SAPIENS Partnering for Success

Sapiens eMerge Technology White Paper

Table of Contents

1. Introduction

The Sapiens eMerge (abbreviated: eMerge) solution is a collaboration of technology, methodology and services, aimed at eliminating software development backlogs and solving complex integration issues by delivering rapid, mission critical enterprise business software solutions to customers.

This document discusses Sapiens eMerge technology only. For more information about the other Sapiens eMerge solution elements, see accompanying papers: Sapiens eMerge solution brief and Sapiens eMerge methodology brief.

The Sapiens eMerge technology platform is a core development and deployment environment that is designed to express business logic in a declarative manner with business rules, thereby providing a unified and open infrastructure for complete business software solutions. A key advantage of Sapiens eMerge is the ability to extend the productive life of existing legacy systems, while simultaneously providing a rapid migration path to new generation Internet and e-commerce technologies. The use of advanced, rapid application development technology allows enterprise-specific enhancements to be made in a shortened timeframe and with a vastly reduced maintenance burden when compared to other technologies.

1.1 Highlights

Sapiens eMerge technology:

- Is a rapid, declarative, rules-based business system development and deployment environment. An eMerge developer specifies WHAT and not HOW, thus eliminating procedural code. Using an English-like rule syntax, developers specify standard business logic only, and the rule-based engine automatically infers all of the non-standard operations;
- Boasts a central, dynamic, rule-base repository that contains all application definitions; Any application element defined in the repository, need only be changed once in order for all of its appearances in the application to be automatically updated;
- Includes an integrated GUI object modeling tool that accelerates business modeling and the rapid generation of all application definitions to be stored in the central repository;
- Contains an enterprise-scale transaction engine - capable of running heavy duty transaction loads, supporting thousands of simultaneous users;
- Is an integrated multi-tier, multi-platform and cross-database environment. Applications developed on one platform may be deployed on another without changes;
- Is truly globally-oriented. An application may be deployed in multiple languages – up to 99 – simultaneously at multiple locations;
- Completely separates the data, logic and presentation layers of an application; this provides scalability and flexibility to the developer;

- Includes a set of ready-made functions and object behavior attributes for the use of developers. These can be easily included at development time, using the comprehensive eMerge Development Workbench with its built-in business rules wizard;
- Is an open platform supporting Java, COM/COM+ and XML interoperability with eMerge objects, including IBM's WebSphere platform and other industry-standard technologies.
- Fully integrates back-office legacy applications to the Web.

All of the above factors combine to accelerate development by a conservative factor of 4:1 and reduce maintenance by at least 80%, in comparison to traditional software development environments.

1.2 Architecture

Sapiens eMerge incorporates an open multi-tier architecture that includes:

- Web browser clients
- Application servers:
  - interaction server
  - enterprise server
- Integration tools
- Adapters

The *eMerge* development environment, *eMerge Development Workbench*, includes a dynamic graphical modeling tool which drives system development from a business analysis perspective. Revisions are performed on the model, thus retaining consistency between the graphical model and the operating environment (as opposed to most popular modeling tools that enable one-way code generation only).

Rapid applications are built using *eMerge*'s unique rule-based "positive thinking" inference engine that infers the negative consequences of an operation. This means that there may be no need to design or code the negative logic in an application. The *eMerge Enterprise Server* will infer it from the positive rules that are entered. This represents a reduction in logic specification of 5:1, and dramatically enhances the quality of the delivered application since most 'bugs' arise in the negative logic.

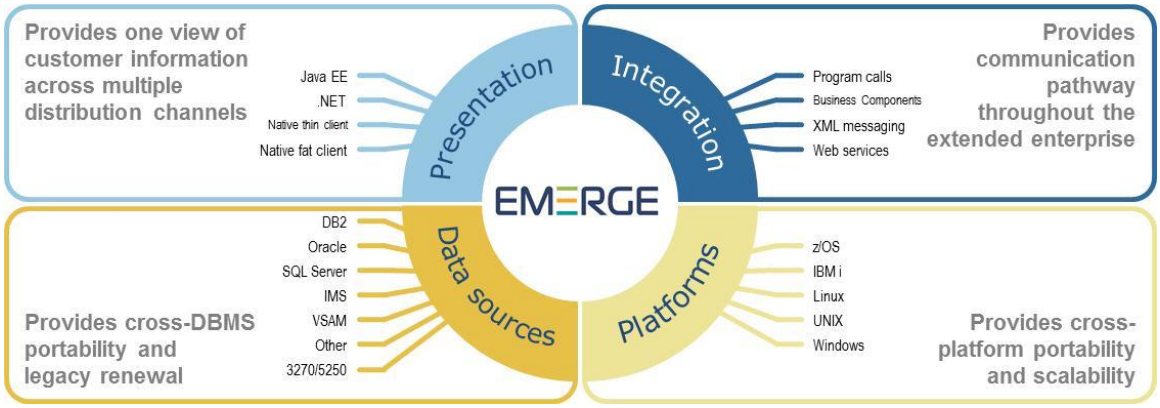
The *eMerge* servers can reside on one of several supported platforms. Thus, an application is portable. For example, if an application is developed under iSeries (OS/400), it can be run in CICS on a mainframe server, or on another Windows, Linux and UNIX servers. *Sapiens eMerge* handles the architectural differences between the different server platforms.

Via *eMerge Web-Services, XML* and *eMerge Business Components*, *eMerge* applications are open to all object-oriented development environments. Consequently, *eMerge* objects are accessible to .NET and Java code, thus leveraging existing applications and developer skillsets. Server and data source

independence allow the customer to leverage on past investments when migrating to more contemporary environments.

The following diagram shows user access to an e-business application via the Internet, intranet, MS-Windows, 3270 terminals and COM/EJB clients using *Sapiens eMerge*:

Figure 1: eMerge Operating Environment



1.3 Rapid Development Methodology

Sapiens' Rapid Architectured Application Development (RAAD) Methodology is an architectured approach to Rapid Application Development (RAD) that fully provides for the rapid and evolutionary development of the enterprise's e-business solution. Sapiens' field proven experience in RAD projects and in large-scale mission-critical system development has evolved and matured to create a methodology that collectively supports both RAD and traditional processes under one architectured methodology.

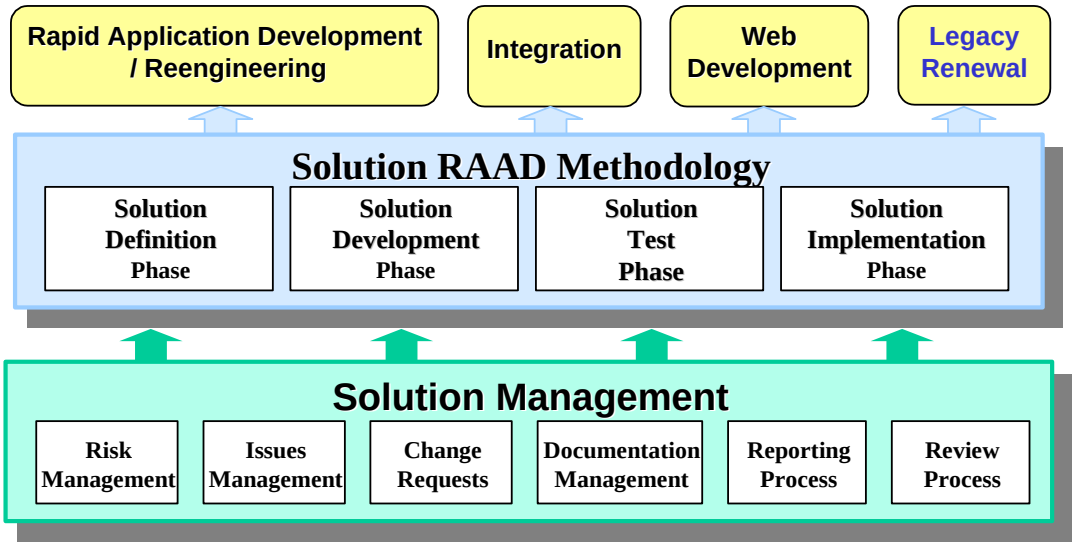
The iterative and evolutionary process of Sapiens' RAAD Methodology effectively accommodates solutions of varying scope, from small-scale solutions to large-scale enterprise-wide mission critical solutions. Practices are collectively applied to facilitate both traditional processes that warrant the structured and formal delivery of solution deliverables, together with RAD processes that support iterative and progressive prototyping for the early development of working components.

In order to successfully meet today's growing and demanding business and IT requirements, enterprises have begun and will continue to adapt their business, operations, and culture to web-based and e-business requirements. Enterprise's today are faced with the need to carry out application development and reengineering, application integration, business process automation, legacy renewal and creative front-end Web exposure to deliver complete and open system solutions with time-to-market and with value-to-market.

Sapiens' RAAD Methodology covers the whole process, from the initial capture of business requirements through to implementation of the final solution and incorporates ongoing management involvement and active user participation throughout all phases of solution Definition, Development, Test and Implementation. Supplementing the set of standards and procedures for all

solution phases an associate Solution Management System provides integrated procedures and tools to ensure that projects are delivered on time, within budget, and at the highest quality level.

