I^{2}M - A Technical Perspective

By Dave Hillman, Vice President, Software Development
Template Software
August 17, 2007

Introduction

The Intelligent Information Manager (I^{2}M) is a powerful data-centered, decision support technology for managing access to complex, disparate data sources; reasoning about the data; and then applying the data to get things done. Originally designed to solve a Fortune 100 customer problem for integrating enterprise data and supporting schedules, it has grown into a multi-faceted tool with applications across many domains and customer needs.

There are classic problems in data management including the ability to universally manage any dataset, applying reasoning to a dataset on a consistent basis, and then applying that dataset to directly accomplish work. Each of these problems has challenged industry for a number of years resulting in singular solutions to each of these. Template Software has developed a strategy for dealing with all three. This strategy utilizes an integrated metadata model for managing data, applying rules to the data, and then integrating with various COTS (commercial off-the-shelf) software applications.

This white paper provides an overview of the I^{2}M technology suite including the background that it arose from, its data model, operational features, architecture, and how it can be applied to a variety of complex data and business problems.

Background

The basis of our efforts to develop this technology was relatively straightforward: a Fortune 50 customer with a classic problem: how do you exploit volumes of enterprise data to make real-time decisions about getting work accomplished on a day-to-day basis. Imagine keeping tracking of hundreds of thousands of parts only partially organized as vehicle subsystems such as the engine, fuel system, transmission, and even the body and molding (see Figure 1). Now correlate this information with product management listings that identify specific models and configurations of vehicles. Ultimately, the goal of this system was to use the data to produce and update the thousands of schedules associated with the parts, subsystems, and product listings. A key aspect of this problem included: the need to properly associate various parts to programs and subsystems; to determine which needed, or didn't require, schedules; and finally, keeping everything updated on an on-going basis.

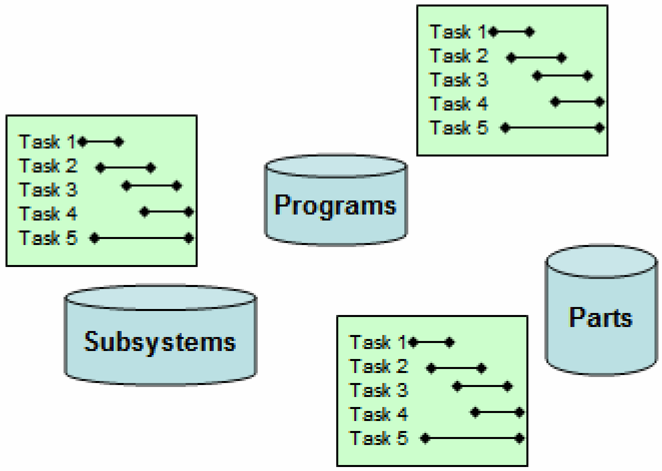

Figure 1. The problem: separate data and schedules.

In breaking down this problem, there were a number of complex issues to be resolved (see Figure 2). While all of the data is maintained by the same organization, it was very poorly integrated in that it was not trivial to simply link the products to the subsystems to the individual parts. Another problem was the ability to manage the schedule creation and management process beyond that of a very time and cost-intensive manual effort. Finally, a critical aspect required daily updates that would add to, modify, or delete the considerable amount of data imported at Day 0. These updates would drive the creation and updating of schedules. In addition, there is a longer term need to deal with additional data sets in the form of new databases that will need to be seamlessly integrated into the existing data environment.

