

Business Modeling with the UML and Rational Suite AnalystStudio

A Rational Software White Paper

Rational[®]
the software development company

Table of Contents

INTRODUCTION TO BUSINESS MODELING	1
What is Business Modeling?	1
Why Business Modeling?	2
BUSINESS MODELING WITH THE UML.....	4
Use-Case Diagrams.....	5
Activity Diagrams	6
Class Diagrams	8
Interaction Diagrams.....	9
BUSINESS MODELING WITH RATIONAL SUITE ANALYSTSTUDIO.....	11
Business Modeling in the Rational Unified Process	11
Business Modeling in Rational Rose	15
Business Modeling in Rational RequisitePro.....	17
CONCLUSION	21
Additional Reading	22
Glossary	23
About Rational Suite AnalystStudio	24

Introduction to Business Modeling

These days, software teams can build some pretty amazing systems, compared to only a decade ago. A lot of progress has been made in building and deploying complex systems; from modularization and encapsulation through structured programming and object-oriented development to visual modeling and process automation.

One area where software professionals have not been good is ensuring that they deploy the *right* system: the one that really will make their end-users lives easier, and consequently will help their business succeed. Too often, solutions are quickly sketched (if at all) before they are designed and implemented, and once deployed, they are not actually solving any specific problem.

Understanding *where* systems are being deployed, *who* uses them, *how* they need to integrate with existing systems, and *what* specific business tasks they automate is key to deploying a successful system.

This paper concentrates on Rational's support for business modeling, provided in Rational Suite AnalystStudio (as well as Rational Suite DevelopmentStudio), a suite of tools that assists software teams in solving the right problem and defining the right solution to that problem. The support for business modeling provided by Rational uses the same use case approach to modeling a system. This approach makes business modeling more accessible to software professionals familiar with use case and UML modeling techniques.

If you want to maximize your ability to deliver a successful system or if you are involved in the early phase of a software project, then reading this paper is highly recommended.

What is Business Modeling?

Business modeling is a technique to model business processes. Business models provide ways of expressing the business processes in terms of business activities and collaborative behavior.

Models are helpful to document and comprehend complexity. They are powerful for two main reasons:

- They make communication easier,
- They allow us to easily manipulate solutions without affecting the 'thing' being modeled.

In the face of increasingly complex systems, visual modeling becomes essential. As the complexity of systems increases, so too does the importance of good modeling techniques. These models allow different stakeholders to view the system from different angles and communicate their perspectives with each other. Using a rigorous modeling language standard is one essential ingredient to a good modeling technique.

Business modeling is a technique to help answer critical questions, such as:

- How do you know you have identified all system use cases?
- What do the workers (users) do before using our system?

- What business value does the system bring?
- What is the business system this computer system will be supporting?

It is also important to note that the scope of business modeling may vary. Typically you don't need to model the *entire* business, unless your goal is to re-engineer the organization. The goal of a business modeling effort can span anywhere from understanding the business for which you are building a computer system to making significant business process improvements.

Because more and more business processes are automated by software systems, business modeling is becoming a necessary technique to ensure that automation solutions are adequate.

Business modeling should not be confused with business process engineering. Business modeling is a subset of the various techniques to engineer a business. Business modeling does not imply changing the way you do business. It is simply a technique to visually document what your business does.

Why Business Modeling?

The systems that we build are deployed into organizations that already have business processes in place. Whether you sell a Customer Relationship Management System, a Library Information System, a Billing System, OLAP (Online Analytical Processing) system, OLTP (Online Transaction Processing) system, or any other, that system will be used by an individual during the course of a specific task – interacting with a customer, looking for articles, billing a customer, etc.

How can you ensure that your system will provide value if you do not understand how it will be used, who will use it and in what circumstances it will be used?

To ensure that we are building customer-oriented solutions (i.e. information systems that satisfy our customers), we must not overlook:

- the environment in which these systems will work,
- the roles and responsibilities of the employees using the system,
- the "things" that are handled by the business, as a basis for building the system.

It is from this internal view of the business that you can better understand the system to build.

This benefit becomes essential when building e-business systems. By this we mean automating our business processes to make them available to customers over the Internet. The rising demand for business modeling as a part of any software development effort during the last two years can be attributed to the necessity of business modeling for building successful e-business systems. Successful e-business applications are ones that have accurately emulated real-world processes and leveraged new technologies and new channels. These applications allow both, a company to operate more efficiently and a customer to receive better service. If the system fails to accurately emulate the real world (i.e., deliver the same end product as the real-world), then quickly the system will be unsuccessful (i.e., nobody will use it).

One of the great benefits of business modeling is to elicit *better* system requirements, requirements that will drive the creation of information systems that actually fit in the organization

and that will indeed be used by end-users. Once business models are specified, establishing relationships between system requirements and the business models enables business change to be incorporated in the definition of the system to build.

If you consider software development to have two primary drivers, cost and quality, business modeling helps these factors by providing the following:

Cost Drivers:

- *Get it right the first time* – The development of a business model helps ensure that time is not wasted on the development of a business system by having unnecessary development iterations.
- *Access to business users* – Often the crucial business users are not present at the time an important business question comes up. Business models will not solve this problem, but they will help to reduce it, by providing an adequate description of the business for developers. This will reduce the amount of time squandered by developers waiting for business comments or feedback.
- *Test early* – The cost of finding ‘bugs’ later in the lifecycle is much greater than if those problems are identified early. Note that ‘bugs’ are not only software-related, there are (many) requirements bugs too and these are much harder to find and more expensive to fix than software defects. By providing a business perspective to the software development process ‘bugs’ may be highlighted or avoided much earlier in the lifecycle.
- *Support for iterative development* – The nature of software development requires the adoption of an iterative lifecycle. What defines an iteration or increment is a decision made by the project manager based on risk, dependencies and business functionality. By having an understanding of the value of the business elements the practitioner can make informed decisions to provide return on investment earlier.

Quality Drivers:

- *Business architecture and software architecture* – The structure of the software must absorb change (i.e. it must be maintainable). Change is often directed from the business - therefore the software must be structured in a consistent manner to the key business abstractions. Business modeling provides those abstractions and ensures all project stakeholders understand them.
- *Development of accurate systems* – Business modeling utilizes a common framework and language that is understood by both, programmers and business domain experts, resulting in systems that effectively address business needs.
- *Establish a cornerstone for parallel system development* – Using a common framework enables system definition, design, development, and testing to occur in parallel based on the same set of information.
- *Build systems harmoniously* – By leveraging business modeling, all stakeholders involved in the development process have a common bond from which quality processes and teams can be established.

- *Robust system development* – Business modeling accurately depicts real-life processes in a manner that ensures complete coverage. Coverage includes proper requirements definition, accurate development against the requirements, and full testing coverage for quality assurance. The end-result is a complete quality system.

If business models are not produced you run the risk that developers only give superficial attention to the way business is done. They may do what they know best, which is to design and create software, without taking the business processes into account. This unfortunate situation could result in a waste of costly business process engineering effort. Additionally, there is an increased risk that the systems that are built do not support the true needs of a business.

Business Modeling with the UML

Modeling the business is not something new. You may be familiar with workflow charting techniques such as the IDEF notation. However, a problem that more frequently surfaces is how difficult it can be for business analysts to effectively communicate their findings to the system folks on the team.

For over 5 years now, the Unified Modeling Language (UML) has been providing system architects, working on system analysis and design, with one consistent language for specifying, visualizing, constructing, and documenting software systems.

The UML is used to model a broad range of systems (software systems, hardware systems, databases, real-time systems and real-world organizations). Rational support for business modeling uses the UML to model real-world organizations and their processes.

By sharing a single notation and a single tool across system and business, business analysts and system analysts can better communicate their needs, which is key to building a system that solves the customers' problems.

The table below describes the business modeling notation and icons used in the UML. The business-specific icons help distinguish business elements from system elements in the visual models.