Figure 1: eMerge RAAD Methodology



2. Benefits of Sapiens eMerge Technology

Sapiens' eMerge technology utilizes an advanced, multi-tiered development and deployment environment which provides the customer with the following benefits:

- Rapid Time To Market

Sapiens eMerge's primary benefit is faster time to market in e-business application development. Because development is much faster, new systems are delivered on – or ahead of – schedule. Technical detail is automated and there is no need to write code, so the development team can focus on business issues. Time to market is accelerated and the enterprise is better able to stay ahead of the competition.

- Openness

eMerge's openness not only facilitates the integration of new technologies; it also means ease of integration with legacy data and systems. eMerge business objects are available to other tools as true objects that can be manipulated according to the object-oriented rules. This enables developers in various COM and EJB environments to easily access eMerge's data and applications.

- Platform Independence

This powerful feature delivers full development and execution cross-platform portability. Businesses can quickly create a system on one platform and later deploy it on another, without additional investments.

eMerge runs on the following server and client platforms:

  - Servers
    - Windows
    - Linux
    - IBM iSeries (AS/400)
    - IBM mainframes running z/OS
  - Clients
    - Web browsers
    - Development Workbench (browser user interface for Windows clients)
    - 3270 screens

- Database Independence

User data can be managed by the eMerge Enterprise Server or can reside in an external data source. Databases currently supported are:

  - Linux: Oracle, DB2

- Windows: SQL-Server, Oracle
- DB2/400
- DB2 under z/OS
- Legacy databases: IMS/ESA, VSAM
- Scalability

Sapiens' eMerge technology provides scalability, facilitating progressive, cost-effective expansion of system capacity to accommodate increased volumes and system scope. eMerge automatically manages the technical complexities of handling multiple platforms, making it possible for applications to run unchanged on different computers. Migration and scale changes are easy – as business requirements shift, eMerge applications can move with them.

- Reliability

Reliability is built into every eMerge application. The predefined, high level objects have been tested and proven in thousands of core-business systems. Since they encapsulate both data definitions and processing logic, they can be easily re-used, thus simplifying application maintenance.

- Flexibility

eMerge's extraordinary flexibility makes it possible to thrive in today's rapidly changing business environment – to respond to opportunities presented by the Internet, intranets and extranets, acquisitions, regulatory changes, the demands of new business processes and reengineering of installed systems.

- Security

Sapiens eMerge maintains security all along the chain—from the Web client browser to the data sources via user and server authentication, privacy and authorization for secure handling of transactional operations.

Web client authentication, field level authorization, screen privacy levels, back-end integration with other security managers (e.g., ACF2, RACF or TSS) and event logging and tracking are all provided.

- Support for Multi-lingual Applications

With Sapiens eMerge, e-business applications are built that allow end users to work simultaneously in more than one national language within the same application. Drop-down lists, file names, error messages and data values can also be multi-lingual. For example, in an application that displays a list of city names, it may be defined to appear in English for English language users and in French for French language users. Sapiens eMerge also supports DBCS (double byte character set) languages (e.g., Japanese, Chinese and Korean).

- eMerge Projects Require Fewer Resources

eMerge's RAD development technology and methodology translates into lower staff resourcing requirements for projects. Users are able to realize business benefits sooner, with faster and higher returns on their technology investment. Post-production maintenance is also dramatically reduced by 80% in comparison with traditional 3 GL's such as Java, .NET and COBOL.

3. Sapiens eMerge Technology Components

A Sapiens customer asked to describe the key differentiator of Sapiens technology would likely reply that it is the extremely short distance from business rule conception to a fully operational enterprise solution.

Sapiens eMerge users are equipped with technology that is uniquely designed to address current and future mission–critical, e-business challenges. It allows enterprises to respond rapidly to growth and market changes while providing a system foundation that is scalable and easy to maintain.

To achieve this vision, Sapiens eMerge realizes bold and comprehensive technological goals, several of which are listed below:

- Fast and modern GUI rule-based development platform, which reduces time to market.
- Utilization of a RAAD (Rapid Architectured Application Development) analysis and design methodology.
- Distributed applications and database management systems, in order to ensure full serviceability of the system in various scenarios.
- Utilization of the Internet's well-established communication system.
- Interoperability with emerging development standards – COM, EJB; messaging standards – XML; and leading Web technologies – application servers, integration and marketplace solutions.

To successfully realize these goals, Sapiens eMerge utilizes the following technology components:

eMerge Enterprise Server

eMerge Enterprise Server is the core of Sapiens eMerge, a rules-based transaction engine that serves the other components of eMerge. eMerge Enterprise Server employs a revolutionary approach to the application development cycle by eliminating procedural code. To reduce development and maintenance time during the many phases of an application's life, the data objects, rules and user interface that make up the application are identified and defined, instead of writing lines of source code. These definitions are contained in the eMerge knowledgebase, a powerful object-oriented data dictionary in which all application definitions are stored.

For further details please refer to Chapter 4 eMerge Business Integrity Server.

eMerge Development Workbench

eMerge Development Workbench is the eMerge GUI development tool. eMerge Development Workbench provides a single integrated development environment from the analysis stage to the testing and maintenance of working applications. It features:

- A business rule-based iterative development approach instead of coding;
- A graphical modeling tool that ensures model correctness;
- An integrated knowledgebase which is constantly in sync with the model, thus simplifying reengineering tasks;

- An inference engine ("positive thinking") that reduces the application development phase by as much as 80%;
- An incredibly short distance from conception to realization of new applications.

For further details please refer to Chapter 5 eMerge Development Workbench.

eMerge i.way Interaction Server

By using eMerge i.way, eMerge applications are fully Web-enabled. By utilizing the Web's well-established public communication system, Sapiens can provide customers with dramatically improved cost-effectiveness of information transfer, increased reliability and accelerated development timetables.

This powerful tool accelerates development productivity and shortens implementation time. The creation of a common user interface serves internal, as well as external organizational needs. Using eMerge i.way serves the customer's main system objectives by providing the following benefits:

- High availability
- Scalability to accommodate future growth
- Reliability
- Recoverability
- Security
- Technological independence
- Multi-language capability

For further details please refer to Chapter 6 eMerge i.way Interaction Server.

eMerge Integration Tools

The Sapiens eMerge Integration Tools are:

- Legacy Adapter – enables non-intrusive legacy renewal by capturing the 3270/5250 data streams of legacy applications and treating them as data sources.

  The key benefit of using Legacy Adapter rather than a standard legacy renewal tool (such as one based on screen scraping technology), is that once the proper mapping is done, eMerge sees the legacy application as an object to which new business rules can be applied, just like any other eMerge object. New objects can also be added that interact with the legacy objects. This, in effect, enables business processes embedded in the legacy application to participate as equal peers in a complex integration scenario.
- DBMS Adapter - incorporates external data into eMerge applications.
- Custom Adapters – enable access to non-supported data sources.
- Business Components - eMerge Business Components provide eMerge with openness to .NET and Java applications. Through eMerge Business Components, .NET and Java application clients can access eMerge applications by exposing eMerge objects as

components with properties and behaviors (methods) that can be invoked. To theses applications, eMerge objects appear as classes that can be manipulated and integrated as application components. eMerge Business Components provide eMerge with the features needed to successfully integrate within emerging Web application serving platforms such as IBM's WebSphere, BEA's Weblogic and Microsoft's Enterprise Server.

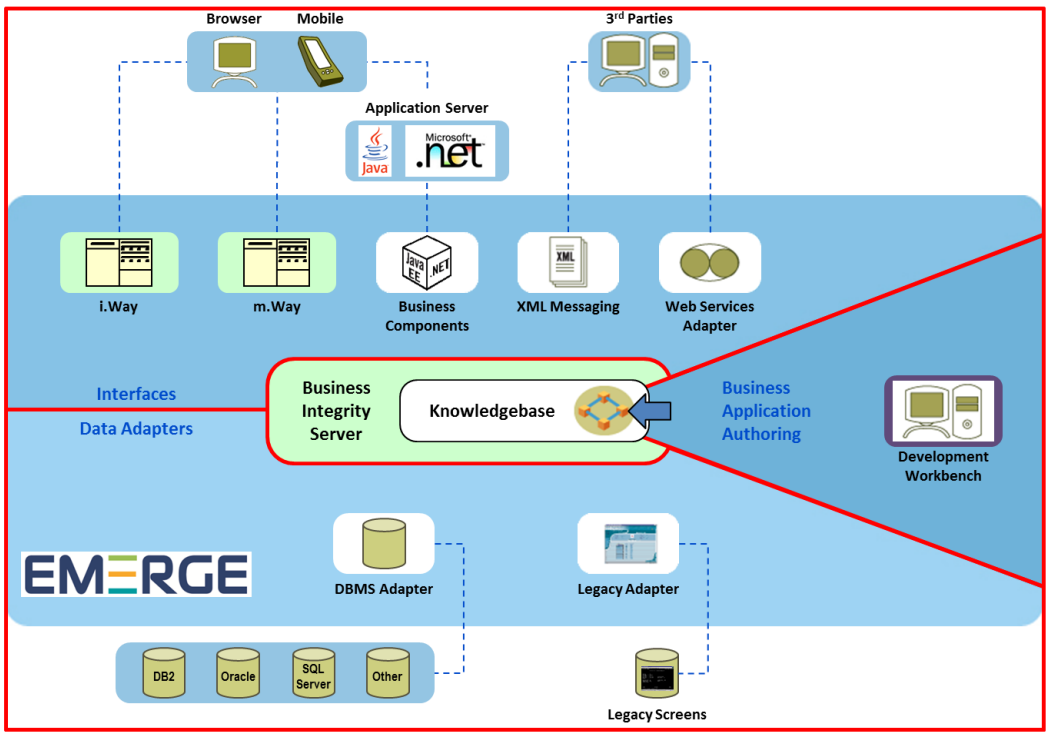
- MQSeries Adapter – supports message invocation from external applications to eMerge applications, with eMerge able to respond back to the invoking application.
- XML Adapter - a peer-to-peer environment (inbound and outbound), that serves cases where disparate, remote, and separately-controlled applications need be integrated. The typical and primary utilization of the Web-services and XML Adapters is in B2B and B2C integration scenarios.