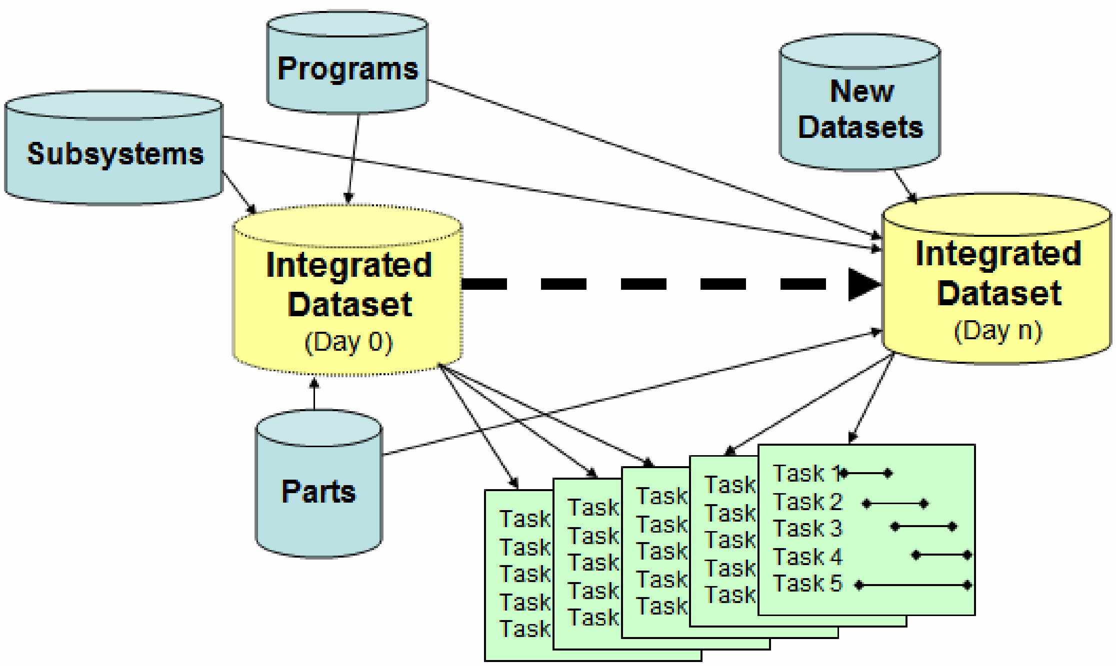

Figure 2. The desired solution: integrated data and schedules.

Our goal in developing a complex, integrated solution focused on the following:

- Developing an integrated data model that would enable rapid inclusion of all existing data, incorporation of on-going updates to that data (including deleting existing and adding new data), and the ability to add new data sources over time.
- Providing a set of tools to easily view and analyze the data in an enterprise environment.
- Enabling the application of business rules to consistently analyze the data; the result of this analysis would include adding new data “about” existing data; changing data, and triggering system processing.
- Integrating a COTS-based schedule management tool to create schedules associated to the data and managing information within those schedules.

The I^2M Solution

In the following sections, several key points of our technical solution will be discussed: the data model, data presentation, using linking and rules, COTS integration, architecture, and applications of the I^2M technology.

Data Model

A fundamental goal of the I^2M data model was the need to capture and use any data from any source. Therefore, the data modeling strategy employs a multi-layer approach that enables organization and inclusion of data (see figure 3). The organizational layer is referred to as a “program” while the data inclusion aspect is called a “component”. Programs have a many to one relationship to programs and to components and both are linked to supporting attributes that identify features about the program or data fields about the component. Components employ a many-to-many relationship to each other to accommodate any number of internally or externally defined relationships.

