

# Bridging Business Models to System Implementation

Intelligent Modeling for Effective Alignment of Processes,  
Use Cases and System Specifications

*This white paper is provided courtesy of*



# Executive Summary

Business process automation has long pursued a formalized, business-driven application development and implementation life-cycle. We want to understand our business and gracefully evolve that understanding into the design and implementation of business systems. We want the deliverables of each life-cycle stage to flow into those of subsequent stages with forward and backward traceability. And we do not want to retool our development staff.

Manual attempts to record and manage the life-cycle details have proven to require more effort than our project deadlines will support. In response, organizations often employ shortcuts such as discarding business requirements at the beginning of design. Past efforts lacked a single methodology, especially in the area of analysis and design. As a result, it was difficult to organize teams that could maintain a single methodological focus.

Many of the prior solutions required the use of a single CASE tool to which many organizations could not or would not conform. These monolithic tools required substantial investment in equipment and training to arm a relatively small portion of the organization.

Today, modern computers are abundant and powerful. Organizations do not have to invest in special high-powered workstations and boot-camp training to roll out a concept-to-code solution. Applications can more easily exchange data within the same machine or among machines with a drag and a drop. Everyone is connected and routinely exchanges data in electronic form.

New modeling techniques and approaches have evolved up through the new century, primarily due to extensive business process analysis and modeling, system and application modeling, and component-based development initiatives. Today, process and workflow models, relational database technology, and the UML-based use case and class modeling represent standards that are largely unchallenged. Collectively, these are the key components of a continuous life-cycle.

This paper describes the Metastorm approach and technology for using an enterprise modeling approach in the modern application development and implementation life-cycle.

# Contents

<b>INTRODUCTION .....</b>	<b>3</b>
<b>THE APPLICATION DEVELOPMENT AND IMPLEMENTATION LIFE-CYCLE.....</b>	<b>4</b>
<b>ENTERPRISE MODELING .....</b>	<b>4</b>
CONCEPTUAL MODELING AS THE STARTING POINT.....	4
<i>Workflow Model.....</i>	<i>4</i>
<i>Use Case Model.....</i>	<i>5</i>
<i>Workflow Model to Use Case Model.....</i>	<i>6</i>
FROM CONCEPTUAL TO LOGICAL .....	6
<i>Use Case Model.....</i>	<i>7</i>
<i>User Interface Design .....</i>	<i>7</i>
<i>Functional (Component) Design .....</i>	<i>8</i>
<i>Data Design .....</i>	<i>8</i>
<i>System Interface Design.....</i>	<i>8</i>
<i>Implementation.....</i>	<i>8</i>
METASTORM PROVISION.....	9
<i>Models, Not Simply Diagrams.....</i>	<i>10</i>
<b>SUMMARY .....</b>	<b>10</b>

# The Application Development and Implementation Life-cycle

The application development and implementation life-cycle starts with the business (or enterprise) and ends with an implemented set of system applications and components. As we traverse the life-cycle, we find numerous points of iteration, but ultimately the life-cycle progresses through three major (classical) stages: conceptual, logical and physical.

The conceptual stage of the life cycle involves understanding the business. Sometimes this is done to achieve business process improvements; other times it is used to guide the development or purchase of application systems. Work here is often called enterprise modeling or business analysis. A key work product is business requirements.

During the logical stage, business requirements help guide the engineering of system components. This stage is commonly known as system architecture and design. The key work product is system specifications.

The physical stage commits engineering decisions of the logical stage to a particular platform. It involves the low-level engineering, coding, testing, deployment and maintenance. The key work product is a set of system components and an implemented production system.

In essence, application development life-cycle involves two major elements, the business and its systems. Business Process Analysis (BPA) and complementary approaches represent the modern methodology for improving business processes and systems, and thus the business. BPA challenges current practices to invent new ways of conducting business that meet the demands of the market and embrace the capabilities of modern technology.

To effect changes in the business we employ technology. The modern approach for analysis, design and development of systems is accommodated by UML (Unified Modeling Language). Class and use case modeling, very popular analytical techniques popularized by UML, add an effective means of describing business situations in terms of interactions between people and system functions.

