

Use Case and Interviewing Techniques for Focused Requirements Capture



WEST POLE

A Whitepaper by West Pole, Inc.
© 1996-2005 All rights reserved.

Many systems rolled out in corporate America today are later found to be somewhat faulty – either they don't work exactly as the users envisioned, or there are some cumbersome aspects to the work process, or the initial design was too short-sighted or inflexible for a changing business climate.

When development teams can spend the time needed to capture the business requirements using a process such as RUP, it ensures a much better fit between the production system and the business needs.

You can extend the Rational Unified Process[®] methods for eliciting requirements by fine-tuning your use case interview skills – and borrowing techniques from the field of usability.

This whitepaper walks you through the use case development process, guides you around common use case pitfalls, and provides useful tips for forming your use cases.

Background: Use Case Terminology

If you are new to developing use cases, you should become familiar with some basic terminology.

A **requirement** is a condition or capability to which the system must conform. **Requirements management** is a systematic approach to:

- Finding, eliciting, organizing, documenting, and managing requirements
- Establishing and maintaining agreement between the customer/user and the project team on the changing requirements

An **actor** is someone or something outside the system that interacts with the system. Primary actors have a goal when using the system, while secondary actors *support* a goal on the system, instead.

The **use case** itself is a description of a set of sequences of actions (including variations) that a system performs, yielding an observable result *of value* to an actor.

Use cases have an inherent value to the business organization because they give context for the system requirements, are easy to understand, and facilitate agreement with internal and external customers.

Hunting and Gathering Your Requirements

You have just been assigned to a new project, the kickoff meeting was this morning, and you have a deadline. Your role on the project is to talk to users and gather requirements. What's the best way to go about doing that?

It's time to find your pith helmet and put together a plan for hunting and gathering your requirements.

Basic Steps for Gathering Requirements¹

There are two phases for gathering requirements in the Rational Unified Process: analyzing the problem and eliciting stakeholder needs.

First, Analyze the Problem

The first phase for gathering project requirements is to think through the business problem at hand.

1. Gain agreement with the customer on the definition of the problem, and its causes.
2. Identify the project stakeholders (anyone who is materially affected by the outcome of the system).
3. Identify all constraints, such as environmental, political, technical, economic, system, or feasibility constraints.
4. Define system boundaries, in terms of user population, existing legacy systems to be leveraged, interfaces, etc.
5. Create a problem statement as part of the Vision Document.

Then, Elicit Stakeholder Needs

After the business problem has been analyzed, it's time to ferret out the needs of the project stakeholders.

1. Hold a requirements workshop and/or brainstorming sessions for stakeholders.
2. Identify use cases, create an early use case diagram with brief descriptions to give you a roadmap of the proposed system.

¹ A summary of the Rational Unified Process for requirements gathering, part of Rational Software's best practices.

3. Conduct user interviews.
4. Use questionnaires to gather data from a large user population, or to get more quantitative data (does not replace user interviews).
5. Role playing with interviewees (particularly useful if you cannot directly interview users – have the stakeholders role play instead).
6. Business modeling, which is helpful for more complex projects, or for situations where the business process is poorly documented.

Working with the “Users in the Mist”

Dr. Dian Fossey was a zoologist specializing in the field study of gorillas – Hollywood depicted her life story in the movie, “Gorillas in the Mist”, starring Sigourney Weaver.

Your job, as a requirements analyst and use case developer, is to apply and adapt those same field study techniques (now employed by usability specialists) to the task at hand.

The benefit of using an *observational* approach over an *informative* approach is that people often *say* what they do on the job in a very different manner from how they actually *do* their job.

Your goal is to capture requirements and use case data as impartially and accurately as possible, ensuring a better end result for the business.

Contextual Interviews

Lifted from anthropological field study and usability best practices, the contextual interview technique can be an invaluable tool.

The benefit of using an observational approach over an informative approach is that people often say what they do on the job in a very different manner from how they actually do their job.

In a contextual interview, the interviewee pulls out artifacts from their work during the interview. This typically results in a much richer explanation of the process, as the interviewer and interviewee walk through the artifacts.

To conduct a contextual interview:

- Have the user show you what they do in their work environment – perhaps the user walks you through how they create a weekly sales report, using last week’s report artifacts.
- Look for extraneous sources of information:
 - **Cheat Sheets for Data Entry** – for example, a user might retrieve an example form from their desk drawer during data entry.

- **Post-It Notes** – for example, when the user has shipping code values on a post-it note stuck to the monitor.
- **Inquiries to Co-Workers** – notice to whom they talk during the process (or ask in which situations a co-worker needs to be consulted).
-

Case Study: Contextual Interviews

At a recent contextual interview conducted by West Pole to develop use cases, several of these tools were seen. While an interviewee was walking the analyst through how she performed some data validation, she rattled a business rule off the top of her head. When asked how she knew that, she pulled out a sheet of paper from a notepad on her cluttered desk and pointed to the business rule.

Another user in the same department performed a different step in the process. Part of her job was to enter the same data through a web interface. The interface was laid out differently than the form-based data she was working from. After inquiring how the user knew what data went in what field, the user opened her desk drawer and pulled out a printout of the web interface, with notes mapping the fields to the form she uses.