For further details please refer to Chapter 7 eMerge Integration Tools.

4. eMerge Business Integrity Server

eMerge Business Integrity Server serves as Sapiens' rapid development environment and provides a fully integrated approach to systems analysis, design and implementation, and life cycle maintenance. The following diagram illustrates the eMerge Business Integrity Server architecture:

Figure 3 : eMerge Business Integrity Server Architecture



1

eMerge employs a revolutionary approach to the application development cycle by eliminating procedural code. To reduce development and maintenance time during the many phases of an application's life, the data objects, rules and user interface that make up the application are identified instead of writing lines of source code. As such, eMerge's development environment is not a programming language. By applying object orientation, the data and presentation objects are encapsulated together with the rules.

The application elements (i.e., the data objects, presentation objects, and rules), that govern the application's behavior and how it is presented to the user, are defined to eMerge Business Integrity Server via modern GUI modeling tools. From these definitions, eMerge generates a default application — a fully functional working model. The developer successively enhances and extends this working model, and reviews the results of changes as they are made. Once the model fulfills the customer's defined business requirements, it is moved into production.

The application elements describing the application are maintained within a comprehensive knowledgebase that is part of the eMerge Enterprise Server.

The eMerge knowledgebase is a powerful object-oriented database in which all application definitions are stored. The user data can reside in another DBMS, with the necessary interface provided by eMerge. eMerge provides comprehensive interfaces to the following DBMSs:

- Oracle
- SQL-Sever
- DB2/400
- DB2 (under z/OS)
- IMS/ESA (under z/OS)

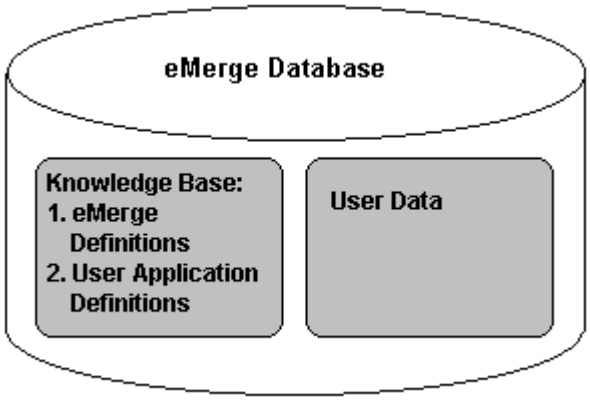
Custom Adapters are also provided so that interfaces to other DBMSs or VSAM data sources can be created (see Paragraph 7.3 below).

Included in the eMerge knowledgebase definitions are:

- The eMerge Business Integrity Server's definitions for the eMerge development environment
- User application definitions:
- data objects (descriptions of objects such as entities and associations)
- presentation objects (descriptions of data entry screens, menus and screen flows)
- rules (that control the data objects and the presentation objects)

The following diagram illustrates the structure of the eMerge development environment, which maintains the eMerge definitions of the development environment, the user application definitions and the user data in the same manner, using the same standards and procedures:

Figure 4: Structure of eMerge Development Environment



4.1 Working Environments

eMerge Business Integrity Server resides on a server and runs under a number of operating systems and working environments, as follows:

Servers	*Operating Systems*	*Working Environments*
Intel Servers	Windows	Windows Server
Intel Servers	Linux	Redhat
IBM iSeries	i/OS	i/OS
IBM zSeries	z/OS	CICS, IMS/DC, Batch

4.2 Working in More than One Language

eMerge systems allow the end user to work in several different national languages within the same application. Languages that require a double-byte character set are also supported (e.g., Chinese, Japanese).

Multilingual support extends beyond the basic level. For example, a drop-down list can appear in the same application in multiple languages, based on user preferences. Language directions (LTR, RTL) can be defined at the field level, freeing the user from manual direction changes within an *eMerge* application.

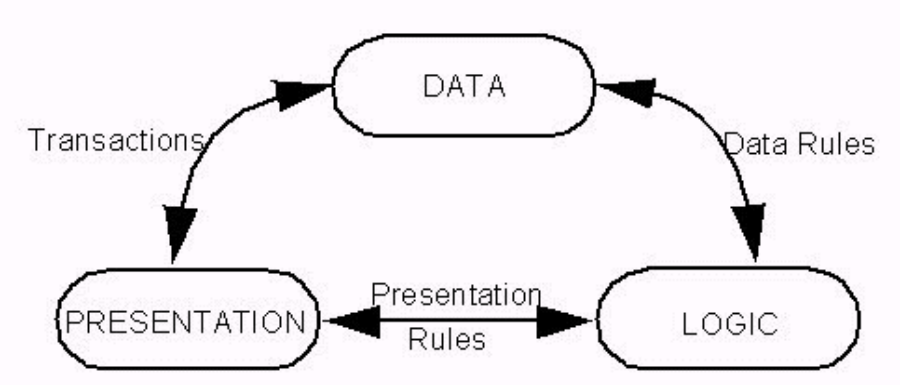
4.3 Developing an eMerge Application

An eMerge application comprises the following elements:

- Data
- Presentation
- Logic

The following diagram shows the interaction between these elements and the way they are linked. They are maintained within the knowledgebase and can be viewed and updated through *eMerge Development Workbench* (see Chapter 6 below).

Figure 2 : eMerge Application Building Blocks



Data

The Data component consists of both the user data and information about the structure of the application (the application data).

Presentation

The Presentation component involves the screens and reports, either online or batch, that allow the user to view and interact with the information.

Logic

The Logic component provides the application functionality. This includes both data functionality (e.g., what happens on entering a new employee) and presentation functionality (e.g., passing values from one screen to another).

4.4 Operations

The operation is the means by which the Presentation interacts with the Data. An operation is used to access a target object in the database for update or display purposes. All data is displayed through operations and all updates to the database are made through operations.

4.5 Rules

There are two classes of rules: Data Rules and Presentation Rules.

- Data Rules provide knowledgebase integrity. A data rule may be as simple as maintaining counters, or as complex as a full financial analysis.
- Presentation Rules control application flow. That is, how the data is displayed and how the user interacts with the application.

4.6 Development Utilizing eMerge's Tools

eMerge Development Workbench, *eMerge*'s development tool, provides a single integrated development environment with which to develop applications from analysis to the testing and maintenance of working applications. An application is modeled in *eMerge Development Workbench*, and based on user feedback and design considerations, continues to be modified through implementation until it is a fully functional application ready for release.

eMerge Development Workbench's Graphical User Interface (GUI) on the client PC is used to create a business model of the organization's structure and processes. Information regarding this model is also **stored** on the enterprise server where *eMerge* resides. Using the eRAD technology inherent in *eMerge* systems, this business model can be converted into a working application. For further details please refer to Chapter 5 below.

4.7 Data: The Constructor

The *eMerge* constructor provides the tools needed to create object hierarchies. These object hierarchies are created automatically when using *eMerge Development Workbench*. Objects are defined as constructors that are linked together to create the hierarchies. The *eMerge* constructor manages the conversion of definitions into the internal data structures used to generate a default application. The default application generated includes the definition of composites, presentation and logic in *eMerge*.

4.8 Presentation: Screens

The user interface (i.e., the online presentation of the application and its data to the end user) is managed by *eMerge*. *eMerge* manages the chaining of screens in the screen flow, the passing of data between forms, and the setting of display characteristics.

4.9 Logic: eMerge Features and Rules – Business Rules Language (BRL)

eMerge systems ensure basic data integrity and validity. Many complex logic requirements are managed by utilizing various built-in features. Where the complexity is such that the standard features do not cover the required logic, *eMerge* provides a special, English-like, nonprocedural, declarative language called Business Rules Language (BRL) for defining rules. A rule is only specified for one of the many possible cases (normally insertion of a new occurrence: the positive case). From this positively defined rule, *eMerge* infers the requirement needed for a particular change to the object (i.e., insert, delete, etc.). For example, in an inventory system, the quantity of an ordered product should be subtracted from its quantity-on-hand in the warehouse. This is the positive requirement that is defined. The negative implications, such as changing the quantity ordered, canceling an order, or changing the product ordered are handled automatically by *eMerge*.

4.10 Query (QUIX)

Requests for selected information (i.e., queries) are defined using an English-like language called QUIX. QUIX is used to extract information specific to a customer's requirements and then to output this information, together with any additional statistical information the customer requests. Results of the QUIX query can be viewed as an online report or can be printed in batch. In addition, the extracted information may be used for further processing by *eMerge* (e.g., updating only the queried data) or by an external tool (e.g., Excel).

4.11 Reporting

Reports can be created either online or in batch via QUIX (see above). In addition, for more complex reporting requirements, DBCOBOL or DBPL1 may be used. DBCOBOL is 4GL extensions of COBOL, respectively. They may be used to access and update the database in batch, as well as to produce reports. A DBCOBOL program can also be called online to display data on a screen.