## Enterprise Modeling

### Conceptual Modeling as the Starting Point

Enterprise modeling implements the conceptual stage of the life-cycle. This is where the enterprise is scoped into manageable pieces. Select pieces are then studied to improve business processes (business process improvement) and/or to formulate business requirements (business analysis).

Enterprise modeling does not imply studying the entire enterprise. In many cases, it is conducted within a defined scope in preparation for new systems development or the implementation of enterprise software. The following discussion focuses on two models, the workflow model and the use case model, which play a major role in enterprise modeling.

#### Workflow Model

A key tool for enterprise modeling is the workflow model (or business activity diagram). This model describes the precedence of business activities within a business process. It enables analysts to identify and correct bottlenecks and inefficiencies. In addition to precedence, the workflow model can express which department is responsible for each activity. This is done by placing the activities of a given department within the “swim lane” that represents the department.

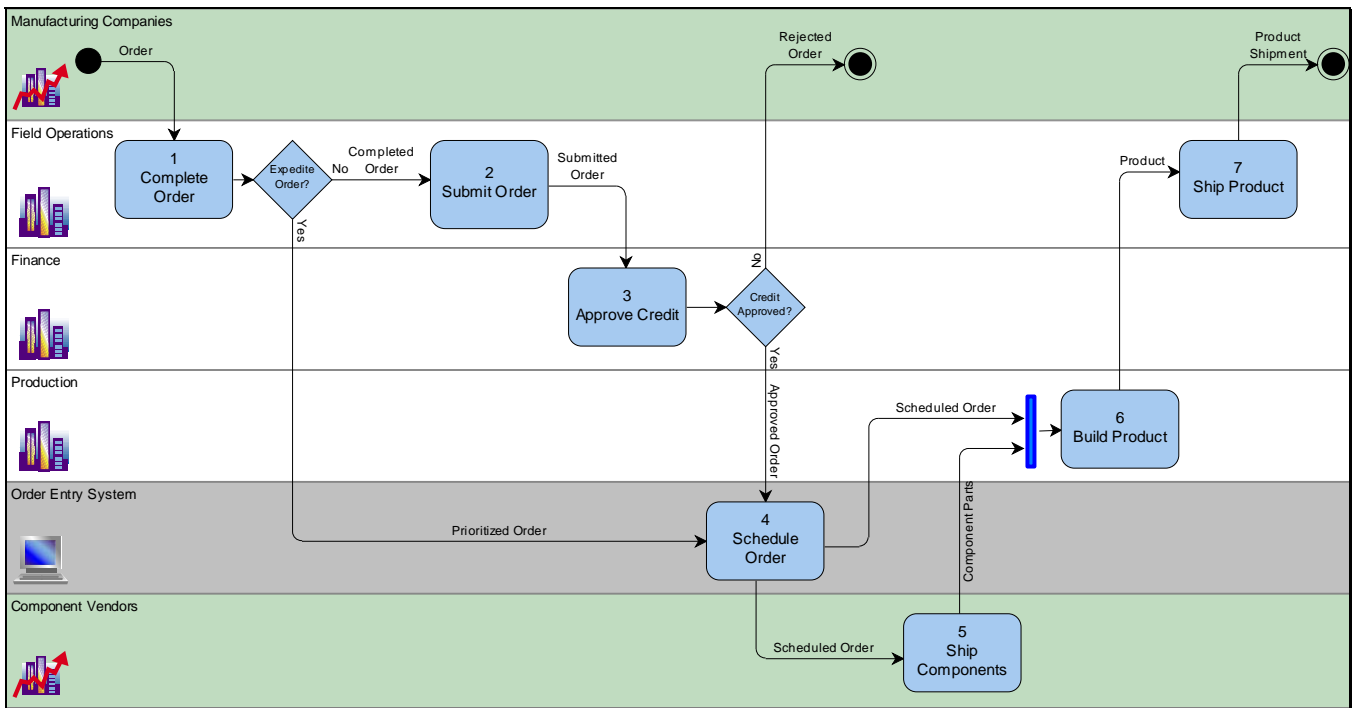


Figure 1 – The workflow model describes the flow of activities within a business process.

