EXECUTIVE SUMMARY

Smart & Secure Enterprise Software

IQware - MUMPS Interface

V1.0
July 18, 2006

1.0 IQware Overview

IQware, Inc. makes industry-specific, patent-pending *Master Data Management (MDM) Business Intelligence (BI) Software* products. Generically, IQware's software does individually-tailored content acquisition, content management, content delivery and content presentation across multiple vertical markets. Specifically, IQware performs interactive data acquisition, analysis, reporting, auditing, presentation, mining and archiving. The unique value of IQware software is that it is secure, immune to desktop viruses, platform-independent and rule-based.

IQware's MDM/BI software translates raw data into useful information through assured, accurate interactive analysis and presentation. IQware helps convert that information into knowledge by supporting its successful application - and then by managing the results. IQware naturally works with various ISR and database systems as dictated by the particular IQware deployment. Such systems include (but are not limited to) relational databases such as Oracle, SQL Server, etc. and database programming environments such as Dbase and MUMPS. IQware can also interface with traditional "flat" file systems, including OVMS, UNIX, IBM using any combination of direct, indexed or sequential access modes.

IQware also provides information assurance through a variety of (patent-pending) internal, secure mechanisms. Information assurance is an essential component of MDM because errors embedded in the "original content" – or in the acquired raw data – are significantly amplified and magnified by the "downstream" IT systems. Such IT systems have few reliable, independent mechanisms(s) for information verification. The consequences of such errors percolating through the organization are severely expensive at best - and fatal at worst.

IQware can acquire data from any source. This makes IQware ideal for deployment across existing and disparate IT systems. IQware's products also extend rather than compete with the functionality of traditional Enterprise Resource Planning ("ERP") systems. IQware focuses on the regulated industries, which have the following requirements and/or desires:

- Tailored, secure content management, analysis, delivery and presentation
- Information assurance and validation
- Interactive and comprehensive reports
- Interoperability and compatibility with new, emerging technologies
- Very high security with complete, tamper-proof audit trail
- Integrate information disparate IT and legacy IT systems
- Real-time (or nearly so) performance
- Inadequate information integration
- Inadequate tracking, auditing, reporting capability
- Cannot meet regulatory compliance with existing IT systems
- Cannot meet performance requirements with existing IT systems

These industries generally have one of more of the following characteristics:

- Deployment of legacy systems
- Many disparate data sources
- Multiple disparate IT systems
- Inadequate information assurance
- Inadequate information integration
- Inadequate tracking capability

EXECUTIVE SUMMARY

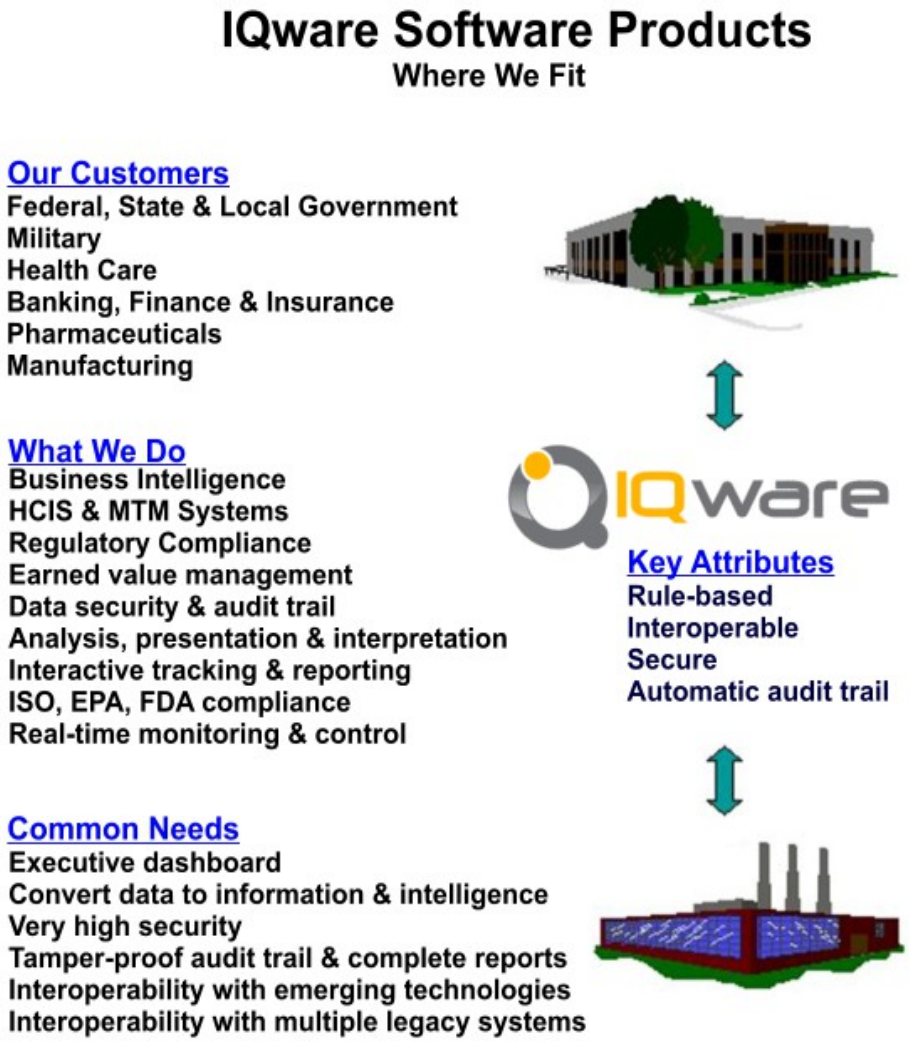
- Inadequate auditing capability
- Inadequate reporting capability
- Cannot meet compliance with existing IT systems