4.12 Security

eMerge provides a security system for applications developed using *eMerge Development Workbench*. This system enables protection of data down to the level of specific individual field values.

Security for an application can be activated or deactivated at any stage. Whether application security is either deactivated or undefined, some simple security mechanisms are nevertheless available. These include the following:

- specifying the application databases that a user is allowed to access;
- specifying the terminals from which a user can access an application;
- restricting the user to a specific flow of application forms.

The system can also restrict user access for the retrieval or updating of data in any part of the database, including restriction by key ranges within the same constructor. The *eMerge* product, as an open system, can also interface to external security systems if required (e.g., ACF/II, RACF, TSS).

4.13 The Database Management System (DBMS)

eMerge contains a native Database Management System (DBMS), which is the default storage method used for both the *eMerge* knowledgebase and the user data. By choice, user data may be stored in an external database instead.

An application is created by storing object definitions within the *eMerge* knowledgebase. All objects are defined only once to the knowledgebase and can be used and reused in different parts of the application. Any use of the object refers back to the single definition provided. If the definition is changed, the change is immediately effective throughout the entire application.

eMerge supports a multiple application environment, in which many applications can access the same *eMerge* knowledgebase simultaneously. These applications may use the same logical files or different logical files within the *eMerge* knowledgebase.

4.14 Interfacing to an External DBMS

Through the Business Logic Processor (BLP), data is accessed natively or from external data sources, using the *eMerge* Integration Tools. The *eMerge* Integration Tools include *eMerge DBMS Adapter*s, which enable the *eMerge* user to incorporate non-*eMerge* data sources and databases into their applications. The *eMerge DBMS Adapter*s map VSAM, DB2, SQL-Server, Oracle, IMS or DL/I data structures into *eMerge* format.

For unsupported databases, the *eMerge* Custom Adapter transforms data into *eMerge* format.

All data access, whether via *eMerge* native DBMS, *eMerge DBMS Adapter*s or *eMerge* Custom Adapters, is handled by the BLP and is transparent to the rest of the *eMerge* system.

For further details about *eMerge* Integration Tools, please see Chapter 7 below.

4.15 Database Administration Support

eMerge provides DBA utilities for installation, maintenance, tuning, backup and change management for the database (including external databases, such as DB2, Oracle and SQL-Server) and for the environment (e.g., security, TP monitors, etc.).

5. eMerge Development Workbench

eMerge Development Workbench is the eMerge development tool. eMerge Development Workbench provides a single integrated development environment to develop applications, from the analysis stage to the testing and maintenance of working applications. Applications are developed using an iterative

development approach. This contrasts with customary serial ‘‘waterfall’’ approaches to application development.

Applications are modeled in eMerge Development Workbench and, based on user feedback and design considerations, continued modifications are performed until the model is ready for release. The customer’s users participate in the development process, not just during the early conceptual stages, but throughout the development life cycle.

eMerge Development Workbench incorporates a client/server architecture. A business model of the organization’s structure and processes is created using eMerge Development Workbench’s Graphical User Interface (GUI) on the client PC. This model is then stored on the eMerge Enterprise Server and can be subsequently converted to a working application. The working application runs on the server operating system and interfaces to the customer’s DBMS. The customer can specify to which DBMS it wishes to interface (e.g., DB2) as early as the analysis stage.

As soon as a portion of the business model has been created, down to the level of an individual object, the customer can automatically generate a default working application to test. It is also possible to run the model and the test application in separate windows, applying and testing revisions immediately. Both the model and the application definitions are stored together in the eMerge knowledgebase.

eMerge Development Workbench is used to create the following:

1. A Concept Model
2. A Business Rules Model
3. A Presentation Model

These models are defined below.

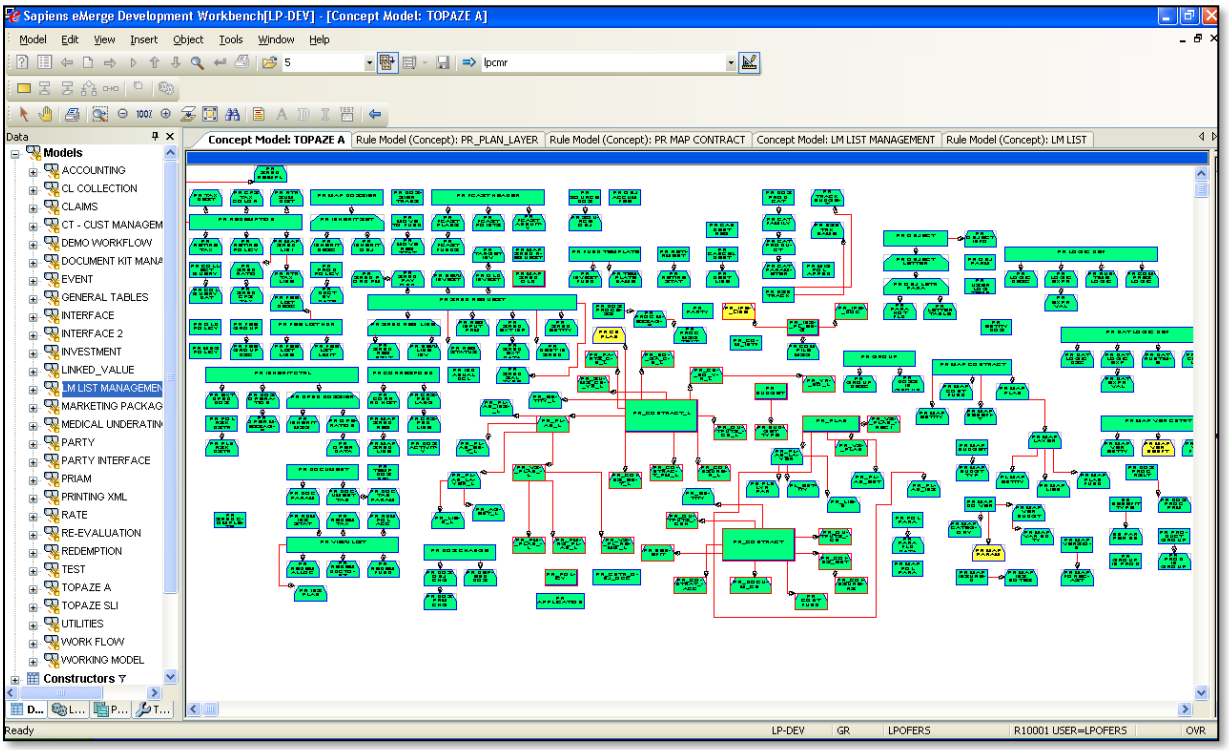
Concept Model

The data aspects of an organization are modeled via the Concept Model. The Concept Model is an extended version of a basic Entity-Relationship (E-R) diagram. In addition to modeling the entities that comprise the organization, and the associations that associate these entities, the Concept Model provides additional concept types to better mirror the real world. When using one of these additional concept types, defaults are set which are specific to the type of the concept. Concepts are also assigned default concept types that streamline the development process. Thus, the Concept Model results in the following benefits:

- Definitions specific to the type of the object are automatically assigned, ensuring accuracy and simplifying the Modeling process.
- The model is a better representation of the real world. As such, it is easier to read and to develop.

A Composite Model is the graphical display of the structure of an eMerge composite in a hierarchical (tree) format. The tree's root represents the root class of the composite and the dependent nodes represent the dependent classes as they are defined in the composite.

Figure 6 : Concept Model



Business Rules Model

The application logic is modeled via the Business Rules Model. The Rules Model can be used independently of a Concept Model or a Presentation Model. The Rules Model can also be used to review and extend existing processes and rules created in the past, as well as to create new processes and rules.

eMerge refers to a group of business rules as a ruleset. The Rules Model provides a graphic picture of the rulesets using eMerge Development Workbench's GUI on the client PC. The Rules Model is developed through controlled analysis, design and implementation stages.

A ruleset is attached to one of the following triggering objects:

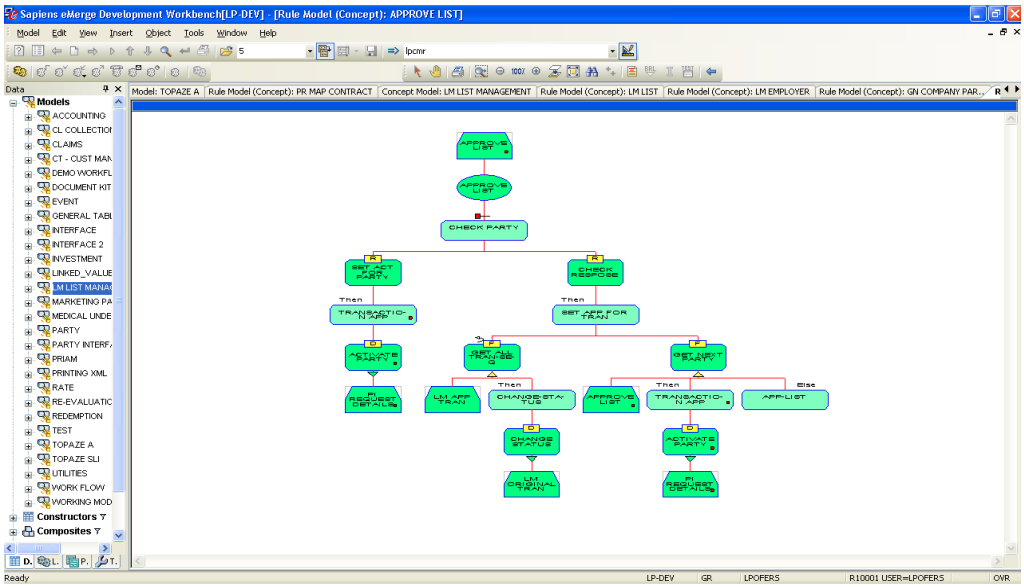
- data object
- concept
- constructor
- class in a composite
- presentation object
- function
- operation

- program
- query
- block in form

A triggering object can have more than one ruleset attached to it. Any change to the state of a triggering object (i.e., user action or data update) triggers the rulesets. When a ruleset is triggered, the rules in the ruleset are checked to determine relevance to the situation and, if so, they are executed. A rule is provided only for a positive event. The rule is automatically extrapolated to provide the rules needed for all the "negative" events that can occur for that case. For example, if a rule is written for the insertion of a configuration object, the negative cases of changing or deleting the configuration object are automatically inferred.