The above workflow model (Figure 1) depicts the activities involved in fulfilling orders and the departments or systems responsible for each activity. Since business processes can be very complex, it is often useful to nest workflow models rather than place all details on one model. This creates a hierarchy of models, each of which describes the details within a single activity.

To achieve process improvement, activities and workflows (the links) carry substantial information such as the time and money they consume. By analyzing process costs and timing, business analysts can optimize workflows, consolidate or eliminate activities, etc. to design a better business process.

### Use Case Model

The use case model is a very popular technique for expressing how people, departments and systems interact to accomplish a portion of the business (or system). It is a simple model, yet provides a powerful means to express continuity from the business to supporting systems.

The essential idea of a use case model is to represent how actors (job roles, departments and external systems) interact with recognized business or system functions. A completed use case model represents a prototypical interaction from which numerous usage scenarios can be derived. Simplicity is a key reason for the popularity of the use case model. It often depicts a concrete situation observed in the physical world.

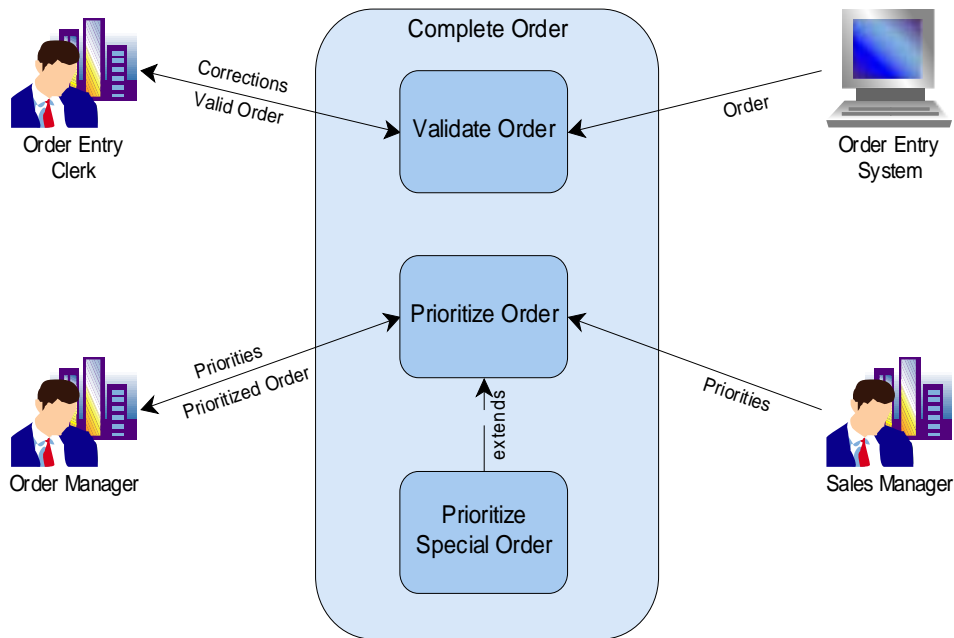


Figure 2 – The use case model depicts a prototypical scenario.

Figure 2 depicts the interplay required to complete an order. This involves both validation and prioritization. The order entry clerk and the order entry system both participate in order validation. The order manager and sales manager participate in prioritization. Occasionally, an order will require special handling. The use case model depicts this with “prioritize special order” shown as an extension of the normal “prioritize order.”