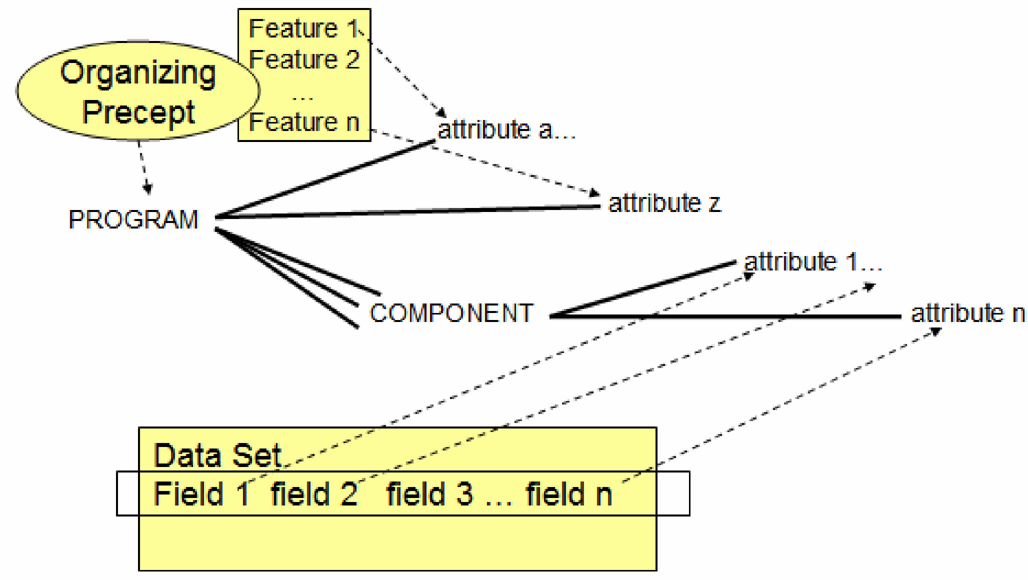

Figure 3. Relationship of programs to components to attributes.

The terminology for "programs" and "components" has been a useful paradigm; the term program offers a connotation of hierarchical relationships while components are typically the myriad parts that define complex entities. Neither term is intended to refer to physical analogs such as "nuts and bolts" or even programmatic references such as the "F-15 Eagle aircraft". They are merely conveniences that define the ability to organize multi-layer set of relationships across data domains.

One advantage of this data structure is the ability to handle and represent data in highly integrated domains or as loosely coupled data with few, if any, useful relationships. In initial efforts, this model was very useful in first assimilating data that was later linked to describe complex interrelationships. In subsequent data modeling efforts, this same approach was also used to present large amounts of unrelated data efficiently and consistently.

Data Presentation: A Two-pronged Approach

The hierarchical structure of the I^2M data model enables the user to present virtually any data in a very structured and consistent fashion. Furthermore, the data can be rapidly aggregated for presenting information via web-based interfaces.

We have developed two primary models for presenting data: a classic tabular view and a tree-based structure. Both views share a common tree-based view of the program structure linked to subordinate views of component data.

Figure 4 illustrates the table-based view of data that enables the user to select a program and view information (attributes) about the program and then drill down into the components and attributes that make up the program. The user can "drill-down" into subsequent layers of information and "drill-up" to see relationships. The advantage of this approach is that the user can start from one program and drill down to a very detailed level, for example, a component switch; and then drill back up to find out which other programs also have that switch.

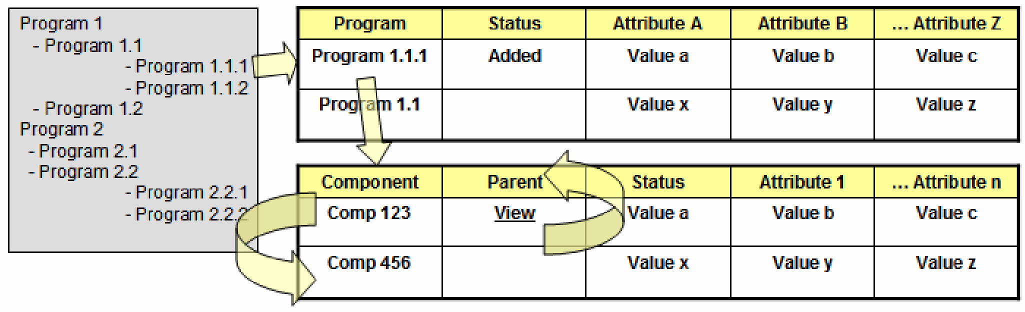

Figure 4. Table-based view and drill-down presentation of data.

Figure 5 illustrates the tree-based "meta" viewer which presents information in a hierarchical fashion. The data viewer is selectable allowing single and multiple levels of viewing to manage data clutter. As in the tabular view, the data can be drilled for access to detailed information.