When a rule attached to a data object causes another object to be modified, the rules attached to the other object are automatically executed. This creates a ripple effect until every object that is involved has been updated based on its ruleset. The customer need only consider the rules attached to the object being directly handled, while eMerge manages the ripple effect.

Figure 7: Composite View (Business Rule)



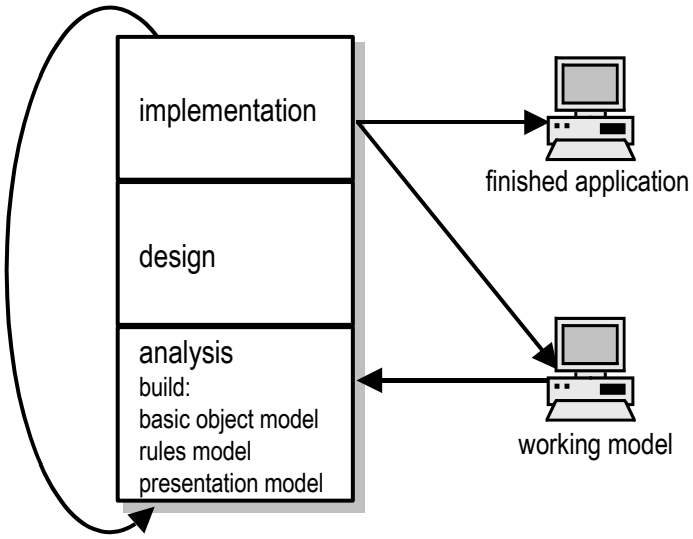
Presentation Model

The application flow is modeled by the developer. Functions needed to support the application flow of work and the Man-Machine Interface (MMI) are modeled via the Presentation Model. eMerge Development Workbench allows the customer to graphically decompose the functions in a tree structure, down to the level of the data (the concepts defined in the Concept Model).

5.1 The Object Model Working Environment

The Concept Model, the Rules Model and the Presentation Model are developed through controlled analysis, design and implementation stages. At each stage in the development, users can be shown the current working model and their feedback can be incorporated in the next iteration. When the user is satisfied, the working model becomes the finished application, ready for testing before going into production.

Figure 8: eMerge Application Development Flow



The modeled application includes the database specifications, the screen specifications, the screen flow, and the rules which govern how the data in the database behaves. No manual conversion process is necessary to apply all of these definitions.

eMerge Development Workbench incorporates a client/server architecture. Modeling is performed on a client workstation while definitions are stored in the eMerge knowledgebase on the eMerge Enterprise Server. As such, the following benefits are enjoyed:

- The server is always current with the latest definitions.
- More than one developer can work on the same application simultaneously.
- Objects are available to the entire organization and are reusable. Thus, a developer can incorporate objects defined by another developer (even if the object was defined for another application in the organization).
- Customers can take advantage of the Change Management feature of the eMerge Enterprise Server.

Each model can be viewed on two levels:

1. Width-level. A wide view of the model is provided and is presented graphically. This facilitates viewing many objects, together with their basic information, at a glance. The basic information

displayed includes the object names, the object types, the development stage of each object and the links that exist between the objects. The object type is indicated by the shape of the object symbol in the model. The development stage is indicated by the color of the object symbol in the model. The width-level view is not cluttered with specific information about individual objects. The viewer can focus on parts of the model or view the entire model on the screen.

2. Depth-level. The viewer can focus on one particular object and can see all of the detailed information about that particular object (e.g., its fields) via dialog boxes and eMerge Development Workbench screens.

5.2 The eMerge Development Workbench Development Stages

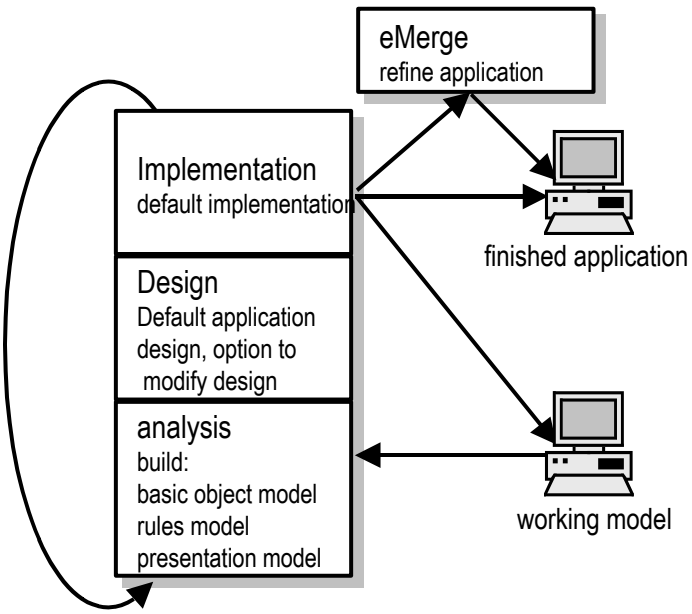
Work performed with the eMerge Development Workbench is divided into the following distinct development stages:

- Analysis
- Design
- Implementation

Each concept, rule or function is always in a clearly defined development stage. The development stage is represented in the model by the color and/or pattern of the symbol in the model, so that the development progression can be viewed at a glance. Each model can be developed independently. In a single model there can be concepts, rulesets or functions at each of the development stages.

All the models together provide an integrated working application. The following diagram presents the application development flow using eMerge Development Workbench.

Figure 3: Application Development using eMerge Development Workbench



Changes to a model can be made directly in the *eMerge Development Workbench*. This process leads to an iterative development approach, where the feedback received from the user is incorporated into the analysis and design. In addition, refinements can be made directly to the applications that are outside the scope of the *eMerge Development Workbench*.

5.3 Emerge – Rule Based Development Example

In a typical 3GL/4GL application, developers must code explicitly for every business case that may arise. This is not the case with *eMerge*, as shown in the following example:

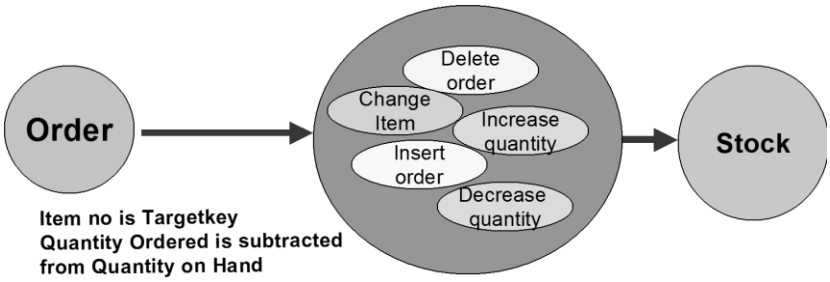
5.3.1 Order Processing Example

When a quantity of an item is ordered, the stock level must be adjusted accordingly (i.e., the *Quantity Ordered* must be subtracted from the *Quantity on Hand*). This is straightforward when inserting a new order line. However, other possible business scenarios must also be considered, such as:

- Increase the quantity ordered
- Decrease the quantity ordered
- Change the item
- Delete the order line
- Delete the order

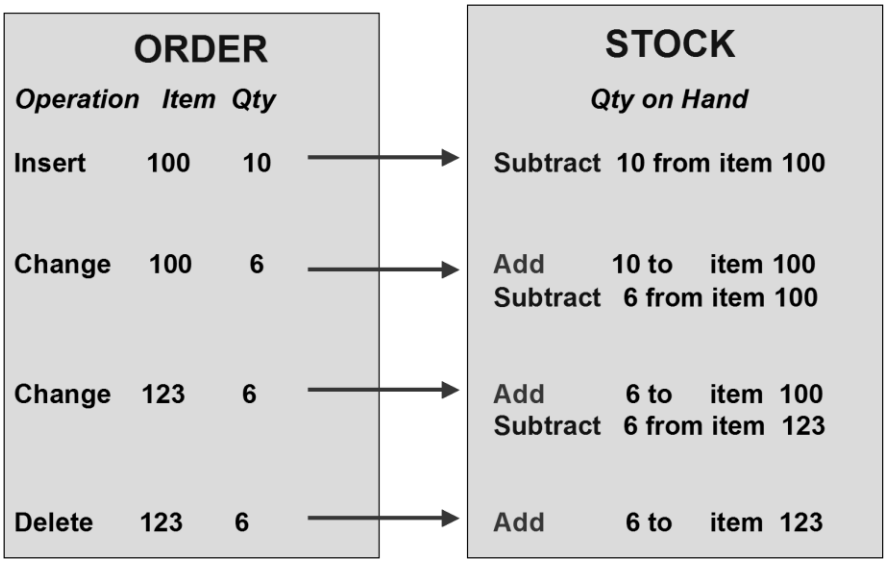
eMerge rules are always written for the positive case of insert. The change and delete cases will be automatically inferred by *eMerge*.