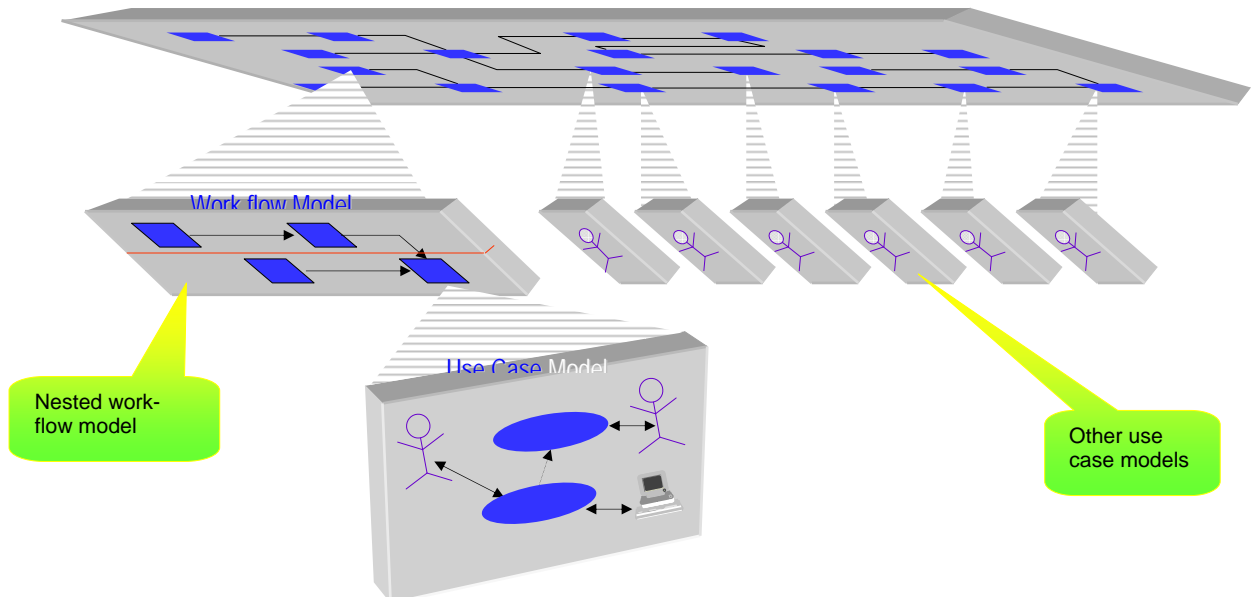
The informality of the use case model contributes to its simplicity but hinders overall consistency. It is normal to produce many use case models, each representing a different perspective of the business.

Use case models require careful management. These models express details within a rather small scope and pay little attention to other use case models. On a large project, use case models often contradict each other. It is also easy to overlook areas where the use case should be used, leaving holes in the requirements.

Large projects benefit from a more global model to help avoid overlap and ensure the use case models collectively represent the entire scope of the effort.

### Workflow Model to Use Case Model

The workflow model is a natural host for use case models. Each lower-level activity in a workflow model depicts a significant but well-scoped portion of the business (or portion of a system). A use case model is used to expose how users interact with system functions to accomplish the activity.



*Figure 3 – The workflow model provides a broad scope for hosting use case models.*

Figure 3 depicts the ties between the workflow and use case models. The workflow model covers a wide scope involving numerous departments performing a variety of activities. By itself this model provides both a broad and detailed view of the business.

The use case model complements the workflow model. Each activity on the workflow model is a candidate subject for a use case model. The use case model describes what happens inside of the activity in terms of functions (sub-activities or use cases) and the business and/or system entities that interact with these functions.

## From Conceptual to Logical

The transition from conceptual to logical involves recognizing system components to implement the business components of the conceptual model. Use case models provide an excellent vehicle for making this transition

### Use Case Model

The use case model plays a major role in the transition to the logical stage. It provides an abstract view of how the system users (and external systems) interact with automated business activities. This provides a structure for identifying and designing user and system interfaces.

