Direct Model Execution

The key to IT productivity and improving business performance

Most companies suffer from the same symptoms. The flexibility required to adapt existing IT infrastructure to rapidly changing business requirements is not there. Yet the cost of IT maintenance and change management is rising, adding further strain to the notoriously difficult interactions between business and IT. Attempts to reduce total cost by outsourcing or off-shoring workload to lower cost labor resources shift the problem without attacking the root cause: missing transparency and control due to a complete lack of accurately documented production systems.

There are three main reasons for the status quo. (1) No methodology is available to integrate the complete software lifecycle from process design to application and therefore the gap between business and IT cannot be closed. (2) The development tool chain from software architecture design to implementation, quality assurance and operations is broken at various stages, which dramatically increases overall complexity. Constant technical evolution combined with heterogeneity caused by mergers and acquisitions increase the complexity even further. And (3) most the fundamental problem with today's software development methodologies remains the notoriously undocumented code left behind by developers who need to perform their work under constant time and budget pressure. Lack of time and money paired with a shortage of skilled resources cause a permanent state of compromise and unfinished business, which cannot sustain the external pressures of globalization and a modern networked economy.

A radically new, but standards compliant and best practice oriented approach is needed to solve these three challenges. Direct Model Execution is such a radically new approach which helps transform existing software infrastructure via a completely model based approach where traditional programming has been completely eliminated. Direct Model Execution raises the level of abstraction from non-transparent, mostly undocumented code to unambiguous executable models which serve simultaneously as documentation and production runtime. The resulting development speed, quality, transparency and control induce a decisive shift from a dependency on individuals to an exclusive - and desired - dependency on architecture and process. This is a critical factor in restoring a functioning dialog between business and IT, one which enables a seamless lifecycle approach from process to application

Given past failures in raising the level of abstraction, automation and reuse, Direct Model Execution in the context of integrating heterogeneous applications seems to be nothing more than wishful thinking. To most IT architects it's simply too good to be true. It has thus gone largely unnoticed that leading organizations have been successfully using this approach for several years. And the results are nothing short of impressive. Total cost and time-to-market are slashed, leading to previously unattainable levels of flexibility, sustained operational efficiency and consequentially – true enterprise agility, enabling strategic competitive advantage. At the same time, Direct Model Execution enables an end-to-end Model Driven Integration which extracts more value from prior software investments, despite constantly changing business requirements dictated by ongoing process automation both within the enterprise and along the supply chain.

This White Paper explains the fundamental principles underpinning Direct Model Execution and Model Driven Integration and introduces what is seen by industry analysts and opinion leaders around the globe as its first field-proven, commercially available implementation: the E2E Bridge® platform.

Introducing Direct Model Execution

Model Driven Architecture® and the holy grail of IT

Proposed by the Object Management Group (OMG), the Model Driven Architecture (MDA) offers an open, vendor-neutral approach to mastering technological change. It does this through separating business logic from underlying platform technology so that software components can be redeployed without need for manual re-engineering. This is achieved through the use of models that cover every aspect of the IT architecture, from the definition of metadata to the orchestration of activities and the deployment of software components. These models enable the separation of business process definition and implementation logic, permitting maximum reuse of software components and thereby avoiding the need for repeated investments in time and money.

At least, that's the theory.

Code generation leads to unnecessary complexity

With the advent of MDA, software development tools have introduced code generation as a means to free the developer from mundane, repetitive coding tasks. While superficially this may seem to increase productivity and quality, in most cases code generation only provides an initial skeleton derived from structural design. The developer is still left with the daunting task of manually implementing – and maintaining – actual application or service behavior in form of traditional program code.

With debugging and runtime performance modifications still tied to the code level, the dubious gain of model driven automation is quickly offset by the significantly higher skill set expected from the developer. In addition, automated code generation tends to result in massive code bloat that causes significant performance problems in production, which increases infrastructure requirements massively. Rather than simplifying development, the hybrid approach based on code generation minimizes the value of MDA – and increases total cost, rather than reducing it.

E2E unlocks the value of Model Driven Development

E2E's Direct Model Execution approach applies MDA core values - abstraction, automation and reuse - to application integration and SOA development. Code generation and traditional programming is eliminated altogether. Instead, a high performance UML® Virtual Machine is used to execute standard-based models as designed.

Furthermore, MDA principles are applied consistently across the complete development lifecycle. In addition to the structural and behavioral design of integration services and processes, architectural design, Model Driven Security and model based quality assessment are included to provide end-to-end exclusive use of models in the enterprise software domain.