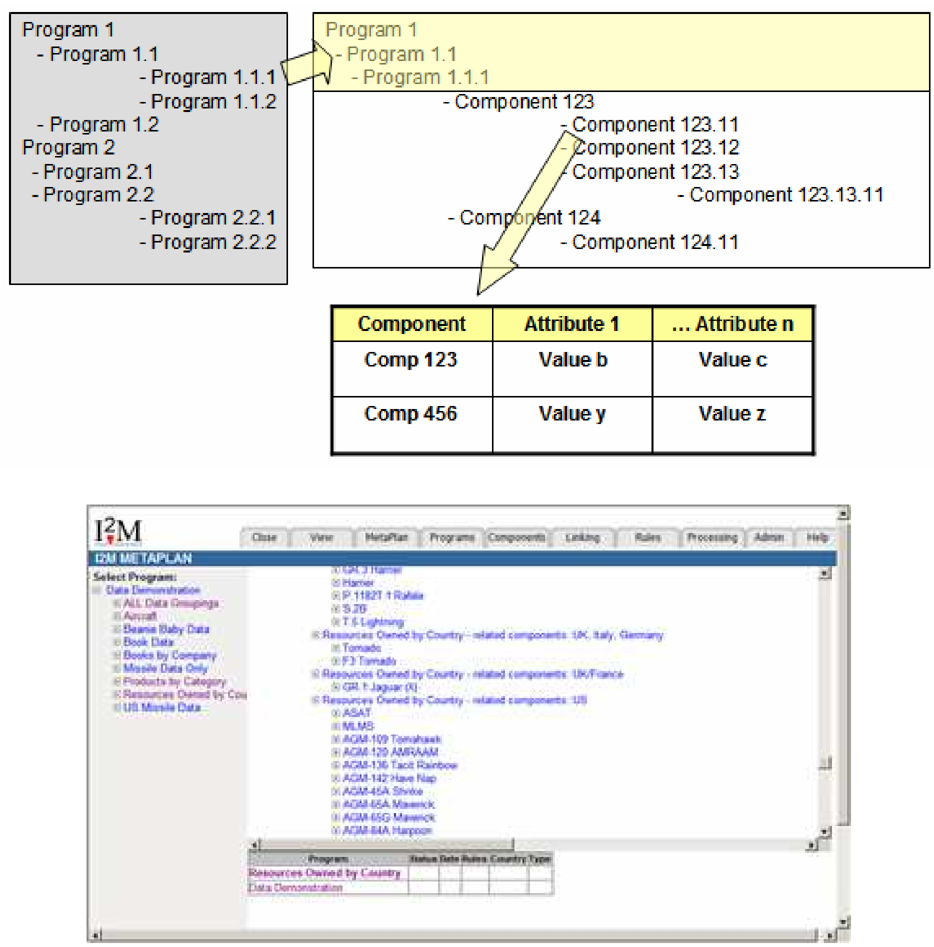

Figure 5. Tree-based "meta" data viewer; also a screen shot of the application.

Providing two different views of the same data has certain advantages. We have found some users prefer the tabular spreadsheet view over the tree-view and vice versa. Furthermore, the tree-view provides a graphical perspective of the complexity of data and subordinate relationships. Figure 9 provides a screenshot view of the I^{2}M data search capability. In this mode, a user can search component names and attributes by various values, with or without wildcards, and perform searches within searches. This facility is very effective for manually linking components to programs and to each other.

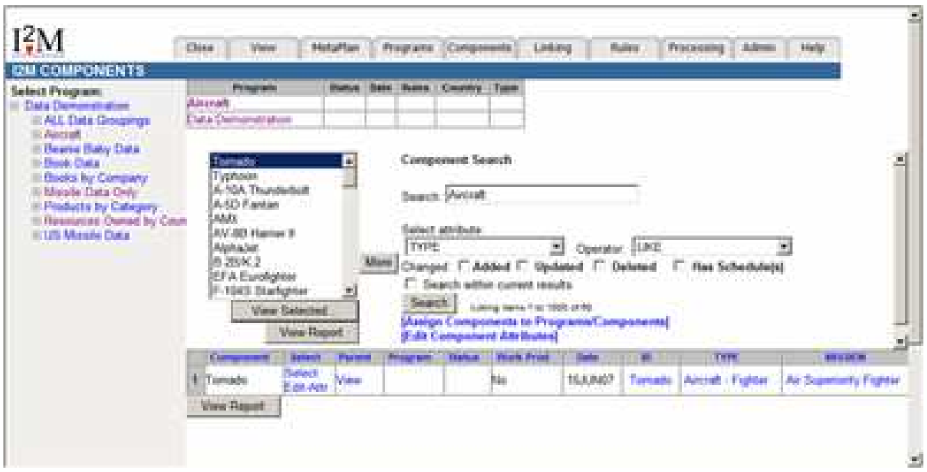

Figure 6. Search capability.