The *eMerge* rule for handling all of the above cases would be:



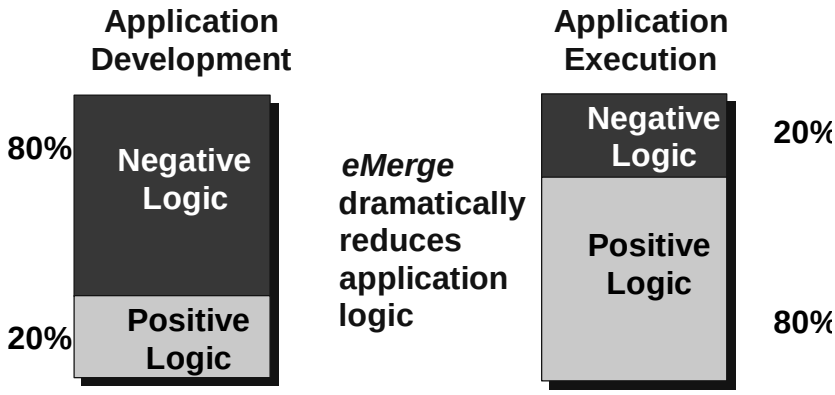
5.3.2 Positive Thinking Example

Using the above example of subtracting the Quantity Ordered from the Quantity on Hand, we can see the actions taken by *eMerge* to some of the different possible updates.



5.3.3 Effect of Positive Thinking

During application development, only 15-20% of the developer's time is spent analyzing, designing, coding and testing the positive logic in an application. This is the function that the user can easily describe to the developer (i.e., When I take a sales order for a product, I need to subtract the quantity ordered from the amount of that product in stock). The other 80-85% of the developer's time is taken up with the negative logic (i.e., What should be done if the order is deleted, or a product code in the order is substituted by another one, or one line of the order is deleted, or the quantity ordered is changed?).



When the application is running in production, 80-85% of the operations will exercise the positive logic — taking orders. Only 15-20% of the operations will exercise the code created to handle the negative consequences. In traditional application development platforms, this translates to poor payback for the developer's efforts. However, with *eMerge*, this "productivity cliff" is avoided.

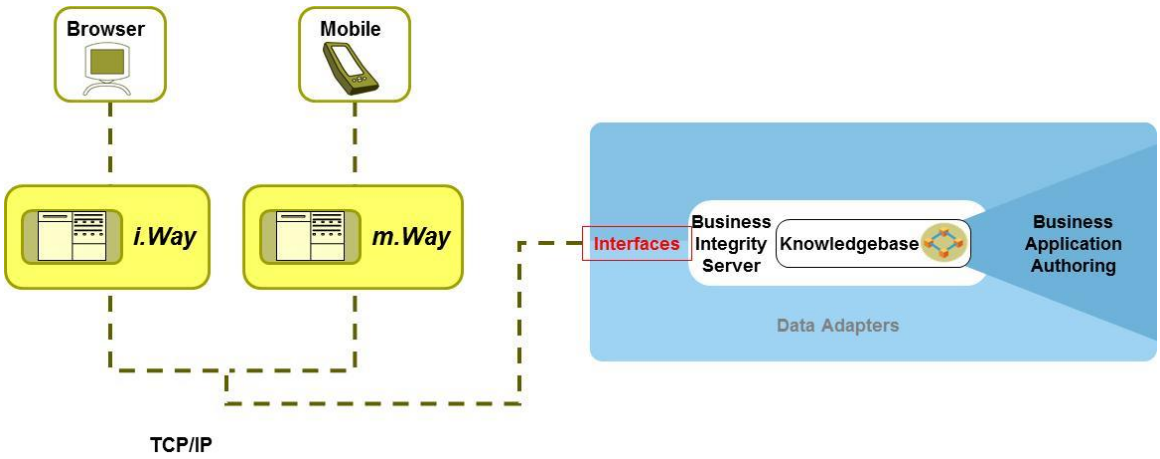
eMerge Enterprise Server contains an inference engine that can infer the negative consequences. This means that there may be no need to design or code the negative logic in an application. The *eMerge Enterprise Server* will infer it from the positive rules that are entered. This represents a reduction in logic specification of 5:1 in the development effort, and dramatically enhances the quality of the delivered application since most 'bugs' arise in the negative logic.

6. eMerge i.way Interaction Server

6.1 Introduction

Sapiens eMerge i.way Interaction Server (“*eMerge i.way”*) enables customers to access their applications via an Inter/intra/extra-net Web site. Users worldwide can access *eMerge* applications without first setting up a communication link, or installing software. The customer determines how much of his/her application is accessed and who is granted access.

Figure 4: Inter/intra/extranet user access to applications



6.2 Thin Client/Thick Server Architecture

eMerge i.way Interaction Server provides a thin client/thick server architecture to *eMerge* users. A PC running a browser or Windows is the client, whereas the application executes on an enterprise-wide server.

The thin client/thick server architecture provides the *eMerge* applications with the following benefits:

- The ***power, high volume*** and ***secure data storage*** of the server.
- ***Improved response time*** due to less interaction with the server (e.g., some validity checks are done at the client site).
- ***GUI and WUI (Web User Interface) technology*** with a ***user-friendly environment*** that includes clear and uncluttered screens, graphical presentation, pull-down menus, context-dependent help, multiple sessions and multiple windows per session, as well as integration with other workstation-based tools.

eMerge i.way replicates a subset of the *eMerge* knowledgebase that is stored on the server, to create a *local knowledgebase* on the PC. Local knowledgebase integrity is maintained by the definitions in the server knowledgebase. Included in the local knowledgebase are all the definitions necessary to support flow control and screen management. This means that communication with the server is only necessary when the database must be accessed for data. The database need not be accessed for definitions. **This leads to faster response times for developers and end users.**

6.3 eMerge i.way Capabilities

eMerge i.way makes it possible to plug in a Web browser and use the Internet as a client for legacy applications, as well as to develop exciting new products or services that use the Internet, intranet and extranets.

eMerge i.way is used in conjunction with the *eMerge Development Workbench* to generate GUI presentation for supported Web browsers like Microsoft's Internet Explorer and Netscape's Communicator.

eMerge i.way's open architecture enables *eMerge Business Integrity Server*, *eMerge i.way* and even the Web clients to inter-operate with external applications by:

- exchanging data;
- invoking external applications;
- invoking functionality on external applications;
- enabling standard-based environments to manipulate objects on the *eMerge Enterprise Server.*

eMerge i.way application forms are automatically generated as DHTML pages enriched with Java applets for display on the Web client.

An *eMerge i.way* application and another application can be within the same browser instance (but in another frame) or within another browser instance.

Users can dynamically modify the appearance of, and control interaction within application forms by defining external components. Some uses are:

- enabling and disabling interactive elements within an application form;
- controlled animations;
- showing graphs rather than constructors of data;
- special controls for application forms.

6.4 Utilizing eMerge i.way

eMerge i.way may be incorporated into a customer's business environment in several ways. For example:

- Online quotes for insurance policies can be provided through an insurance company's Web site, as a convenience to existing policy holders, to attract new customers, or to provide independent agents with comparative rates.
- Off-site sales personnel can submit or retrieve data via the company extranet, and keep their corporate database fully synchronized.
- Customers can place online orders through their company extranet. The system can verify data that is entered and can check the order against the inventory files of the host computer. This allows the company to keep the quantities on hand in sync with the order processing system.
- Loan officers working at a client site can search the bank's database and warehoused industry statistics to retrieve and implement information necessary for mission-critical decision-making.

6.5 Positioning eMerge i.way within the eMerge Enterprise Server Environment

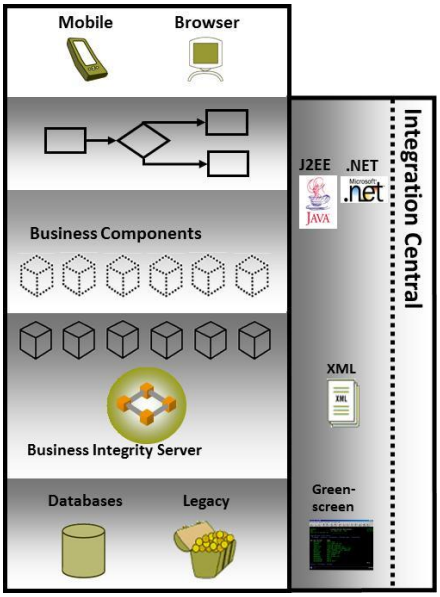
eMerge Enterprise Server is designed around the concept of a "thin-client/thick-server". Application behavior is encapsulated within business objects that reside on the Enterprise Server, while client computers handle presentation, user interaction, local validations, and workflow. The thin-client/ thick-server architecture supports a variety of client platforms:

- PCs running popular web browsers (internet Explorer, Chrome, Firefox etc.)
- Various mobile smartphones
- 3270/5250 terminals

The Sapiens *eMerge i.way* security mechanism enables the developer to restrict or grant entry to customer applications or data, as deemed appropriate. Thus, customers can use their legacy applications in their default Web format or extend them to take further advantage of the Web environment.