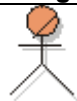
Modeling icon	Name	UML Definition
	Business actor	Someone or something, outside the business that interacts with the business.
	Business worker	Role or set of roles inside the business. A business worker interacts with other business workers and manipulates business entities.
	Business entity	A "thing" handled or used by business workers.
	Business use case	A sequence of actions a business performs that yields an observable result of value to a particular business actor. (In this paper, synonymous with business process)
	Business use case realization	A collection of diagrams to show how the organization elements (workers and entities) are deployed to support a business process.
	Organizational unit	A collection of business workers, business entities, relationships, business use-case realizations, diagrams, and other organization units. Used to structure the business (object) model by dividing it into smaller parts.

Table 1: UML business modeling notation

The value of using the UML to model a business is to reuse an established standard notation (the UML) to provide a common language and potentially a common tool (a UML visual modeling tool) for all modeling needs.

The UML provides different diagrams. Each UML diagram provides a different view of the business:

- use case diagrams describe the business **context**.
- activity diagrams describe **behaviors** in the business, or business workflows.
- class diagrams describe the static **structure** in the business.
- interactions diagrams (sequence diagrams and collaboration diagrams) describe the dynamic **interactions** between employees and things that they manipulate. Thus they indicate how the behaviors described in activity diagrams are realized.

Use-Case Diagrams

The first step in business modeling is to define the interaction between the entities outside of your business (suppliers, customers, partners, colleagues in departments interacting with the department you model, etc), and your business processes. This is documented in a **business use-case diagram**. A business use-case diagram visually represents the interaction between the primary services (business use cases) your business provides and those to whom the services are provided (business actors). See Figure 1.

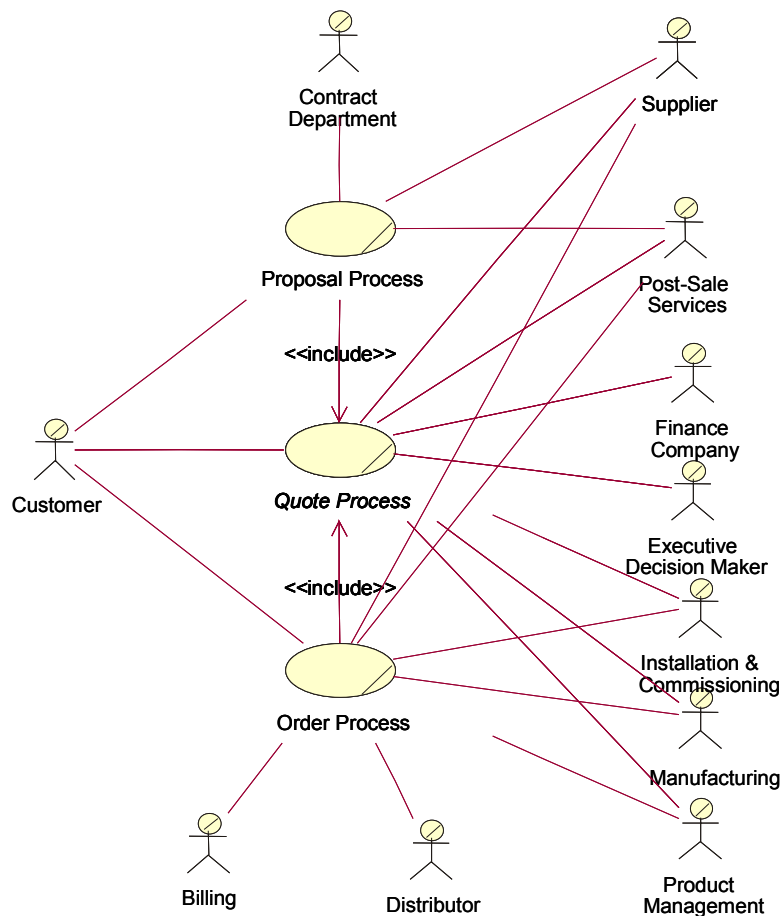


Figure 1: Example of a business use-case diagram.

From a business use-case diagram, the workflow of each business use case is further detailed to document the sequence of activities comprising the entire business process represented by the business use case. There are two ways to document the business workflows. Activity diagrams are the UML diagram of choice to document business workflows graphically. You can also document business workflows textually in a document. Even though those two methods sound redundant at first, experience has shown that there are benefits in documenting business workflows both textually and graphically. Different stakeholders like different formats, and expressing the business workflow textually as well as graphically in a diagram helps flush out the activities that comprise the business workflow.

Activity Diagrams

Activity diagrams as defined in the UML represent the dynamics of a system by expressing flows. They are basically flow charts that are used to show the workflow of the system.

A **business activity diagram** provides a graphical way to document a business workflow. It provides a simple and intuitive illustration of:

- what happens in a workflow,
- what activities can be done in parallel,
- whether there are alternative paths through a workflow.

Activity diagrams also describe the **roles and areas of responsibilities** in the business – in other words who is responsible for doing what in the business. Roles and areas of responsibilities are documented as columns (UML swimlanes) in the activity diagram. Swimlanes show which business workers participate in the realization of the workflow

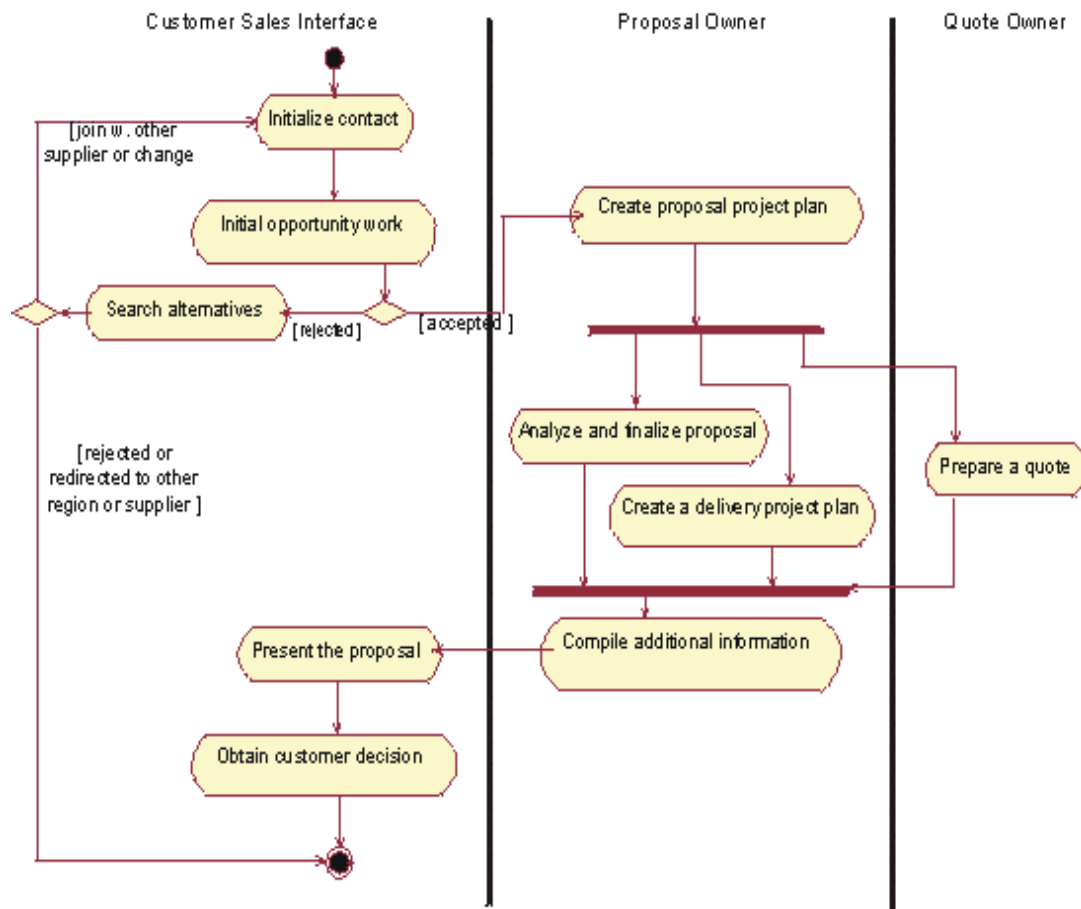


Figure 2: An Activity Diagram documenting how the business performs a *Proposal* process, with three areas of responsibilities (Customer Sales Interface, Proposal Owner, and Quote Owner).