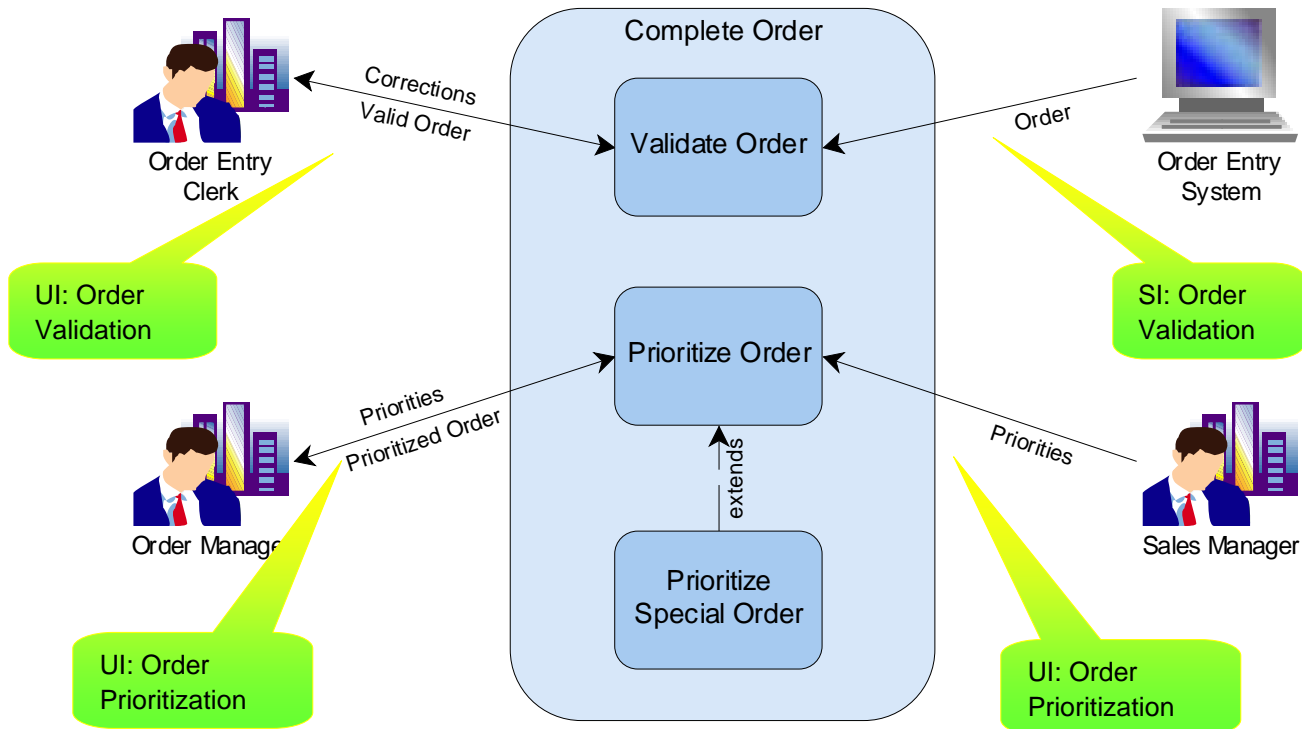


Figure 4 – The use case model exposes candidate interfaces.

User and system interfaces are a characteristic piece of the system architecture. Historically, this is also a point where life-cycle continuity broke down. When developed under the structure of a workflow model, the use case models collectively describe all key user and system interactions. Designing the user interface for the system involves inspecting each use case model to identify candidate user and system interfaces. A candidate user interface occurs whenever a business entity (e.g. a role or department) interacts with a function. The use case model the above figure illustrates three candidate user interfaces and one candidate system interface.

### User Interface Design

A candidate user or system interface provides a problem scope for design. The design process involves understanding the user interface in terms of the data and function provided to the user. Several models come into play when describing an interface. The models and their usage depend largely upon the complexity of the interface. In many cases, a designer would first express the user interface as a class model. The class model depicts the business classes, attributes, associations and operations involved in the user interface. In complex cases the designer would produce a sequence model (or collaboration model) to describe the dynamic behavior of objects within the interface. In all cases, the designer would provide detail to the user interface in textual and/or storyboard form. This detail is primarily used to communicate design intent to the developer of the user interface.

The interface in Figure 5 between the order entry clerk and the “validate order” function is further detailed in a class model and then as a prototypical dialog. The class model depicts the data and function made available to the order entry clerk during this interaction. The associated dialog provides a graphical “sketch” or rough idea of how the user interface should appear.