Linking and Rules: Intelligent Data Processing

I^{2}M is built upon two intelligent processing strategies: first, linking programs and components together to establish relationships and second, querying the data with if-then rules to make decisions. These two processing paradigms provide a powerful and flexible decision support capability for the product.

The linking process is based on correlating data at the program level to components via linking components created from data from the components themselves. For example, a component attribute could have ten values, stored in attributes, which can be linked to other components with related values. The process works like this:

- A component attribute exists with the following values across multiple components: “red”, “green”, “orange”, “yellow”, and “blue”.
- A set of new “link components” are created based on these values: “component: red”, “component: green”, etc.. These components are linked to a program.
- Linking “rules” are established in which attributes are assigned specific values: e.g., “attribute x > 10”, and “attribute b like ‘red’”, and so forth.
- The software then finds all components that have similar values and links them into the new “link components”.

This technique enables users to define new data structures based on the data that is available and to repeat that process as new data is added over time.

A rule-processing engine is a key part of the I^{2}M architecture. Rules, as shown in figure 7, are in the format of "IF-THEN" clauses: the if-clause enables the user to evaluate attribute values for programs or components while the then clause provides some action to be accomplished. Rules are organized at multiple levels: they can be assigned to users, organized as sets of rules for creation and editing, finally as they are assigned to programs. This enables rules to be selected and fired against specific programs as desired by users.

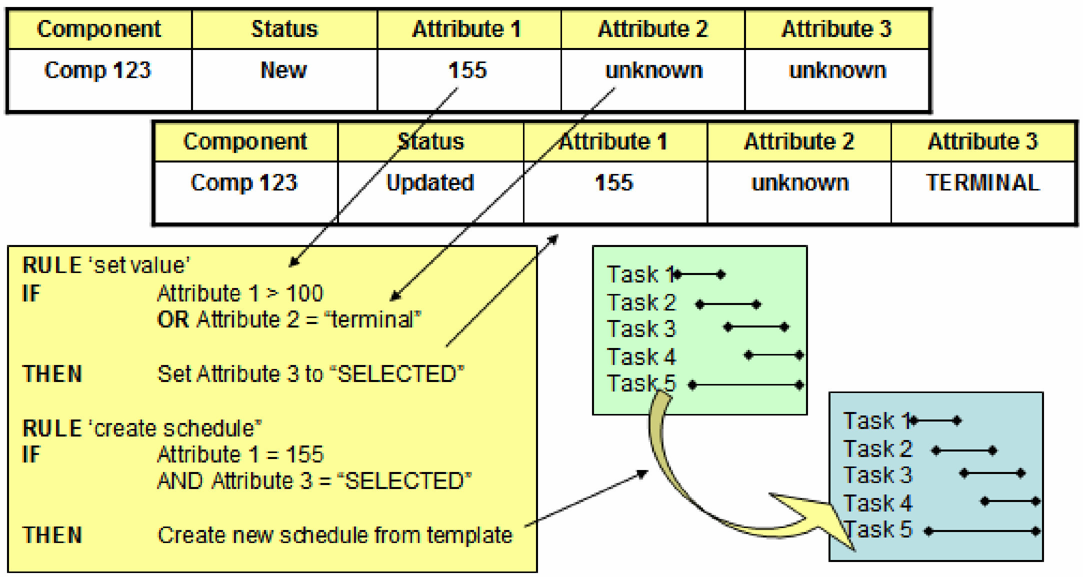

Figure 7. Using rules to evaluate data and trigger actions.

Actions, as "THEN" clauses, are triggered following evaluation of the "IF" clause in the rules. Rule actions offer a broad set of options for dealing with data:

- Monitoring the status of data by keeping track of data changes and aging.
- Set a value in a new or existing attribute.
- Trigger an action in another COTS application (see the next section).
- A specialized case of actions is based on the "true" rule, i.e., the "IF" clause is defaulted to "true", in which the rule action is automatically triggered; this can be used to set a consistent data value across a number of programs or component data.

This capability offers the potential for future growth of our tool set as new actions are incorporated.