Some business activity diagrams show swimlanes and others do not. Should you add swimlanes to all activity diagrams? No. A common way to organize the activity diagrams for a business process is to have one overview activity diagram, without swimlanes, that covers the entire business use case workflow, and where "macro activities" are shown. Such activity diagrams would be included in the business use case model. Then, a more detailed set of diagrams would

use swimlanes to represent the business workers involved in the process, and the activities at the business worker level. The detailed activity diagrams are included in the business object model.

Besides areas of responsibilities and specific activities, business activity diagrams can also show **business entities** being manipulated in the activities. Business entities represent objects that are either created, updated or used during the course of the business activity. Things like customer profile, paycheck, order, etc. If you model the business with the goal of eliciting better system requirements, some of these business entities will become analysis classes in your system design, in essence reusing the business artifacts in the design of the system.

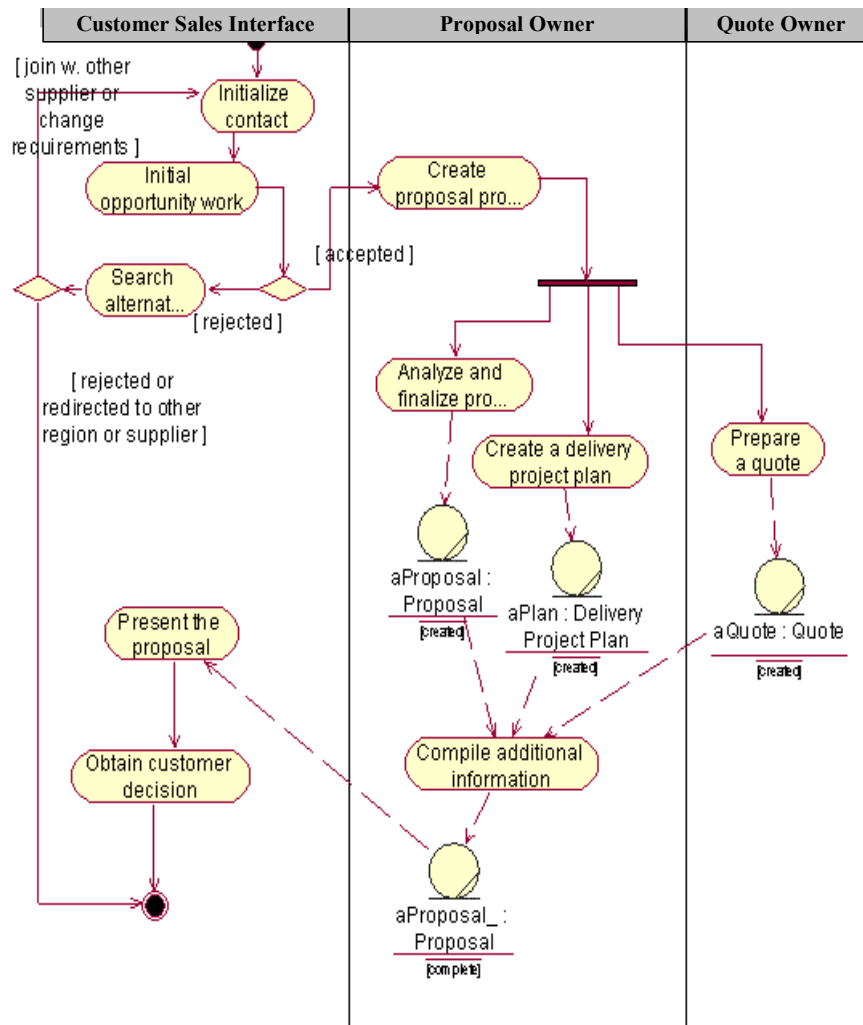


Figure 3: An activity diagram showing business entities (aProposal, a Quote, aPlan) and their states (created/complete).

Class Diagrams

Business class diagrams document the internal structure of the business. Each class in this diagram either represents a **business worker** (employee of the business) or a **business entity** (a 'thing' that the business manipulates). The purpose of business class diagrams is to document

the relationships between business workers and business entities. It provides a way to visualize *who* interacts with *who* and *who* is responsible for *what*.

Business class diagrams are used for two main purposes:

- To show **which** business workers and business entities are collaborating to implement a business process.
- To show **static structure** and relationships among business entities. A class diagram would be used to represent the org chart of a business (using organization units and business workers).

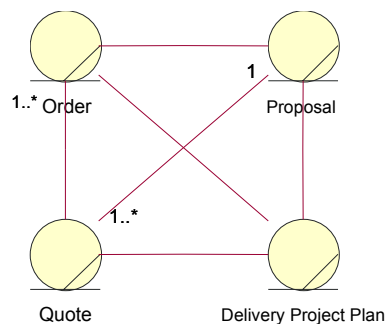


Figure 4: A class diagram showing relationships among business entities.

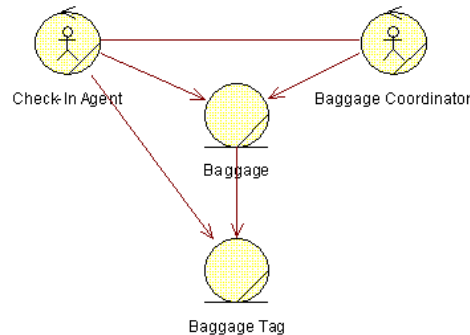


Figure 5: A class diagram showing relationships between business workers (*Check-in Agent*, *Baggage Coordinator*) and business entities (*Baggage*, *Baggage Tag*), showing that the Check-In Agent has the knowledge of a Baggage Tag, but the Baggage Coordinator does not.

Interaction Diagrams

There are two kinds of interaction diagrams, namely **sequence** and collaboration diagrams. The information on these diagrams is the same; they are just presented differently. As a general rule,

collaboration diagrams are easier to start off with, but sequence diagrams are more useful for more complex interactions.

A **business collaboration diagram** documents how business workers and business objects actually interact to perform a business function. Collaboration diagrams show the messages exchanged between business workers and business entities during the course of realizing a business use case (implementing a business process). Collaboration diagrams can clearly indicate areas of improvements by visually showing business workers being responsible for a large number of tasks.