IQware's patent-pending software has three critical attributes:

- It is hacker-proof and immune to desktop viruses (patent-pending)
- It is rule-based so it can be changed on-the-fly and without programming
- It is interoperable so it works with emerging hand-held wireless devices

IQware's products are currently targeted toward the following industry segments:

- Federal government (communications, auditing, tracking, analysis)
- Military (DoD, information integration, analysis, reporting, C^3I)
- State government (infrastructure monitoring, control and auditing)
- Local government (infrastructure monitoring, control and auditing)
- Health care (regulatory compliance, information integration, reporting auditing)
- Pharmaceutical production (regulatory compliance, manufacturing)
- Pharmacies (retail MTM implementation and regulatory compliance)



2.0 MUMPS Overview

MUMPS (sometimes called "M") is a general purpose programming language that supports a native hierarchical data base facility. The language originated in the mid-60's at the Massachusetts General Hospital and it became widely used in both clinical and commercial settings. ANSI, ISO (ISO/IEC 11756:1992) and DOD approved standards were developed for Mumps, although several of these have lapsed.

Mumps differed from other mini-computer based languages of the late 1960's by providing: 1) an easily manipulated hierarchical data base that was well suited to representing medical records; 2) flexible string handling support; and (3) multiple concurrent tasks in limited memory on very small machines. Syntactically, Mumps is based on an earlier language named JOSS and has an appearance that is similar to early versions of Basic that were also based on JOSS.

MUMPs was originally written for the DEC family of mini-computers, including the very popular PDP-11 series. The operating systems supported included RT-11 (single-user), TSX-11 (multi-user), RSX11, RSX11-M, and RSTS/E. DEC-10 and DEC-20 systems also supported MUMPS applications for a number of hospitals.

Initially, the MUMPS' structure was limited by the constraints imposed by the minicomputers and operating systems on which it was originally implemented. An early design goal for Mumps was to provide multi-user, time-shared access despite the very limited memory and mainly single-user operating systems available at the time. Consequently, early implementations of Mumps were mainly standalone, interpreter based, dedicated operating systems. In these implementations, each user was assigned a very small region of memory for both code and local data. Source code was loaded from external storage and stored in memory in source form. Since Mumps programs were mainly interactive and dominantly data base access dependent, direct interpretation of source code did not introduce serious performance penalties. Typically, program partitions were less than 4,000 bytes, including all data, stacks, code and buffers thus allowing multiple time-shared partitions on even the smallest machines.

Because of early mini-computer memory and address space limitations, MUMPS applications usually consisted of many small, task-specific programs that were modular, compact, highly abbreviated, concise and focused on a limited objective. Program modules then, as now, were loaded frequently from a library. Applications typically consisted of a tree-like hierarchy of program modules that often corresponded to the structure of the underlying database in an early form of object-oriented programming style.

An excellent example of an early system structured this way can be found in the structure of the COSTAR System that employed well over a thousand tightly coupled and encapsulated separate code modules to service an ambulatory patient record data base. A more recent example is the widely used Veterans Administration Distribute Hospital Computer Program (DHCP) system, now known as Veterans Health Information Systems & Technology Architecture, which consists of several thousand Mumps routines. Of course, DHCP now refers to the "Dynamic Host Configuration Protocol" which allows network devices to automatically "lease" a valid IP address from a server.

Initially, all Mumps implementations were pure interpreters. As Mumps evolved, various methods were developed to partly compile Mumps code to intermediate representations, similar to Java byte codes or UCSD Pascal P code. However, due to indirection and the Xecute command, the interpretive nature of the language was always present. Indirection permits a Mumps program to dynamically construct and execute Mumps expressions and commands.

In 2000, development began on a compiler for Mumps, written in C and C++ that translates Mumps to C++. The compiler initially began as an interpreter implementation. It supports fast, flexible, multi-dimensional and hierarchical storage, retrieval and manipulation of data bases ranging in size up to 256 terabytes (TB).

3.0 IQware Interface to MUMPS Applications

IQware can share data with MUMPS applications by directly accessing the on-disk records created and used by the MUMPs applications. IQware can also interface with MUMPs applications by dynamically launching a MUMPS-coded "agent" to access the requested data records on an as-needed basis. A third way that IQware can connect to MUMPS applications is by dynamically copying MUMPS data records to its internal relational database. This lets modern structured applications and relational database tools be used on the data records. Note that this dynamic record transfer can be bidirectional, if desired. In this case, the MUMPS "agent" must be setup for record locking to prevent conflict issues. A sample deployment is shown below.