COTS Integration

The ultimate goal of I^{2}M COTS integration is to correlate data to work products. A work product is a custom piece of information in an external application that is either derived directly from data in I^{2}M or is created as a result of reasoning about data in I^{2}M. This strategy has a key advantage in that end users are often very comfortable with existing COTS application work products, e.g., Excel spreadsheets or Gantt charts in a project management tool. The problem is that creating these work products from volumes of enterprise is time consuming, and furthermore complicated by keeping those work products updated from the source data as it changes.

I^2M was initially developed to support Actano's RPlan web-based collaborative project management software product. The support for RPlan was in two primary areas: creation and management of schedules and creation of links between tasks in schedules. Schedule creation is driven by two mechanisms: rule actions which prompt the user to create a schedule for each component and via a rule-driven mechanism where schedules are created in a batch mode. Schedule management involves periodically updating data from the I^2M source data to the schedules. The linking of tasks between schedules is accomplished on a program-level basis in which task elements and link criteria are merged to automatically create linkages.

The integration of RPlan is one example of the COTS integration potential of I^2M. Other project management software can be integrated as well as other products including spreadsheets and graphic visualization tools. Integration with COTS applications can be accomplished through database stored procedure calls, application programming interfaces (API), and web interfaces including SOAP and web services. This "open source" strategy covers most commercial applications and offers a wide variety of options for COTS integration with I^2M.

Architecture

Figure 8 provides a top-level view of the I^2M architecture. The database is currently implemented in Oracle with a variation under development in SQL Server. The primary software environment consists of two components:

1. I^2M graphical user interface (GUI) which provides the user experience.
2. DB (DataBase) Integrator which imports data from external data sources and manages batch processing activities (e.g., automatic schedule creation).

The GUI is actually implemented on two platforms: Java Server Pages (JSP) and Active Server Pages (ASP). The DB Integrator is implemented on Java with a new version being developed in C#, and can be run on the database or application server.

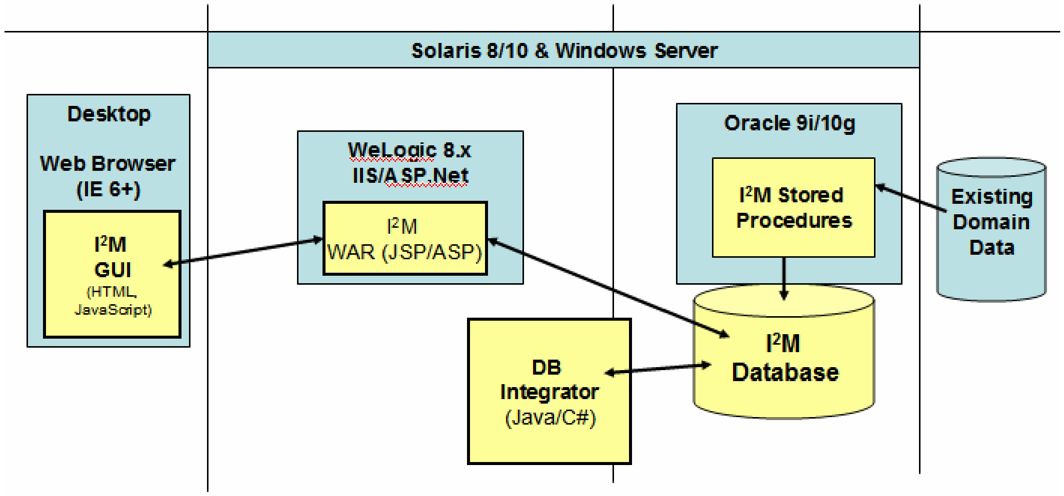

Figure 8. I²M Architecture

By design, the I²M architecture is intended to be cross-platform and flexible to adapt to the latest technology platforms. This approach guarantees that it will meet anticipated customer demands for many years to come. Another advantage of our cross-GUI platform initiative has been the impact on product testing and stability. Our core technology is built as an ASP-based solution with JSP code generated from the baseline. Primary unit and functional testing is performed on the ASP code and continues as JSP code is created. Whenever issues are identified, it is verified on both platforms to ensure stability on the database with code stability first established in ASP and then JSP. This has resulted in a significant reduction in trivial bugs and proven the overall stability and reliability of both code bases.