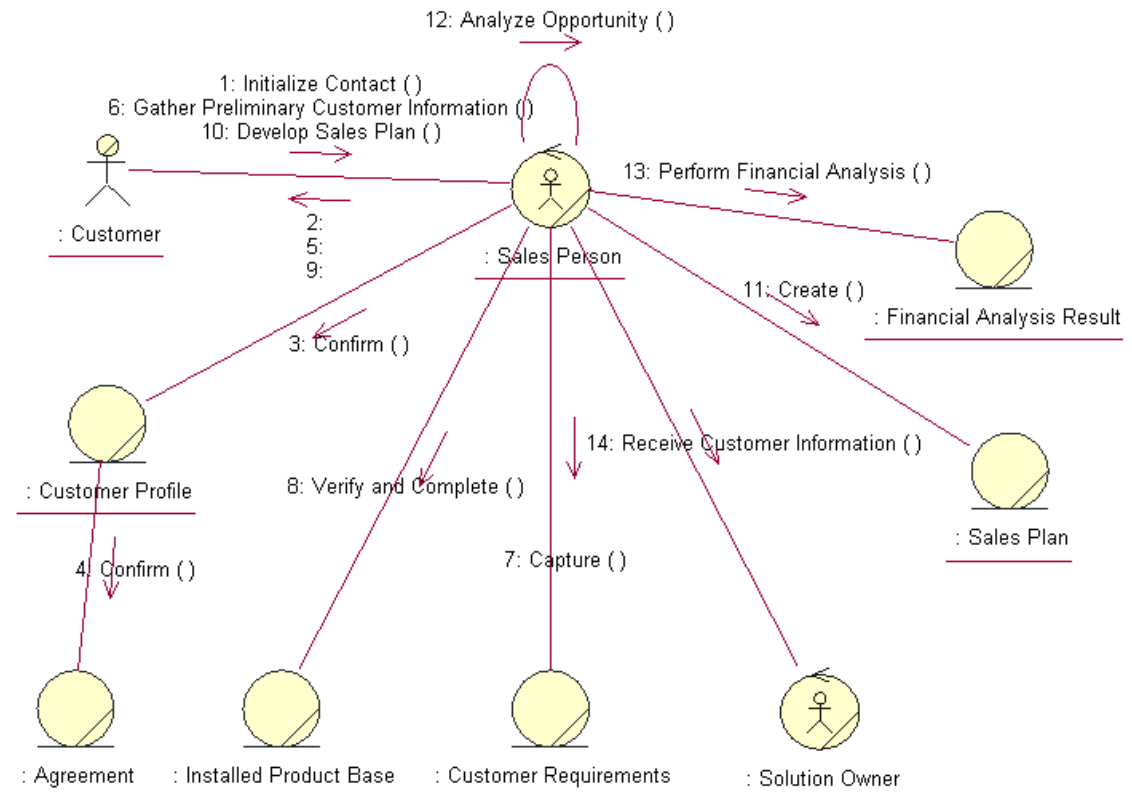


Figure 6: A collaboration diagram showing a view of participating business workers (Sales Person, Solution Owner) and business entities (Customer Profile, Sales Plan, etc) in a *Proposal* process.

The various business diagrams are typically grouped into two business models, one looking at the business from an external perspective, the other looking in:

- the **business use case model**, which looks at the business from an *external* perspective. The business use case model includes a description of the business actors and the business use cases, the interactions between business actors and business use cases (use case diagrams) and high-level activity diagrams to outline the activities for each business use case (high level activity diagrams).
- the **business object model**, which details how business processes are implemented *internally*. The business object model includes a description of the employees of the business (business workers), the 'things' (business entities) that the business manipulates and how the business workers manipulate the business entities to accomplish the business processes (detailed activity diagrams, class diagrams, interaction diagrams).

Business Modeling with Rational Suite AnalystStudio

The business modeling capabilities of Rational Suite AnalystStudio provide a method to represent things like business workflows, responsibilities, and deliverables of an organization in visual models. They also allow the system analyst to track business objectives - and their relationships to system requirements, so that as business objectives change, the system definition takes that change into account, resulting in a system that, when delivered, indeed solves customers' needs.

In the next section, we go into more details on the specifics of the business modeling support in Rational Suite AnalystStudio. That support includes:

- Business modeling best practices (Rational Unified Process)
- The market-leading visual modeling tool for UML modeling (Rational Rose)
- The market-leading requirements management tool to track changes to the business information as they impact system requirements (Rational RequisitePro)

Note that these three tools are also included in Rational Suite DevelopmentStudio.

Business Modeling in the Rational Unified Process

The Rational Unified Process (RUP) provides step-by-step guidance on the various activities of a software development project. Since as explained in this paper, business modeling is a technique that helps analyze the problem that the software will solve, RUP includes business modeling guidelines.

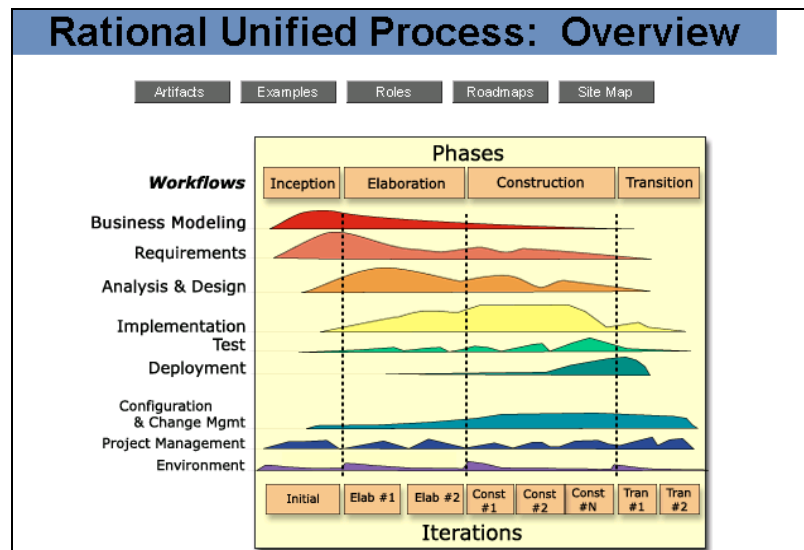


Figure 7: Rational Unified Process Overview.

Business modeling as defined by the RUP is aimed at providing a series of techniques and notation to enable the practitioner in describing and understanding business processes. In particular, the techniques are aimed at describing the business process in such a way that the impact on software systems becomes clear. The business modeling workflow in the RUP can be used as a component in a business process re-engineering effort, a component that concentrates on providing support for the development of the right software.

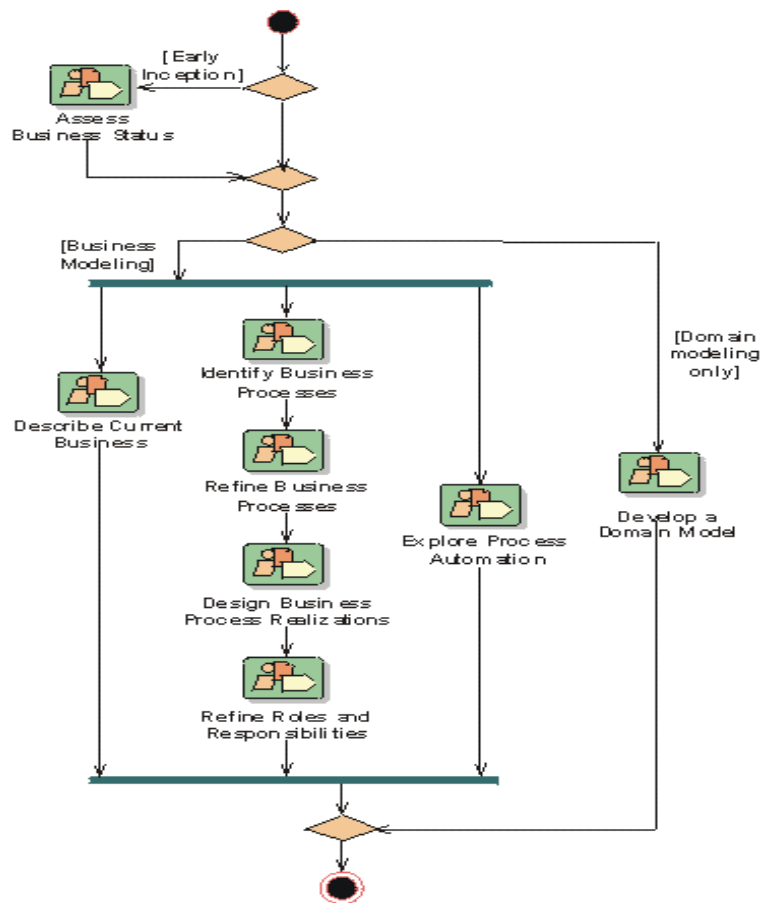


Figure 8: Business Modeling activities in RUP.

Depending on the goal of the business modeling effort, a company would follow different paths through the workflow above. For instance the activity “Develop Domain Model” (Figure 8) would be achieved in a small business modeling effort where you mainly want to document the relationships between the various “things” (entities) that a business may manipulate. You can think of a domain model as a graphical glossary, and its value is to graphically represent relationships and cardinality (multiplicity) between the various things in a business. For instance, documenting the number of saving accounts - or checking accounts each bank customer has may be more meaningfully and accurately depicted in a diagram than in a textual paragraph.

The business modeling activities described in RUP are complemented with tool mentor pages. RUP tool mentors provide specific tools steps to perform these activities when using Rational tools (Figure 9).