The following diagram presents the relationship between *eMerge Enterprise Server* clients and servers:

Figure 5 : eMerge Servers and Clients



6.6 *eMerge i.way* and Application Development

eMerge provides multiple client types with the following major advantages:

- A single developer viewpoint for all client platforms
- A single point of control for application maintenance

These generate substantial savings on investments when migrating from one client to another.

eMerge i.way provides for the evolutionary—rather than revolutionary—migration of *eMerge* users onto the Inter/intra/extranet, since the *eMerge* application development targeted at the Inter/intra/extranet and traditional client/server development does not utilize separate processes. Instead, the same development concepts and tools are used for both.

eMerge Enterprise Server's knowledgebase maintains all types of application objects, including objects for the structure and flow of presentations. The same business logic, presentation structure and flow may be used for all client types. Presentation structure and flow are manifested differently on various types of client machines, depending on the capabilities of the underlying user environment.

eMerge i.way provides effective tools, which comply with existing and emerging standards, to develop and deploy state-of-the-art Inter/intra/extranet applications. *eMerge* application forms are automatically generated via *eMerge i.way* as HTML pages for display on the Web client. This facilitates a multiple-client type of architecture.

eMerge application components can be defined to enable the following:

- enhance the appearance and functionality of a form
- inter-operate with other applications

An application can be enhanced by defining components that embed Web-formatting presentation into the application. Components enable image, lists or graphs to be shown rather than constructors of data.

eMerge i.way inter-operates with external applications and components, both Web-based and desktop, by defining them as components within an *eMerge* application. For example, data can be transferred from an *eMerge Enterprise Server* form to another application while both are open on a desktop. *eMerge i.way* also provides for URLs to external resources.

An HTML page may be composed of *eMerge i.way* frames side-by-side with frames owned by another application. Both applications can exchange messages at the Web browser client since data can be retrieved from other page frames and *eMerge i.way* can send external Java applets to the Web browser clients.

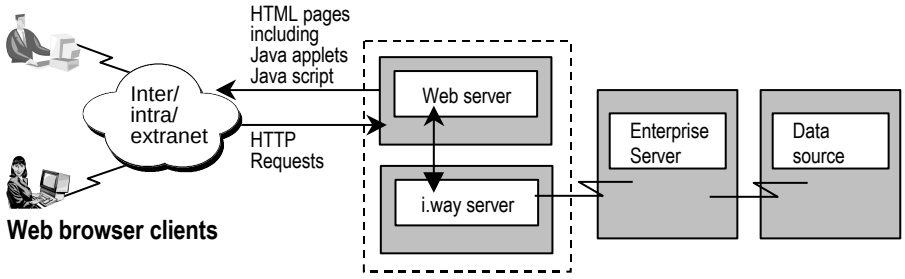
6.7 eMerge i.way Deployment Architecture

eMerge i.way is part of a multi-tier deployment architecture that includes the following components:

- Web browser client (such as MS-Internet Explorer)
- HTTP server (Microsoft IIS)
- *eMerge i.way*
- *eMerge Enterprise Server*
- Data source (e.g., internal or external DBMS)

The following diagram depicts the *eMerge i.way* multi-tier architecture:

Figure 6: eMerge i.way multi-tier architecture



6.8 Web Browser Client

Anyone with Internet access, or access to the customer's intra/extranet, and with a supported Web browser installed on his/her computer, can access a customer's web site, as well as those parts of the customer's application that the customer makes available.

Sapiens *eMerge i.way* sends HTML pages including Java Script and Java applets to handle the following events on the client side, including:

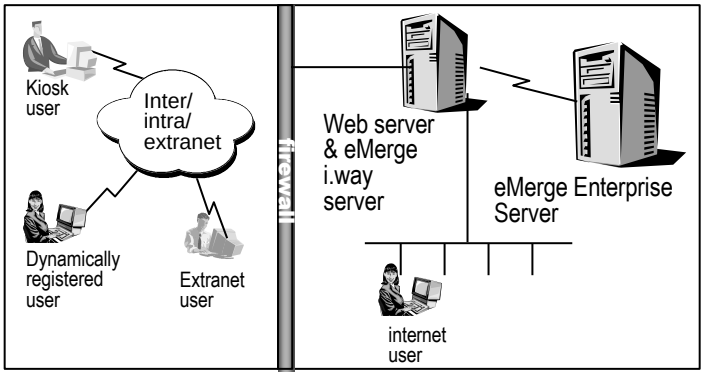
- Navigation assistance;
- HTML page rendering and layout according to *eMerge* knowledgebase definitions;
- User-events processing;

- Local validations;
- Enhancing the look-and-feel by adding GUI capabilities not yet supported by HTML.

The Web client can be one of several user types:

- "***Kiosk"-type user*** – For example, a shopper placing an online order.
- ***Dynamically registered user*** – For example, a user of an online application that requires each user to have a dynamically assigned name and password.
- ***Extranet user*** - For example, a customer can access a wholesale warehouse and order whatever s/he needs. In order to do so, the customer is assigned a specific security level.
- ***Intranet user*** - For example, an employee who accesses the company Web site for information, where the Web site is only accessible to those inside the company firewall.

Figure 7: eMerge Web Client Access Options



6.9 eMerge i.way and Web Server Interaction

A Web server such as Microsoft's IIS, receives all messages from the Web client and forwards the messages to *eMerge i.way*. In turn, *eMerge i.way* sends the resulting messages on to *eMerge Enterprise Server*. The process also works in reverse. That is, messages from *eMerge Enterprise Server* are sent from *eMerge Business Integrity Server* to *eMerge i.way*. In turn, *eMerge i.way* sends the resulting messages via the Web server to the Web client.

Communicating with Web server software (on the same machine), *eMerge i.way* (i.e., the Web Enterprise Server) maintains full status and context information for each Web client currently accessing the Web site. This is true even when the Web clients are concurrently running multiple applications.

eMerge i.way uses an HTML engine with extended functionality based on Java Script and Java applets. HTML pages are generated by *eMerge i.way* from existing *eMerge Enterprise Server* application definitions.

Back-end application definitions can be enhanced by embedding Web-specific objects. For example, background textures can be defined and the display of buttons and menus can be controlled with URL-like displays. *eMerge i.way* understands the semantics of Internet-related objects such as URLs and image maps. Applications can seamlessly integrate Internet-related technologies with existing *eMerge* concepts.

7. eMerge Integration Tools

7.1 eMerge Legacy Adapter

The majority of new e-business applications require access to systems and subsystems - data, business logic and workflow - that are currently managed by mainframe-based applications. This challenge is generally known as "Legacy Renewal".

eMerge Legacy Adapter ("*Legacy Adapter*") meets the requirements for full legacy renewal. Its development environment automates much of the process of building an interface between the legacy application and an application that meets today's e-business requirements. Its runtime environment is robust and efficient, and is transparent to the user of the e-business application. *Legacy Adapter* avoids the disadvantages of other proposed solutions by providing a modern, Web-enabled interface for the data and logic of legacy applications.

Legacy Adapter is based on an application that communicates with the legacy application as a 3270 or 5250 terminal. The aim is not to mimic the legacy application, but to provide a modern application that uses the legacy application screens as an external data source. Thus, instead of modifying the legacy application, it can be accessed through the existing 3270 or 5250 screens interface. We call this non-intrusive back-end integration.

Legacy Adapter's method of non-intrusive, back-end integration is two-fold, consisting of Physical Integration and Business Process Integration, as follows:

- **Physical Integration -** Typically a screen scraper wraps one screen at a time and does not address flow management and session management.

Legacy Adapter provides for more efficient integration by using application wrapping. Application wrapping includes:

- Flow management

  Legacy Adapter addresses the problem of flow management through several unique features. What to the end user is a single flow, can correspond to multiple flows in *Legacy Adapter*. Moreover, a single flow can branch depending on application data. *Legacy Adapter* provides a development environment that enables flow design without programming.

- Session management

  Session management is addressed by providing several session models. A session can correspond to a single flow, or multiple flows. If the session corresponds to multiple

flows, the Sapiens *Legacy Adapter* automatically navigates from the end of one flow to the beginning of the next. In addition, when an end-user session corresponds to several legacy sessions, *Legacy Adapter* provides an elegant solution for catching errors as they occur in order to take immediate corrective action.