Direct Model Execution and Model Driven Integration are not born out of theory, but out of 20 years of practical experience gained from large scale enterprise initiatives to go seamlessly from process to application. With TCO and development cycles reduced by factors and with total quality control and unprecedented transparency, IT organizations finally reach the level of flexibility required to provide businesses with greater agility and to deliver the required functionality with certainty.

Direct Model Execution with the E2E Bridge

The E2E Bridge is the first implementation of a fully integrated development environment for Direct Model Execution and Model Driven Integration. Ever since its first application for massive-scale integration efforts in the Swiss banking sector, the E2E Bridge has made Model Execution accessible to a wide range of industrial sectors, including banking, supply chain, retail, logistics, telecoms, transportation, energy and government.

In practical terms, the E2E Bridge unifies the description of data, processes and backends to simplify integration, migration or consolidation projects end-to-end using a subset of standard UML (see Figure 1), as well as direct import for BPMN and ARIS EPCs as modeling notations. This means:

- Processes, services and events can be represented together in a shared, permanently documented context, transparent for business analysts as well as for developers
- Structural models representing data from several independent applications can be combined in one repository, irrespective of their original format or location
- Backend access can be virtualized in a way that abstracts protocol, middleware, and operating system specific configurations
- Systems and service components are documented in an architectural overview that serves both as definition of the runtime environment as well as the actual deployment scheme
- One single, hierarchical model contains all information, from design to production

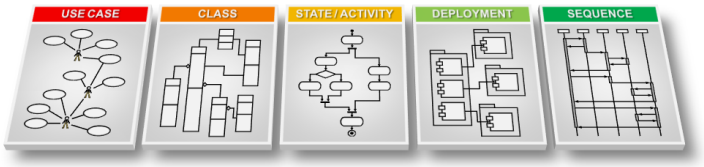

Figure 1: Subset of UML used for Direct Model Execution

The E2E Bridge ecosystem (see Figure 2) consists of three components: (1) the E2E Builder, a development environment, which integrates with XMI-compliant modeling tools, (2) the E2E Server, an execution container that combines the capabilities of a process engine, and enterprise service bus and an application runtime, which executes validated UML models, and finally (3) the E2E Console to manage the service lifecycle, provide runtime monitoring and support multitenancy management.

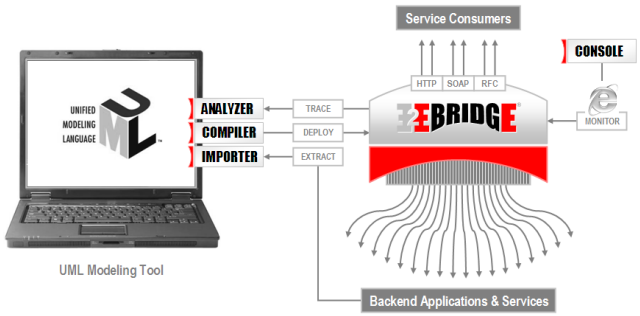

Figure 2: The E2E Bridge Ecosystem for Direct Model Execution

E2E Builder: Model Execution plug-in for UML tools

The E2E Builder is the developer's day-to-day working environment. Implemented as a Java plug-in for XMI compliant UML modeling tools, it is designed to be used by any developer with minimal UML experience, using familiar best practices for development. It includes the following functionality:

- Automated import capabilities for external process specifications in form of BPMN or the ARIS EPC format, combined with backend interface importers for WSDL, Java JAR files, W3C schemas, EDI/EDIFACT, flat files, as well as for application-specific metadata and function interfaces, e.g. SAP IDoc, BAPI and RFC interfaces
- A model compiler that performs UML model validation and optimization, generating runtime models which can be executed by the runtime infrastructure

- A model debugger, which creates UML traces from test cases which can be automatically composed, together with a log file importer to automate regression testing for asynchronous process scenarios

E2E Server: the first UML Virtual Machine for SOA

The E2E Server environment consists of two independent components, the execution container and the console. The container is implemented as a Virtual Machine for UML which executes process and application logic and performs synchronous and asynchronous application connectivity. In other words, the E2E Server combines the functionality of a process engine, and enterprise service bus and a lightweight application server in one container. The modern microkernel architecture provides extreme transactional performance and high scalability with a minimal system footprint. It consists of three subcomponents that can be farmed out onto different server nodes depending on the chosen system architecture and the given production environment, including federation, multitenancy, high availability and failover:

- The frontend covers protocol-independent server capabilities, including handlers for the processing of signals and events. Protocol support covers standards such as HTTP/S and SOAP, but also proprietary application interfaces such as SAP t/RFC.
- The backend houses interface bridging technology for synchronous and asynchronous application connectivity. Out-of-the-box, the E2E Bridge supports over 50 backend adapters, covering access to databases, systems, message protocols and platform-specific connectivity, including Java EE and the proprietary t/RFC protocols from SAP.
- The kernel, the actual Virtual Machine for UML, manages the runtime model repository and executes models which have been previously deployed. Models are executed in memory, allowing for extreme transaction processing on relatively low-cost systems.