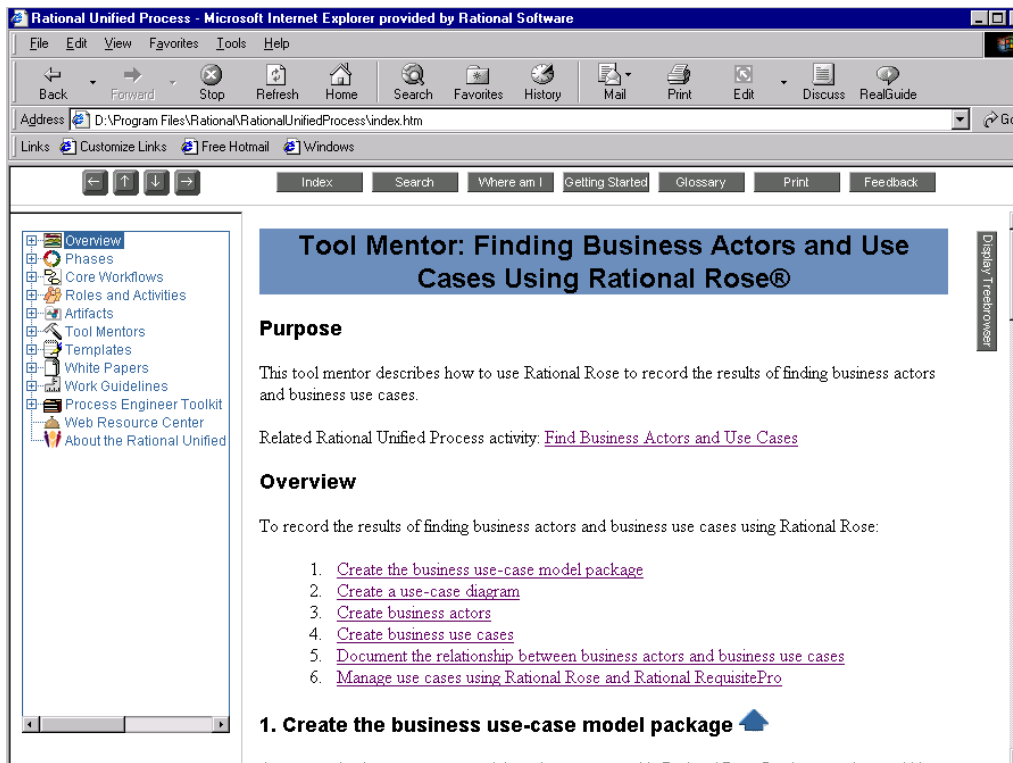


Figure 9: RUP tool mentor for *Finding Business Actors and Use Cases*.

As previously mentioned, some of the discoveries that you make while modeling a business will benefit the quality of the design of the systems deployed.

How does one transfer the information stored in the business model to the system models, so that the system includes the business considerations?

The RUP provides a simple yet powerful algorithm to jumpstart your system design with the information identified in the business models (Figure 10). Specifically, the business entities and business workers will jumpstart the system use case model and design model. By starting the system design from these business artifacts, you can ensure that your design is in touch with your user needs as identified in the business models.

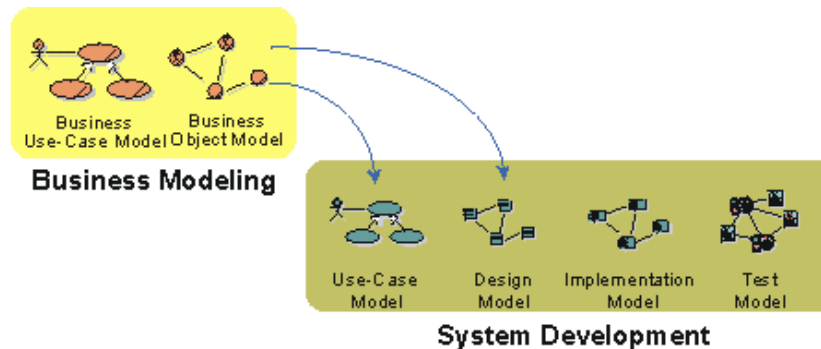


Figure 10: Relation between models of the business and models of a system.

This powerful algorithm simply states that:

- every business worker identified in the business model is a potential system actor,
- every business actor is a potential system actor (if that business actor directly interacts with the system under development, often the case in e-business system)
- every business use case is a candidate system use case (Figure 11).

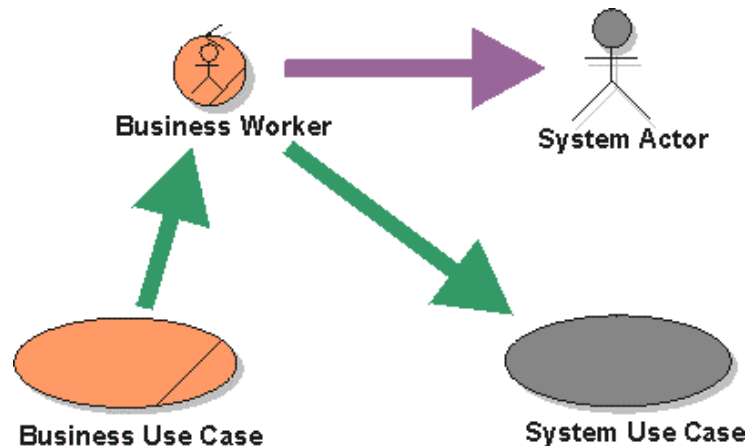


Figure 11: RUP business artifacts potential reuse in system design.

Business Modeling in Rational Rose

Rational Rose is the market leading visual modeling tool. It provides support for any type of UML-based modeling, including system modeling, business modeling as well as data modeling. Rose provides a **single tool** and a **single notation** (UML) for all modeling needs. This unique benefit ensures that business folks and system folks can communicate their requirements clearly and that change in one model is easily visible to all.

The business modeling icons described in table 1 are provided out-of-the-box in Rational Rose.

Using Rational Rose, you describe your business processes using the various UML diagrams described earlier in this paper. Rose even provides a way to auto-convert sequence diagrams into interaction diagrams and vice-versa.

Using UML packages, you organize these diagrams into two packages, one for each business model – the business use case model and business object models. You also organize business actors, business use cases, business workers and business entities into packages.

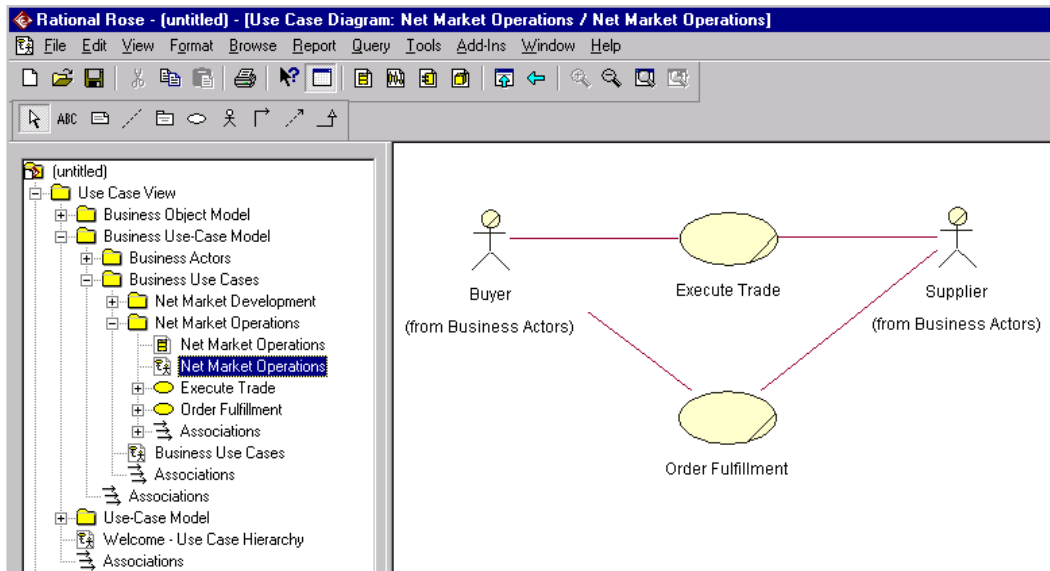


Figure 12: In Rational Rose - Business artifacts organized in UML packages (left) and business use case diagram (right).