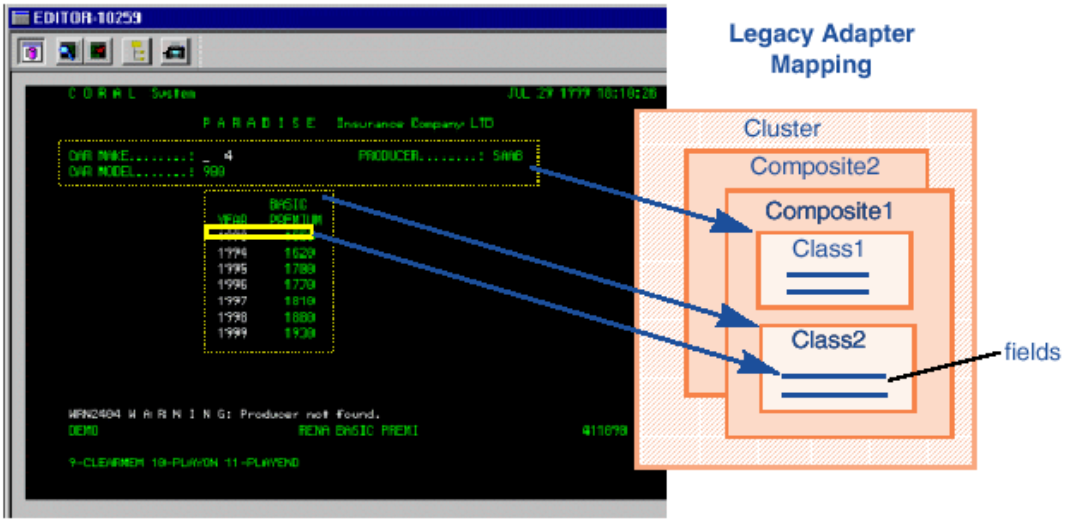
  - Many-to-many associations between screens and high-level business objects.

- **Business Process Integration -** With *Legacy Adapter*, the developer works with 3270 or 5250 screens as standard data source objects. There is no difference between "real" data sources (those residing in a DBMS) and 3270/5250 screen data sources. Developers can define business rules in the same way for all objects in the system—regardless of their originating source.

Events can be produced by Legacy Adapter in response to actions, that occur on a 3270/5250 screen. Legacy Adapter can also produce events in response to changes in the state of other legacy data sources, such as VSAM and DB2. The events are exposed to the enterprise using a unified object-event model.

How is it done?

Development



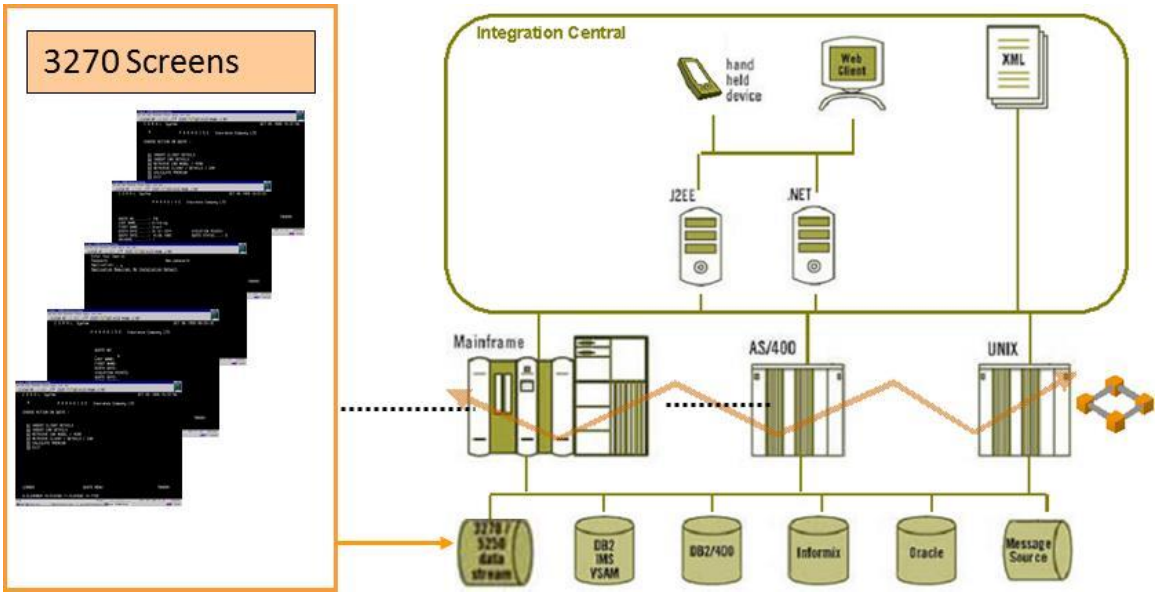
At development time, data appearing on legacy screens are mapped into an object-oriented data hierarchy structure composed of clusters, composites, classes and fields. Legacy screen flows are also captured.

The object-oriented data hierarchy is stored in the knowledgebase in eMerge Enterprise Server. The knowledgebase is available to eMerge Development Workbench to be incorporated into new web compatible forms. Business logic, components, and new data fields can be added to the new forms.

Runtime

At runtime, data (from the legacy application) used in the new presentation forms is stored in the legacy application database. Using eMerge i.way Interaction Server, Internet access to the new forms is made possible via any web browser from any location in the world.

Figure 8: Accessing Legacy Logic as Data



Legacy Adapter's development environment is the *eMerge Development Workbench*, which provides a point-and-click Graphical User Interface with the following modes of operation:

Emulation Mode

In Emulation Mode the workbench emulates a 3270 or 5250 terminal, which enables the user to run a legacy application.

Capture Mode

In Capture Mode, the user runs the legacy application and in so doing, captures the screens and logic flow which implement the functionality of *Legacy Adapter*.

Edit Mode

In Edit Mode, the user identifies screens, provides for error handling, and mapping the legacy images and flow to the data model.

Test Mode

In Test Mode, users may test the functionality of *Legacy Adapter* against the real application.

Once the functionality of *Legacy Adapter* is designed and tested, it can be extended by incorporating it into an overall data model, by adding business rules, business components, and Web enabling.

7.2 eMerge DBMS Adapter

eMerge DBMS Adapter ("DBMS Adapter") enables application developers to incorporate external data into eMerge applications. External data resides in a database maintained by one of the supported DBMSs listed in "Supported DBMSs" below. DBMS Adapter makes external data appear as native eMerge data, so that access to the external data is transparent to both the application and the end user. Thus, the full strength of eMerge can be applied to external data.

DBMS Adapter fetches data from an external database, transforms it according to a map, and holds it in memory as an internal eMerge composite. This internal composite is directly available to the eMerge user, and is indistinguishable from a composite fetched from the eMerge native database. The data can be used with QUIX, BRL, DBCOBOL, DBPL1 and all other eMerge application tools.

Once the data has been processed, it can be stored back into the external database. The external database can be configured as read/write or as read only. An external database that has been accessed by eMerge via DBMS Adapter can still be maintained by the external DBMS.

DBMS Adapter provides the following capabilities:

- Instead of restructuring existing data to use eMerge, the DBMS Adapter can be used to define the connections between eMerge and the data.
- Programming is rarely required since much of the process is automated. However, familiarity with eMerge and with the relevant external data structures is necessary.
- An application can be tested on a native database with the view of ultimately running the application on an external database.

7.3 eMerge Custom Adapter

An *eMerge* application can access non-supported data sources via *eMerge Custom Adapter ("Custom Adapter")*. This is accomplished via a file-level exit - a user-written mapping program that replaces the *eMerge*-supplied adapter mapping mechanism.

A developer can write a *Custom Adapter* to enable *eMerge* to access non-supported data sources. These are data sources where the composite structure or access method are not standard, or databases where the data structure or DBMS is not supported by the *eMerge DBMS Adapter*. The *Custom Adapter* program accesses the data source and converts its composites to internal *eMerge* composites and vice versa. The *Custom Adapter* program may be utilized in the following environments:

- **z/OS, i/OS**

  The *Custom Adapter* can be written in COBOL.

- **Windows, Linux, Unix**

  In these environments the *Custom Adapter* can be written in Java or .NET.

The *Custom Adapter* program must identify errors and report them to *eMerge*. It must allow *eMerge* to handle rollback and recovery in case of error.

8. eMerge – Glossary

eMerge Enterprise Server

eMerge Enterprise Server is the *eMerge* multi-tier development and runtime environment.

eMerge knowledgebase

The application with all its components is stored as part of the knowledgebase. The DBMS of the knowledgebase is called "native database". Application data can be stored in the native database or in another database such as Oracle, SQL Server, DB2 and Informix. The database structure is always stored in the native database.

eMerge Development Workbench

eMerge Development Workbench is the graphical modeling module of *eMerge* that is used to model, define, test and modify objects, attributes and rules covering the entire life cycle, in a workgroup environment.

eMerge i.way Interaction Server

eMerge i.way is a workflow and presentation environment that Web-enables *eMerge* screens by generating DHTML pages and Java applets. It requires third party Web servers and Web browsers. *eMerge i.way* itself is installed on a Web serving machine, interacting with *eMerge Enterprise Server* on one side and with a standard Web server product (MS-IIS, Netscape, WS, BEA, Domino) on the other.

eMerge Adapters

- ***eMerge Legacy Adapter***

eMerge Legacy Adapter is based on an *eMerge* application that communicates with any legacy application that runs on a 3270 or 5250 terminal. The aim is not to mimic the legacy application, but to provide a modern application that uses the legacy application screens as an external data source. Thus, instead of modifying the legacy application, it can be accessed through the existing 3270 or 5250 screens interface.

- ***DBMS Adapter***

eMerge DBMS Adapter enables an application developer to incorporate external data into *eMerge* applications. External data resides in a database maintained by one of the supported DBMSs.

- ***Custom Adapters***

A developer can write a *Custom Adapter* to enable *eMerge* to access non-supported data sources. These are data sources where the composite structure or access method are not standard, or databases where the data structure or DBMS is not supported by the *eMerge DBMS Adapter*.

Contact us

For more information, please visit us or contact us at:

www.sapiens.com

info.sapiens@sapiens.com

Tel: +972-3-790-2010