The E2E Console is a Web-based control portal. It manages the lifecycle of executable models, controls activation and deactivation of individual services and monitors their function. Furthermore, it governs administrative access, multitenancy management as well as installation and de-installation of runtime firmware components.

Based on a modern microkernel architecture, the E2E Bridge runtime environment is optimized for performance, quality of service, scalability and maintainability. It is available for a wide range of operating systems, including Windows, Linux, Solaris, AIX and HP UX. Due to its extremely small system footprint it can be operated on a broad range of platforms, including low-end Pentium III based PC architectures. It is designed to work on minimal virtualization infrastructure. Installation of the complete E2E Server environment takes 15-20 minutes.

The Model Driven Integration process

The governing principle of the Direct Model Execution is that the documentation IS the code. Hence, the developer is naturally forced to provide just enough information to ensure transparent change management at all times, independent of prior knowledge of the development scenario. Given that models are at the same time both the documentation *and* the runtime artifacts, Direct Model Execution facilitates a highly agile development process, called Model Driven Integration (see Figure 3), that is ideally suited to deal with the complexities of SOA development. It allows developers to spend practically all of their time on stakeholder communication as well as service and process design – and not the implementation thereof. Otherwise, the Model Driven Integration process follows common best practices and therefore requires little or no special training to be used effectively.

One of the key strengths of the Model Driven Integration development process is that it does not rely on any preconditions to fit a given operational context, whether at the architectural or at the organizational level. The end-to-end philosophy naturally favors small, incremental steps. Regardless of the architectural approach – top-down, bottom-up or middle-out – the typical development cycle follows a five-step process from process design and service identification to runtime execution in production. The following paragraphs describe each of the five steps in more detail:

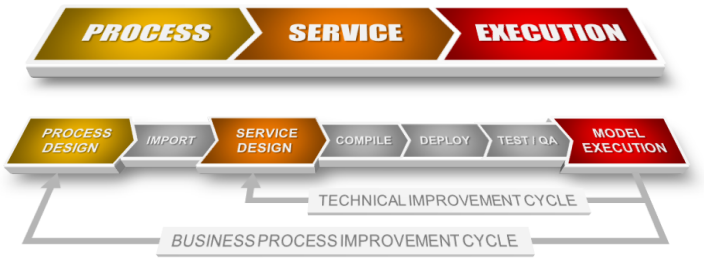

Figure 3: The Model Driven Integration Development Process

Step 1: Process design and service identification

Service identification starts with two sources of information: (1) a clear understanding of the process design that will use the new or modified service, and (2) use case scenarios describing user roles, security requirements and access to the desired applications and data sources. Rather than describing the targeted services in prose, UML use case diagrams, high level activity diagrams and state diagrams serve as the communication vehicle for business analysts and IT specialists, allowing them to work towards an unambiguous common understanding. User role and security related information collected in this form is already compiler-relevant. Therefore reaching agreement between business and IT at this level ensures that the lower level service design complies with an agreed high level design contract. Experience shows that at this stage of the project, iteration cycles are extremely short and progress is made in small incremental steps, helping to bring divergent semantic views between business and IT quickly into alignment.

To enable a tight interlock from process to application, the E2E Bridge facilitates direct integration with the models created by leading Business Process Management platforms. Process models following the standard BPMN notation promoted by the OMG, as well as vendor specific models such as the popular EPC format are supported for direct process import. Depending on the semantic richness of the chosen modeling notation, critical elements required for online monitoring and process performance management, such as Key Performance Indicators, can be added at this stage already, and will be used in real-time during execution.

In order to define the security model to be used by a given application or service, the Model Driven Integration process facilitates fully Model Driven Security, where existing security policies and procedures can be seamlessly linked to business services, in a transparent fashion that provides a clear separation of concerns, simplifying governance and auditing.