You can describe business workflows either using activity diagrams or documents. Describing business workflows in documents allows you to mark business requirements in these documents. To ensure that business requirements can be related to system requirements, Rational Rose integrates with Rational RequisitePro (Rational requirements management tool). This integration automates the creation of business workflow documents directly from the use case diagrams in Rose, simply by right clicking on the use case (Figure 13).

The unique value of this integration is to store key business information specified in the business workflows in the database of system requirements. This ensures that system requirements can be mapped and traced to business considerations.

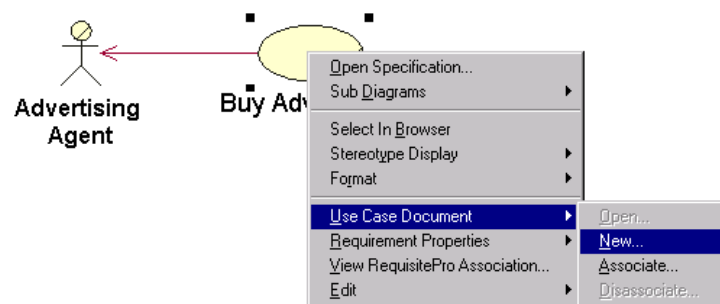


Figure 13: Attaching a business use case document to a business use case in Rose is accomplished by simply right-clicking on the business use case.

That integration also allows users to attach prioritization attributes (requirement properties) to business use cases. For more information on the integration between Rational Rose and Rational RequisitePro, read *Use Case Management with Rational Rose and Rational RequisitePro* at <http://www.rational.com/products/astudio/whitepapers.jsp>.

Within each business use case, an activity diagram can be attached to describe the business workflow graphically. (Figure 14)

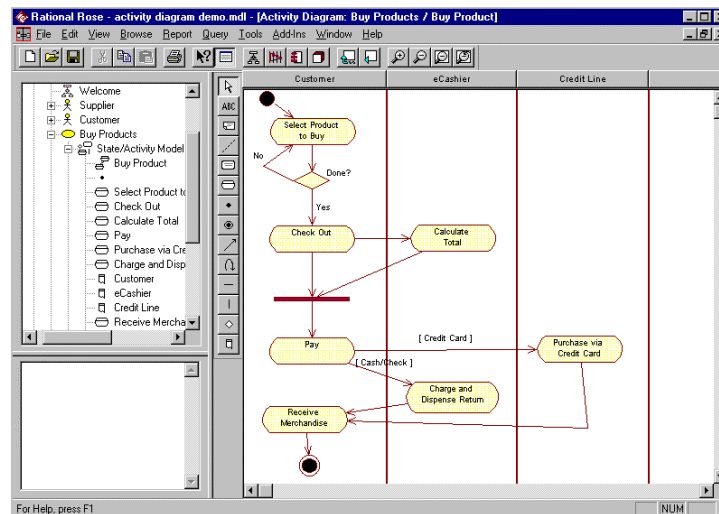


Figure 14: Business workflow documented in an activity diagram in Rational Rose.

All UML diagrams described earlier are supported in Rational Rose as part of the out-of-the-box UML support.

Business Modeling in Rational RequisitePro

As mentioned earlier, business workflows are often documented both textually as well as graphically using activity diagrams. One of the values to documenting business workflows textually is the ability to track and trace the business information to the system requirements level.

Rational RequisitePro provides a document template to consistently document business workflows in Word. That document template also includes a section to record the other textual properties of a business use case, such as business goals, performance goals, risks, etc.

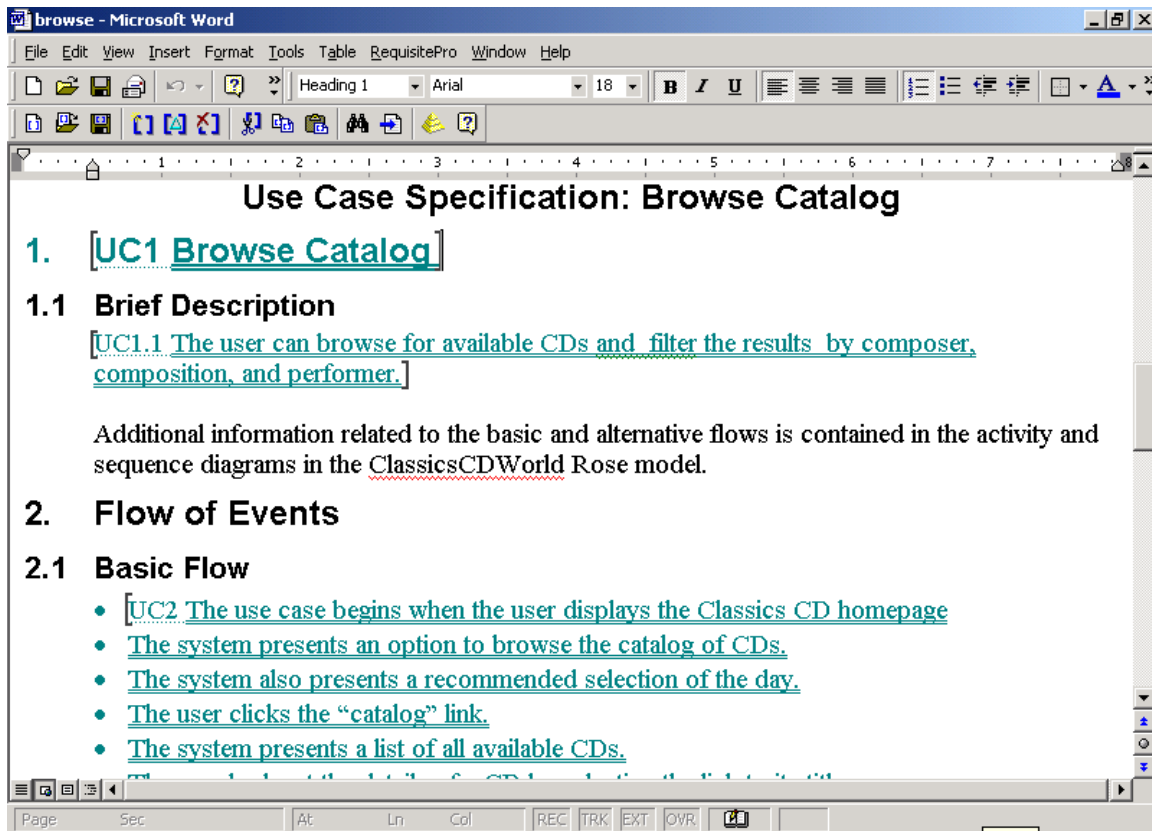


Figure 15: Business use case document in Rational RequisitePro with marked business use case requirements.

Once the business activities are documented in the Word document, critical business information in the document can be marked as business requirements. Business requirements are stored together with system requirements in the Rational RequisitePro requirements database.

The value of storing business information in the system requirements database is the ability to organize business needs and prioritize them (Figure 16).

Requirements:	Priority	Status	Difficulty	Stability	Location
BUC2: Auctions and Reverse Auctions require that...	High	Approved	Medium	Medium	Execu
BUC4: The Buyer reviews available product using...	High	Approved	Medium	Medium	Execu
BUC7: Net Market Business Development	High	Approved	Medium	Medium	NetMa
BUC1: Auction, Reverse Auction and Bid/Ask trading...	Medium	Approved	Medium	Medium	Execu
BUC3: Order Fulfillment includes determining...	Medium	Approved	Medium	Medium	Produ
BUC5: Chat/Collaboration	Medium	Approved	Medium	Medium	Datab
BUC6: Execute Trade	Medium	Approved	Medium	Medium	Execu
BUC8: Order Fulfillment	Medium	Approved	Medium	Medium	Produ
BUC9: Define Net Market Type	Low	Approved	Medium	Medium	Define
* <Click here to create a requirement>	Medium	Approved	Medium	Medium	Datab

Ready 9 requirements

Figure 16: Prioritizing business needs in Rational RequisitePro attribute matrices.