A third user in the department had the task of assigning a product to a vendor by entering the vendor's five digit code. As she talked the analyst through her work process, she pointed to a post-it note on her monitor containing a vendor list and their corresponding codes.

Think Out Loud Techniques

Usability practitioners also frequently use the “think out loud” technique during an interview.

Have the interviewee vocalize their thoughts, as they work with the manual process or existing system. As they work and talk out loud, explaining what they're doing, keep them talking by asking “what if” questions:

- “What if the data doesn't match?”
- “What if you want to cancel here?”
- “What if you don't have that information?”

These exceptions to the typical work process will become alternate flows in your use case specification.

Typical Use Case Interview Mistakes

Some common use case interview mistakes people make are interviewing the wrong people or interview them in a conference room, instead of at their work space.

Novice use case developers often provide the interviewee with too much information about use cases, or fail to return for additional interviews.

Interviewing the Wrong People

You might hear the phrase “You don’t need to interview them... I know what they do.” If you can’t get direct access to the real users, this can be a serious problem – one that can compromise the validity of your data.

Make sure that management understands this, and is willing to accept this risk (or provide access to the people you need).

If there is a valid business reason why you can’t have access to the users, try to meet with several surrogate representatives (the account manager, the salesperson, etc. who have frequent customer contact), and use role-playing techniques to re-enact the user’s workflow.

Providing Too Much Information

Try to avoid providing a lot of information about use case development, and what a use case is. Users will try to explain their work process in the use case format, instead of how they actually perform the task.

Interview Outside Their Environment

When you sit across a conference room table from the user – you’ve created an artificial “sterilized” environment and will get an equally sterilized description of the work process.

Try to meet the user in their own work environment whenever possible – this probably means going to multiple locations if you have a diverse user base.

Interview People Only Once

Performing a second interview session after reviewing the initial interview will help you identify other scenarios.

You can also review your notes (but not your use case specifications or diagrams) with the interviewee to confirm that you accurately captured the data. Often, a user will remember pertinent details during that data review.

It may help to create a use case outline after the initial interview, to generate a list of followup questions. Have the interviewee review the outline – not only does that ensure its accuracy, but it makes them feel a part of the process as well.

Additional Use Case Pitfalls

After you've conducted your interviews, you're ready to develop the actual use cases. Try to avoid these pitfalls as you work.

Confusing Use Case Names

Whenever possible, name each use case with a clear and simple name. Use the "verb plus noun" naming convention, and choose active verbs. Avoid vague verbs, such as "do" or "process".

For example, instead of a use case named "Sales Data Processed", use the name "Create Sales Reports".

Functional Decomposition

Probably the most common problem with use cases is that the use case itself must describe an action that *provides value*. For example the use case "Add Item to Shopping Cart" really belongs as a step in the use case "Order Item." It's important to remember that a use case must provide value – we don't go to Amazon's site just to add items to the shopping cart, do we? No, we go there to order items.

Only One Draft Per Use Case

It is far easier to develop the use case in a series of steps, instead of drafting the use case in one sitting.

First, briefly describe the use case – this is typically one or two short paragraphs. Next, develop an outline for the action in the use case, and finally, create the detailed use case. Don't forget to review the use case elements with users at each step, to ensure accuracy.

Designing Software in the Use Case

According to Rational, the purpose of a use case is "to document a sequence of actions performed by a system that yields an observable result **of value** to a particular actor". A use case specification does not include software design elements.

No matter how tempting it may be to include design details, such as “and then a new window pops up”, save those ideas for the design document instead, or clearly label those comments separately, as design *suggestions*.

Focus on the purpose of a use case – it specifies the *what* but not the *how*.

Including Detailed User Interface Mockups

Your reviewers will be more concerned with the details of the interface than with the use case, but don't jump ahead in the process. Exclude detailed user interface mockups in the use case specification.

If you feel you must include some sort of user interface specification or mockup, make the screen mockups general, and generic. Use fake names for list elements, and don't use typical operating system widgets – let the designers do their job!

When users see detailed screen mockups, they have preconceived notions about what the finished system will look like. Simple line drawings are preferred over actual screens laid out.

Skipping the User Review

Users should review your use case specifications – they may need to be walked through a basic understanding of use cases and what you're trying to capture.

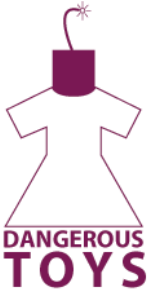
This is a critical step that is often overlooked. If appropriate, get signoff from interviewees on the use case specification.

Use Case Tips & Tricks

Now that you've carefully avoided the pitfalls in developing use cases, you can use these tips and tricks to help you ensure a strong use case specification.