Step 2: Automatic backend interface extraction

SOA-enabling existing backends represents two main challenges: (1) existing applications may not support modern service access protocols, and (2) documentation of metadata and functional interfaces may not represent the version used in production – and in far too many cases, no accurate up-to-date documentation is available to begin with. The model driven integration process addresses both challenges in a generic way. Rather than just providing backend API support and runtime connectivity, the E2E Bridge permits fully automated extraction of metadata and functional interfaces from the production backends. During the extraction process, metadata from backend runtime data dictionaries are transformed into UML class diagrams, whereas functional interfaces are automatically wrapped as atomic services and exposed in the form of a UML action state, together with an appropriate stereotype and tagged values.

The benefits of this procedure are twofold: (1) as a side effect, previously undocumented interfaces are automatically re-documented without developer intervention, reflecting the current production runtime; and (2) the backend is abstracted into a platform-independent representation instead of forcing the developer to maintain a platform-specific model. While the main objective of this step is to service-enable existing backends, many organizations perceive the inherent re-documentation itself as a value-adding capability. Extraction functions provided by the E2E Bridge are generic, i.e. custom extensions of platform-specific metadata and functional interfaces are covered as well. Such generic extraction capabilities are currently provided for WSDL, W3C schemas, EDI/EDIFACT and flat files, Java JAR files, as well as the proprietary RFC/BAPI and IDoc interfaces of SAP systems. Given the automatic nature of the development process, any of these backend types is rendered accessible in a platform-independent fashion within seconds, regardless of the degree of customization or the available documentation.

Step 3: Mapping, routing, events and orchestration

Once relevant metadata and function interfaces have been extracted from the application backends, metadata must be harmonized according to business needs. Simple attribute mapping is performed using UML class diagrams. Quite often, however, more complex transformations are necessary. For low level data transformations, the E2E Bridge implements UML Action Semantics, a standardized script language which offers the developer the right balance between graphical diagramming and precise computing instructions. In order to perform a simple string manipulation, for example, a graphical diagram would be overkill, whereas UML Action Semantics provide a concise way of expressing the desired conversion. In even more complex cases, UML activity diagrams are used to provide the right mapping, especially if conditional rules need to be applied.

Message routing, event processing, business logic, service orchestration and process execution are naturally expressed using UML activity and UML state diagrams. Here's where service-oriented and event-driven concepts seamlessly merge without forcing a particular architecture. In terms of automation, state diagrams have the additional advantage that they provide an intrinsic abstraction for stateful service design, as required for long-running processes. Underneath these, the E2E Bridge allows existing persistency strategies to be reused that may already be established in a given organization.

Step 4: Validating and deploying executable models

After the static and dynamic properties of a service or a process configuration have been modeled – up until now we're still in the platform-independent domain – the runtime context must be defined. This step will bind the designed entities to physical entities such as databases or backend applications. The actual mapping is done using UML component and deployment diagrams that are automatically created using a deployment wizard, to collect relevant information about system nodes, operating systems, IP address and protocol specifications, port addresses and access credentials for backends, etc. Once the runtime context is completely defined, the service model can be compiled and deployed to one or more runtimes.

Technically, the model compiler requires only the XMI representation of the UML model, an XML specification of both the graphical and the logical content of the original model. During compilation, graphical information is discarded and the structure of the model is validated with respect to logical consistency and integrity. If no errors are detected, an optimized model byte code for execution on the UML Virtual Machine is produced.

By specifying the deployment scenario in UML, deploying the compiled service and process models to the runtime infrastructure becomes a transparent – and fully automated – procedure. This is a powerful capability which receives high marks from developers of distributed applications, who are used to the cumbersome, hard-to-maintain build environments known from traditional software development. Manual labor is replaced by the press of a button.

Step 5: Model-based testing & debugging

Every developer who has ever had to debug software components in a distributed heterogeneous application environment understands what it means to suffer. Development tools offer only rudimentary support to better understand runtime behavior of production artifacts once they're deployed and in operation. And asynchronous, event driven scenarios are almost impossible to test and debug, given the unpredictable state that various services may be in.

To better address this challenge, the E2E Bridge offers a model debugger specifically conceived to make quality assessment in distributed, event driven environments fully transparent. It fulfils two main functions: (1) it allows service models to be debugged for environment errors using a UML-based model trace, and (2) it automatically builds a comprehensive test case library for regression and load testing. And for event driven scenarios, test cases can even be automatically recorded to reproduce faulty runtime behavior for quality assurance.