Additionally it allows you to relate system requirements to business considerations. (Figure 17).

Requirement traceability (i.e. the ability to maintain relationships between requirements and related artifacts) has two main values:

- to ensure requirement coverage – Are the system requirements fulfilling all business needs?
- to assess the impact of requirement change - If the business considerations change, how does the system take this change into account?

After business requirements have been identified in Rational RequisitePro, traceability links between business requirements and system requirements are established to ensure business needs coverage and that the impact of changing business needs can be quickly recognized.

Views of requirement traceability can be easily queried to find holes in requirements coverage. Each high-level system requirement (feature requirement) should be traced to at least one business requirement (else why does the system have this requirement?) (Figure 18).

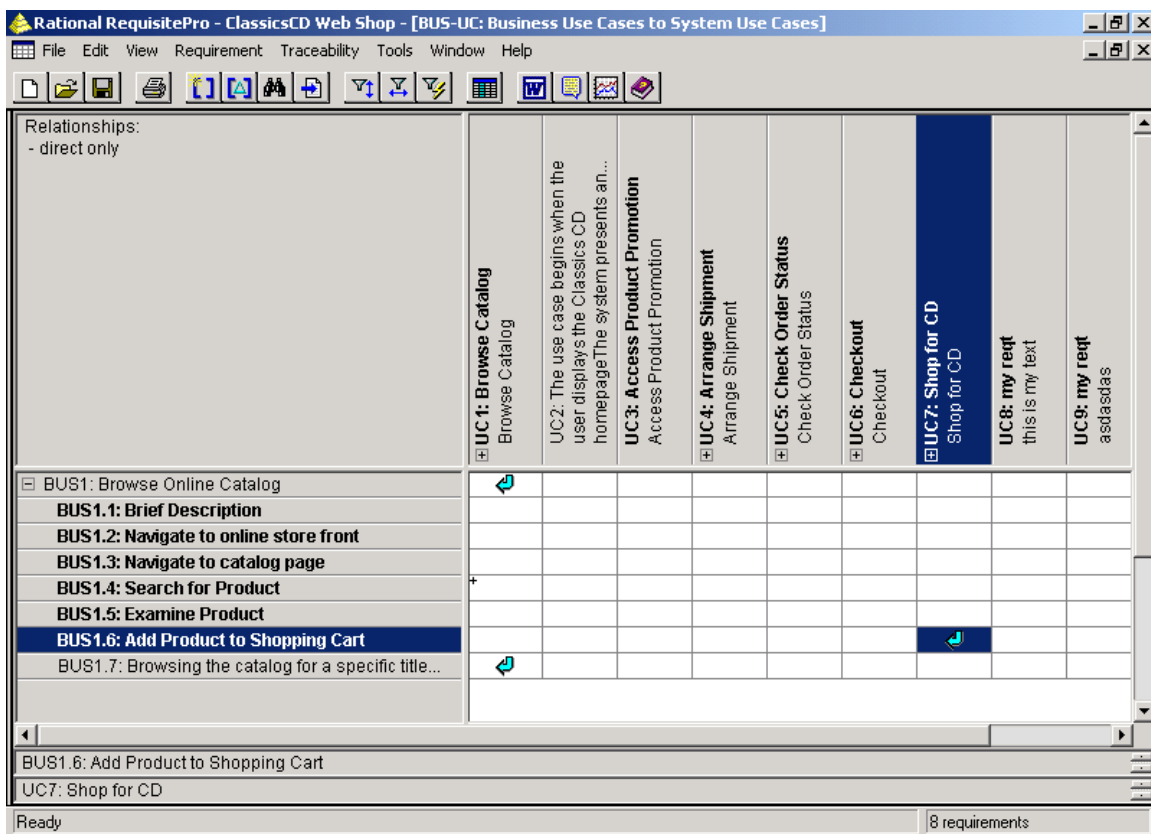


Figure 17: Tracing system requirements to business needs using Rational RequisitePro traceability matrices.

If a business requirement changes as we are building the system, Rational RequisitePro quickly indicates what part of the system definition needs to be updated to ensure that we eventually still deliver a valuable system.

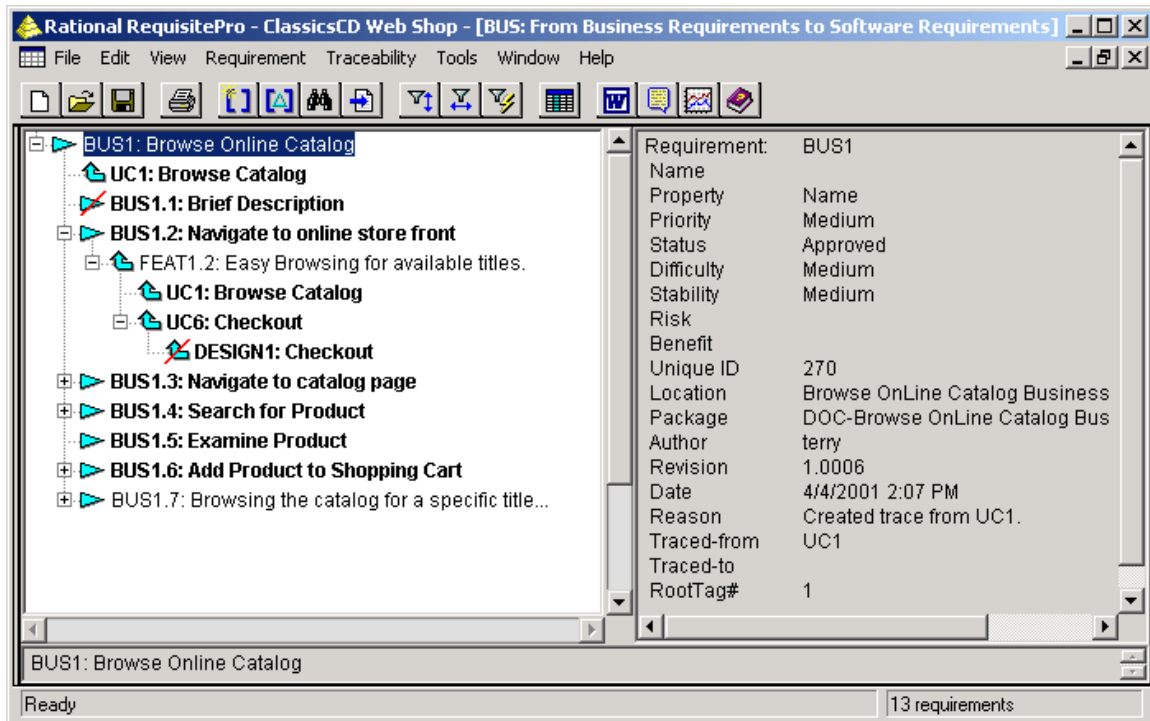


Figure 18: Rational RequisitePro traceability tree showing suspect relationships (red slash) between business requirements and feature requirements.

Conclusion

In this paper, we discussed Rational support for business modeling using Rational Suite AnalystStudio and the UML. That support includes:

- RUP business modeling guidelines that follow proven business modeling methodologies,
- The market leading visual modeling tool (Rational Rose) using common notation (UML) across the team's software developers and data modelers,
- The market leading requirement management tool (Rational RequisitePro) to maintain relationships between business requirements and software requirements, thereby ensuring that as business requirements change, software requirements are synchronously impacted.

Using Rational's approach to business modeling, which reuses the use case modeling concepts from software engineering, business considerations can more easily be taken into account when building software. This in turn ensures that the systems we build are aligned with the business processes in which they will be deployed.