Putting It All Together

I²M set out to solve a complex data and COTS integration problem for a Fortune 50 client. It accomplished this with a flexible data integration facility, decision support tools, and integration with a COTS product that manages hundreds of thousands of data records and the creation and management of tens of thousands of schedules.

Figure 9 provides an overview of the end-to-end processing that includes:

- Importing data from external data sources into the I²M data base as components and attributes. This data can come from any data source and co-exist with any other related or unrelated data.
- Programs with attributes are created and linked to component data.
- Rules are created based on enterprise business processes.
- Work products are created and managed based on rules.

In most enterprises, data is always changing and a key feature of I²M is that data is never static. As the data changes, it can remain under constant analytic scrutiny by the rules and can be applied to create new work products as the data changes over time. Our initial efforts have shown that, in many cases, users discover a number of new ways to use data simply based on being able to view it from different perspectives.

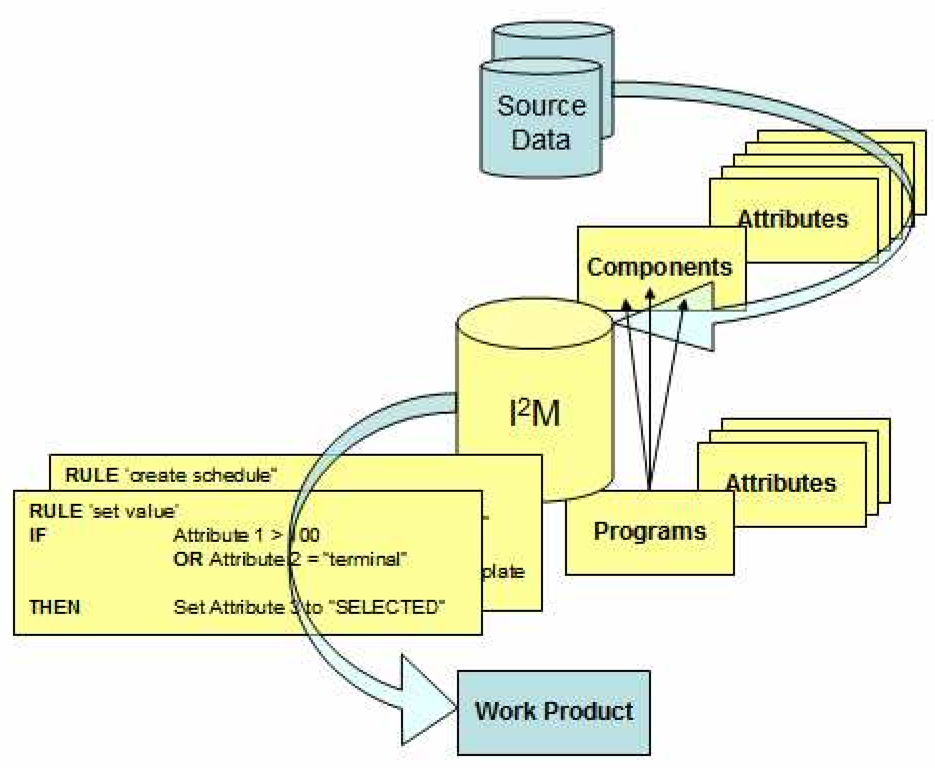

Figure 9. I^{2}M is an end-to-end data solution.

I^{2}M is ideally suited for applications in which large amounts of data require constant analytic scrutiny to generate work products. Our initial solution consisted of:

- The integration of multiple data sources describing hundreds of programs and hundreds of thousands of component parts.
- Program to component linking used to correlate parts to programs.
- Business rules describe which parts are critical to production processes and are critical to the design of the various programs.
- Rules are then applied to determine which components require schedules to be created and maintained.

I^{2}M is also suited for:

- Logistics applications to manage complex systems, subsystems, and parts.
- Manufacturing applications to track raw materials, production, packaging, and delivery of goods.
- Intelligence applications tracking people, places, things, and events.

The key in these applications is the need to apply complex business logic to large amounts of data in real time.

I^2M is the Intelligent Information Manager! For additional information, please contact Template Software.