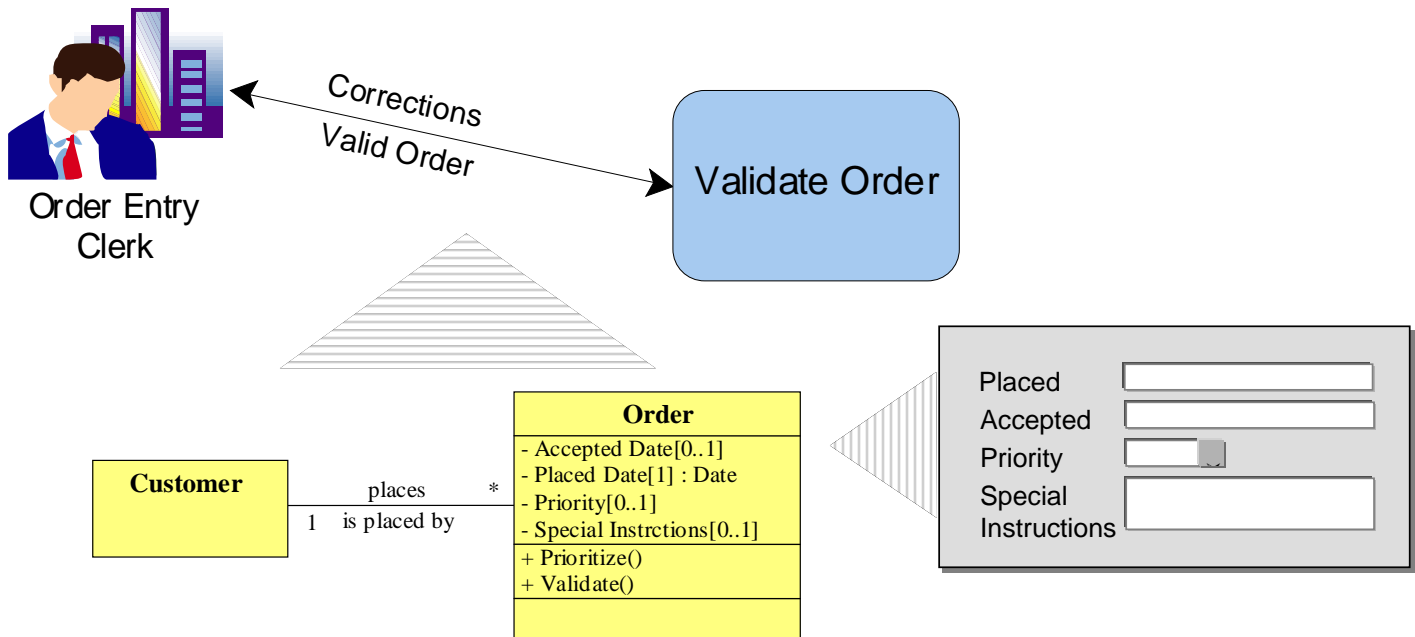


Figure 5 - The use case provides a structured means of identifying user and system interfaces.

### Functional (Component) Design

The first stage of design produces a class model to clarify the components that implement the system activity. Where needed, additional models such as collaboration or sequence models describe the dynamics of how the components interact to perform the activity.

### Data Design

The data architecture is always designed for some data storage technology. Whether the underlying storage technology is relational or object-oriented, designers must deal with the management of persistence. For relational technology, the engineering of elegant persistence mechanisms starts by mapping the persistent data for all components of the function architecture into corresponding relational tables. Subsequent steps include classical relational table design and providing a clean abstraction for interfacing with the function architecture.

### System Interface Design

The interface architecture, like the presentation architecture, starts with the use case models. Each interaction with an external system is a candidate interface. Figure 4 depicts an interface between validate order and the order entry system. The design for this interface involves describing the net data exchanged via a class model followed by low-level engineering based upon the technology employed (e.g. CORBA, batch files, inter-process communication).

