XAU (oz) gold (ounces)
- Convert from / to

### *Unit of measure conversion*

- Get / Set unit system
- Convert to Metric (meter, kilogram, liter, degC)
- Convert to Metric-Small (centimeter, gram, millimeter, degC)
- Convert to English (foot, pound, gallon, degF)
- Convert to English-Small (inch, ounce, ounce, degF)
- Convert from / to

### *Solver (inverse function solution)*

- x = $f^{-1}$ (y)    sol(y,$f$(x))

### *Script commands and conditionals*

- If, Then, Else
- Do (a script)
- Display / Clear Display (a result, text string, picture)
- Print (a model, report, graph or table)
- Load, Unload (a model)
- Select (a page, palette, simple table/graph)
- Start, Stop, Pause, Resume (a simulation, movie)
- Wake, Sleep, Shut Down (the calculator)
- At, After (a time, an interval)
- On (reference to system events)
- Loop / End Loop