The UML model trace deserves special mention. Since models are directly executed, i.e. there is no code, this unique capability returns executed data streams and their performance profile, including all messages sent between the involved systems, back to the modeling environment in the form of fully navigable UML models. Data flows are visualized in several steps. UML sequence diagrams generated for debugging purposes are automatically generated by the UML Virtual Machine during model execution, providing an overview of the backends which were accessed during a service call. The second level of graphical visualization shows how the service was actually executed, by means of UML activity diagrams that display the logical flow of the service. The UML trace optionally contains detailed performance profiles, documenting potential bottlenecks in service behavior under load conditions.

The benefits of Model Execution are endless

Most software architects intuitively understand the significance of the statement "The documentation IS the code!" Even though it sounds like an intangible dream, the apparent benefits are obvious: complete transparency, maximal development speed, controllable production quality and above all the certainty to implement the right things in the right place and the right sequence, as needed by the business. But beyond just simple speeds and feeds, application of model execution in real-life projects shows that using an end-to-end model-based approach to dealing with distributed applications has more far-reaching consequences:

Improved communication between business and IT

Using models from process to application facilitates communication between business and IT. It brings both views into a shared, visual design context. The entire lifecycle of a project, from the perspective of business analysts and developers alike, can be managed by one single tool.

Certainty and timeliness of SOA enablement projects

With direct model execution and model driven integration, service development, deployment and production based on business specifications becomes a predictable effort. From the early simulation of business processes based on orchestrated services to the automated testing of backend access functionality, complete transparency is achieved throughout process and service lifecycles, keeping project deliveries on course at all times.

Consistently high implementation quality

Using model-based automation wherever possible has three valuable consequences. Firstly, it avoids defects based on inconsistent human work performance that can result in unpredictable failures and bugs which later need to be painfully removed, causing unnecessary delays and maintenance cost. Secondly, the validations which can be applied to models allow checking for model consistency and integrity. And thirdly, with developers freed up from coding and debugging tasks, more time can be spent on maximizing design quality and reuse potential.

Value extraction from existing software assets

Based on a non-intrusive approach, direct model execution and model driven integration maximize the value of what has been previously purchased or developed. Once existing legacy software has been encapsulated and documented in the form of reusable, easy-to-maintain services, the decision to replace them can be made based on economic, rather than just technical, criteria.

Elimination of migration risk

Direct model execution enables soft, stepwise migration of software assets and eliminates the technical risk of migration and consolidation efforts. For outsourcing or offshoring initiatives, the permanently documented state of operational interfaces eliminates uncertainties typically associated with such projects.

Call to action: engaging in a Proof-of-Concept

Direct Model Execution and Model Driven Integration are used successfully by leading organizations in various industries, such as finance, telecommunication, energy and logistics, from large mission-critical multinational projects down to small point-to-point integration efforts. The pattern used by all organizations to get on board with the new approaches is always the same: a 2-3 day proof-of-concept that allows organizations to evaluate Direct Model Execution in the context of their own IT infrastructure. Regardless if you are an end user organization, an independent system vendor or a system integrator, ask for a proof-of-concept using the E2E Bride in your own environment to assess the possibilities of Direct Model Execution to solve your own IT challenges.

More information available on E2EBridge.com

Much can be learned from projects that have been performed so far. On the E2E home page, you will find written end user case studies as well as recorded online presentations that describe concrete projects in detail. Furthermore, you will find online demonstrations that show the E2E Bridge in action. In particular, if your IT challenge falls into one of the categories below, your situation is a prime candidate for Direct Model Execution:

Third party software integration with SAP and Oracle

The E2E Bridge is certified for SAP Integration and SAP NetWeaver, and interoperability with Oracle Fusion Middleware has been established in close collaboration with Oracle. For any integration project with SAP solutions, Oracle solutions, or both, the E2E Bridge is the perfect tool to complement your platform strategy. More information on this subject can be found at www.E2EBridge.com/sap and www.E2EBridge.com/oracle.

Modernization and SOA enablement of legacy applications

The transparency which is inherent in Direct Model Execution lends itself to effectively transform hard to maintain legacy applications into state-of-the-art SOA applications. Check out the case study on Intrum Justitia's modernization of its in house developed core application ReCash, which is available both as a written case study as well as an online customer interview. More information on www.E2EBridge.com/modernize.

End-to-end control from process-to-application

If you plan to use Business Process Management or Process Performance Management based on tools such as ARIS, consider complementing your effort with Direct Model Execution. The E2E Bridge helps implement the right thing in the right place and the right sequence, eliminating misunderstandings between business and IT. More at www.E2EBridge.com/aris.

Simply integrated.

Contact E2E

Basel +41 61 270 97 10
Portland +1 503 781 4547
Denver +1 303 881 9519

Email info@E2EBridge.com
Web www.E2EBridge.com