## Implementation

The purpose of an application development life-cycle is to produce a working set of systems and components which accurately support the business.

Systems are developed and implemented (appropriately) from a design. The design specification includes the architectures and the design of the key components (user/system interfaces, functional components, relational tables, etc.). Our industry has an incredible number of authoring, development, and implementation tools and environments. A practical concept-to-code solution is one that enables conceptual and logical components to interface with the varied authoring and implementation environments in use by the IT organization.

Metastorm ProVision is designed to complement the existing base of authoring and implementation environments. Metastorm ProVision's intergrated repository provides intimate traceability from enterprise models through the generation system components.

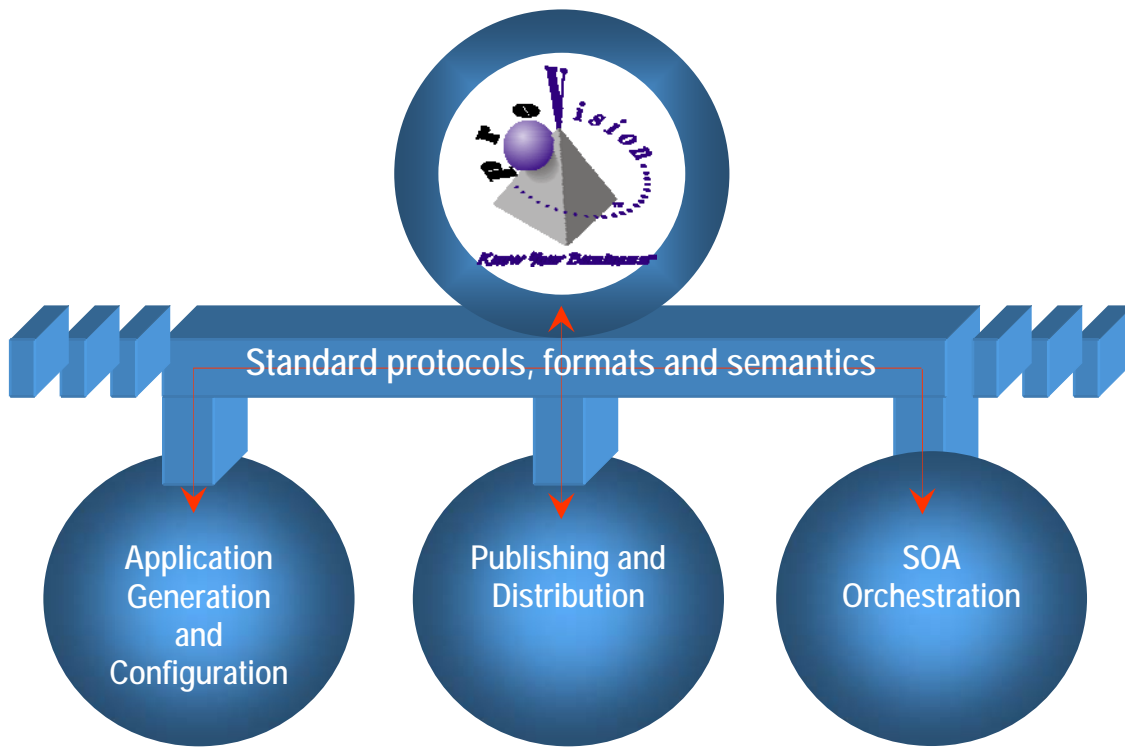


Figure 6 – Life-cycle technologies must co-exist with the many varied authoring environments.

The transition from logical to physical essentially involves identifying and creating system components that reflects design intent. From this point, developers employ authoring environments to conduct detailed algorithmic design and implementation.

Since development often reveals changes in design and business requirements, a two-way interface is required. The key technology for interacting with authoring environments is the XML-based Common Interchange Format (CIF) interface. Developed components from authoring and implementation environments are reverse-engineered to extract design information and automatically update the models in the Metastorm ProVision repository.