Additional Reading

For a more in-depth discussion of the concepts and principles presented in this paper, consult:

- *The Object Advantage--Business Process Reengineering with Object Technology* by Ivar Jacobson
- *Managing Software Requirements – a Unified Approach* – chapter 5 on Business Modeling - by Dean Leffingwell and Don Widrig
- *Business Modeling with UML—Business Patterns at Work*, Hans-Erik Eriksson and Magnus Penker
- *Advanced Use Case Modeling: Software Systems*, Frank Armour, Granville Miller, Iva Jacobson
- The Rational Unified Process - Business Modeling Discipline
- The Rational Edge (<http://www.therationaledge.com>)
<http://www.therationaledge.com/rosearchitect/mag/archives/fall99/index.html>
Business Modeling with the UML, Hans-Erik Eriksson and Magnus Penker
Business Modeling and E-Commerce, Magnus Christerson and Simon Johnston
Activity Diagrams, What they Are and How to Use Them, Maria Ericsson

http://www.therationaledge.com/content/mar_01/m_uml_jh.htm
Introduction to Business modeling with UML – Jim Heumann -
- UML specification at <http://www.rational.com/uml/index.jtmpl>
- *The UML Reference Manual* by Jim Rumbaugh, 1998

Glossary

Actor	Someone or something, outside the system that interacts with the system.
Analyst	Person/people who must interpret the user needs, and communicate those needs to the entire team. Job titles of those with Analyst responsibilities include, but are not limited to, Systems Analyst, Business Analyst, Project Manager, Product Manager, Systems Engineer, Requirements Engineer, Marketing Manager, etc.
Artifact	A piece of information that is used or produced by a software development process. An artifact can be a model, a description, or software.
Business Actor	Someone or something, outside the business that interacts with the business.
Business Modeling	Encompasses all modeling techniques you can use to visually model a business.
Business Object Model	The business object model is an object model describing both the elements of the business (workers, artifacts and other systems) and how those elements work together as a system to support the business processes of that business.
Business Process	"A collection of activities that take in one or more kind of input and create an output that is of value to the customer" <i>Hammer and Champy - "Reengineering the corporation a manifesto for business revolution" 1993</i>
Business Use Case	A business use case defines a sequence of actions a business performs that yields an observable result of value to a particular business actor. A business use-case class contains all main, alternate workflows related to producing the "observable result of value". (In this paper, synonymous with business process)
Business Use-case model	A model of the business intended functions. The business use-case model is used as an essential input to identify roles and deliverables in the organization.
Business Use-Case Realization	Business use case realizations show how the organizations elements (workers and entities such as sales clerk and order) are deployed to support that business process.
Business Worker	Role or set of roles inside the business. A business worker interacts with other business workers and manipulates business entities while participating in business use-case realizations.
Class	A description of a set of objects that share the same attributes, operations, methods, relationships, and semantics.
Domain Model	Captures the most important types of objects in the context of the business domain. The domain objects represent the entities that exist or events that occur in the environment in which the system works. The domain model is a subset of the business object model.
Requirement	Describes a condition or capability to which a system must conform; it can be either derived directly from user needs, or stated in a contract, standard, specification, or other formally imposed document. A desired feature, property, or behavior of a system.
Stakeholder	An individual who is materially affected by the outcome of the system. End users are often thought of as the primary stakeholders, but other stakeholders, such as shareholders and executive management also have a stake in the project.

About Rational Suite AnalystStudio

The Rational Suite family of products were created to unify software development teams while optimizing the effectiveness of individual practitioners on the team.

Rational Suite AnalystStudio is optimized for the “analyst role” on the team, representing the individuals responsible for eliciting stakeholder needs and documenting the requirements of the solution intended to solve these needs. Rational Suite AnalystStudio includes:

- **Rational Rose Data Modeler Professional Edition**, the market-leading tool for visual modeling
- **Rational RequisitePro*** for team-based requirements management
- **Rational ClearQuest*** for comprehensive defect and change tracking
- **Rational Unified Process*** providing a web-enabled set of software engineering processes
- **Rational SoDA*** for software documentation automation
- **Rational TestManager*** for easy team access to testing plans and results
- **Rational ClearCaseLT*** for software configuration management
- **The Rational Developer Network*** helps you learn and use Rational tools and methodologies through online access to training, articles, white papers, documentation, scripts, discussion forums, and more
- **Rational ProjectConsole*** for metrics and reporting within a customizable project Web site

*Common to all Rational Suite products, these tools are part of the *Team Unifying Platform* serving to unify the team with a common set of tools pertinent for all roles on the software team.

The Rational Suite family of products includes:

- **Rational Suite AnalystStudio** – for analysis and requirements management
- **Rational Suite ContentStudio** – code and content management for Web development
- **Rational Suite DevelopmentStudio** – model-driven development on Windows or Unix platforms
- **Rational Suite DevelopmentStudio RealTime** – real-time and embedded system development
- **Rational Suite TestStudio** – functionality, reliability, and performance testing
- **Rational Suite Enterprise** – for the practitioner who spans multiple areas

About Rational Rose

Rational Rose uses a common, standard modeling language (UML) accessible to non-programmers wanting to model business processes as well as to programmers modeling application logic.

Rational Rose allows the analyst to create a graphical representation of requirements, using use-case diagrams and activity diagrams to increase the understanding of the system to build, prior to drilling into requirements details.

About Rational RequisitePro

Rational RequisitePro is a flexible, easy-to-adopt requirements management tool used for documenting, organizing and managing requirements throughout the development lifecycle. RequisitePro increases the likelihood to deliver quality systems on time and on budget by unifying

all team members — project managers, QA managers, testers, developers, etc. — by communicating and managing all project requirements.

About the Rational Unified Process

The Rational Unified Process™ (RUP) is a Web-enabled software engineering process that enhances team productivity and delivers software best practices to all team members. RUP is an easy-to-use online mentor that makes process practical by providing extensive guidelines, templates, and examples for all critical e-development activities. RUP is a customizable framework, which can easily be adapted to the way you work. It is tightly integrated with Rational tools, allowing development teams to gain the full benefits of Rational product features, the Unified Modeling Language (UML), and other industry best practices.

About Rational ClearQuest

Rational ClearQuest is a flexible change and defect tracking system that scales to the needs of any project, for any size team. ClearQuest shortens development cycles by unifying all team members — project managers, testers, developers, and analysts — in managing software development change.

About Rational SoDA

Communicating project information can be time consuming and difficult. Project lifecycle documentation and reports are often produced haphazardly, or are neglected altogether because of the effort and time involved in creating and maintaining them. Rational SoDA overcomes these issues by automating the creation and maintenance of comprehensive project documentation and reports. Unlike manual methods, SoDA generates complete documentation more easily and with greater consistency by automatically extracting data from multiple tool sources.

About Rational TestManager

Rational TestManager is the one place to manage all testing activities - planning, design, development, execution, and analysis. TestManager ties together testing with the rest of the development effort joining your testing assets and tools to provide a single point from which to understand the exact state of your project.

About Rational ClearCase-LT

Rational ClearCase LT is a software configuration management solution for project workgroups. ClearCase LT offers Unified Change Management (UCM), Rational's "best practices" based, out-of-the-box process enabling project teams to manage change at the activity level and control workflow. Enterprise customers requiring distributed servers or server replication can upgrade to ClearCase from ClearCase LT without changing their process, data or the way they work.

About The Rational Developer Network

The Rational Developer Network helps you learn and use Rational tools and methodologies through online access to training, articles, white papers, documentation, scripts, discussion forums, and more.

About Rational ProjectConsole

Rational ProjectConsole is a browser based reporting environment that provides development teams with progress & quality metrics reports based on data that is automatically collected from Rational Suite products, as well as, report generation directly from Suite product artifacts.

Rational®

the software development company

Dual Headquarters:

Rational Software
18880 Homestead Road
Cupertino, CA 95014
Tel: (408) 863-9900

Rational Software
20 Maguire Rd
Lexington, MA 02421
Tel: (781) 676-2400
Toll-free: (800) 728-1212
Tel: (408) 863-9900
Fax: (408) 863-4120

E-mail: info@rational.com

Web: www.rational.com

For International Locations: www.rational.com/worldwide

Rational, Rational logo, Rational the e-development company, Rational RequisitePro, Rational Rose, Rational ClearQuest, TestManager, and Rational Suite among others are trademarks or registered trademarks of Rational Software Corporation. References to other companies and their products use trademarks owned by the respective companies and are for reference purposes only. ALL RIGHTS RESERVED. Made in the U.S.A.

© Copyright 2001 by Rational Software Corporation.

Subject to change without notice.