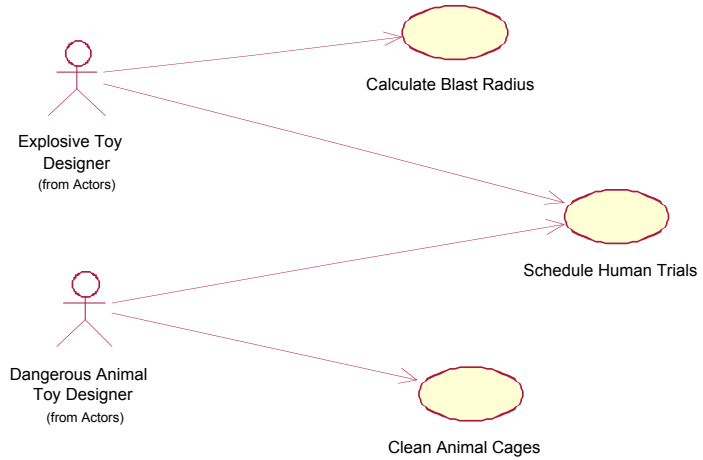
One Primary Actor Per Use Case

If a use case has two primary actors, then create an abstract parent actor to represent the shared roles of the two child actors.



Example: The Dangerous Toy Company

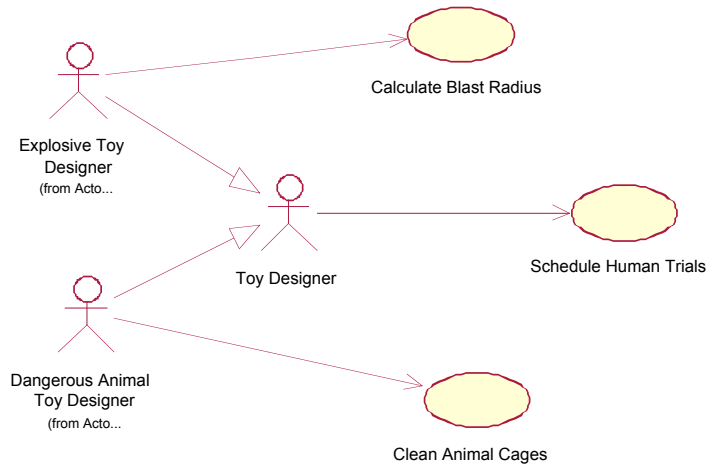
The Dangerous Toy Company



This first use case has two primary actors.

In this type of situation, it's best to create an abstract parent actor to blend the two

The Dangerous Toy Company



roles.

An Alternate to the Alternate

Alternate flows in a use case diagram capture variations from the typical path – such as when errors occur from invalid data entry, or a search result returns no values. The standard use case technique is to place these variations in the alternate flow section of the use case specification.

You may want to differentiate these alternate flows as being either anchored or unanchored:

- **Anchored:** a branch off a specific step in the main flow
For example, when a search returns no data. This alternate flow only occurs from one specific step in the main flow, where the system searches for the specified search criteria.
- **Unanchored:** can occur from any step in the main flow
For example, when the user has the option of canceling the order at any time, as on most e-commerce web sites.

Use Case Outlines

When you develop your use case outline, try to do just a quick draft (less than 20 minutes), first. Then review your work and write the description from the rough outline, afterwards.

You can store your outlines in Requisite Pro, using it as a document repository, which gives you easy access to the early steps in your use case development.

Example: The Dangerous Toy Company

Use Case Outline: Schedule Human Trials

Description

This use case describes the process by which a toy designer schedules the human trials for one of their toys. The toy designer can specify the size of the trial pool, and the area of the body in danger from the toy.

Basic Flow

Log on

Select “Schedule Human Trials”

Specify toy ready for trials

Specify number of subjects requested

Specify area of body in danger

Submit request

View scheduled trials

Anchored Alternate Flows

- 5a. Describe “Other” area of body affected
- 7a. Number of subjects requested is not available

Unanchored Alternate Flows

- A1. Cancel request
- A2. System unavailable

Separate Business Rules

Good software design techniques inform us to avoid repeating business logic in multiple systems. Managing requirements should be no different.

While you can implement the same business rule in many different projects, the core business rules should be stored in one repository. You can use a Requisite Pro project to store business rules, and then trace the application-specific requirements for another project to the corresponding business rules in the central repository.

Case Study: Add a Splash of SoDA

You can automate the use case process, where possible, to reduce headaches for maintaining the use case models.

West Pole recently worked with a client who wanted to merge their use case diagrams and activity diagrams from Rational Rose with the corresponding use case specification from Requisite Pro.

The consultant created SoDA templates that would do this for them, allowing the client to quickly generate full documentation for the system, or a system subset, depending on the report and sections requested.

Conclusion

While Rational has given us many options for eliciting requirements, each business situation is different, and it’s often wise to build on the foundation to suit your situation.

When reviewing your use case techniques, remember that other disciplines, such as usability, also use requirements capture and interview techniques that can be employed.

Adopt techniques and adapt methods to fit your unique situation – and remember that while there’s no right way to format a use case, there are many pitfalls that can be avoided.

Above all, keep in mind that use case specifications are not design documents – they are focused tools for requirements management. With some thoughtful planning and monitoring, the requirements gathering and use case development process can go quite smoothly and ensure buy-in from end users along the way.