### Metastorm ProVision

Metastorm ProVision is the award-winning product that enables concept-to-code using established enterprise modeling concepts compatible with major authoring and implementation environments. It is the only analysis and design tool on the market that explicitly ties workflow models to use cases and class models, providing an easy transition from business analysis to system design and implementation.

Metastorm ProVision supports simple to sophisticated enterprise models using the most popular process and system methodologies. In addition to the workflow model, models exist to represent other dimensions of the business. The enterprise models are:

Business Models	Description
Business Interaction Model	Provides a strategic view of the business. Profiles and scopes the business showing organizational boundaries and interactions with internal and external organizations.
Workflow Model	Represents a business process in terms of component activities and the flow of work among these activities.
Goal Model	Shows the hierarchy of business goals associated with business processes and their activities.
Process Model	Shows the decomposition of the business domain into process and activities.

Organization Model	Represents the organizations and supporting roles of the enterprise in a hierarchical fashion.
Location Model	Shows the various geographic locations of interest to the enterprise.
Event Model	Organizes key business events and sub-events.
System Model	Represents the systems and subsystems that support the enterprise.

The transition from conceptual to logical is best characterized by the use case model. However, Metastorm ProVizion supports additional system models to enable a very detailed description of business and system requirements:

Analysis Models	Description
Use Case Model	Depicts prototypical scenarios involving the “actors” in the business (e.g. roles, departments, systems) and the functions they use to accomplish business activities.
Business Class Model	Shows the properties and interrelationships of the business classes (e.g. order, customer) of interest to the business.
Subtype Model	Depicts the fundamental hierarchies of business objects exposing the supertype/subtype relationships
Sequence Model	Shows how objects collaborate, via message passing, to accomplish a given function.
State Model	Shows the life-cycle of individual objects for a given class in terms of states (key stages in the life of an object) and transitions among those states.

### Models, Not Simply Diagrams

A diagram is a picture. Each diagram can convey thoughts graphically, but offers no support for:

- Integrity
- Properties beyond the generic name and description
- Sharing or reuse of objects on other diagrams
- Intelligent functions such as completeness checking and traceability
- Integration with other diagrams (reflecting a single change across all diagrams)

Each Metastorm ProVizion model is a view into a single integrated meta-model. Changes from one perspective are automatically reflected in others. Each model consists of a logically unlimited number of objects and links among these objects. And each Metastorm ProVizion object is intelligent. It knows how it participates in models and how to interact with other objects. In addition, each object carries characteristic information far beyond name and description.

Intelligent, integrated objects enable quality models that work together to effect transitions from life-cycle stages and provide automatic forward and backward traceability.

## Summary

Metastorm ProVizion supports an integrated enterprise modeling and analysis approach that co-exists with existing implementation environments. It supports the most common workflow and system methodologies with primary focus on the Unified Modeling Language (UML). This support extends across the enterprise using the multi-user capabilities inherent in the product.

## About Metastorm Inc.

With a focus on enterprise visibility, optimization, and agility, Metastorm offers market-leading solutions for Enterprise Architecture (EA), Business Process Analysis & Modeling (BPA) and Business Process Management (BPM). As an integrated product portfolio, Metastorm Enterprise™ allows organizations to maximize business results by unifying strategy, analysis and execution. Metastorm is the only solution provider to bring together these critical disciplines on a single software platform to enable an understanding of enterprise architecture and strategy, accurate impact and opportunity assessment, effective process execution, and accelerated value realization for organizations worldwide. For more information on powering strategic advantage with Metastorm Enterprise, visit [www.metastorm.com](http://www.metastorm.com).



1-877-321-META (6382)    +44 (0) 208-971-1500    [www.metastorm.com](http://www.metastorm.com)

© Copyright 2007, Metastorm Inc. All rights reserved. Enterprise Process Advantage, Metastorm BPM, Metastorm DNA, Metastorm Enterprise and Process Pod are either registered trademarks or trademarks of Metastorm Inc. Other product, service and company names mentioned herein are for identification purposes only and may be trademarks of their respective owners. 7.16.2007