419 Hudson Road • Sudbury, MA 01776 USA
Phone: 978.261.5552 • Fax: 978.261.5553 • www.ebgconsulting.com

A StickyMinds.com Original
October, 2006

Events to the Rescue

By Mary Gorman

Summary: You probably know at least one project that has been mired in controversy and indecision during analysis. Team members arm wrestle about a variety of issues: Should we write use cases or user stories? How does user interface design fit with use cases? What is the "right" size for a use case or story? In many situations, an effective remedy is to define your project's events. In this column, Mary Gorman looks at an example from a recent engagement.

The Pain

Project Pluto was spinning. Some analysts had started and restarted writing use cases (a typical name was "Manage customer data"). Some were off sketching screen layouts. Others were building voluminous spreadsheets to document data details based on existing tables and conversations with the customers. Rumor was the technical team had finished drafting the system architecture, but no one had seen it.

The left hand didn't know what the right hand was doing. The project plan was a shell, everyone was confused, and the work had bogged down. Project Pluto was orbiting out of control, and the sponsor was alarmed.

Events Help Project Pluto Regain Its Balance

I was brought in to help the project. The team needed a strong foundation and a firm definition of its scope. Event modeling to the rescue!

First, the business and project folks collaborated in eliciting the events. We brainstormed "the things that happen that the business must respond to," identifying twenty-seven events to represent high-level behavioral requirements. We used a simple spreadsheet to capture event details: responses, direct and

  mary@ebgconsulting.com

indirect users, pre- and post-conditions, business priority, and anticipated interface requirements.

Then we held a review session with the stakeholders to validate the events and plug some gaps. Although a number of the attendees had not participated in the brainstorming sessions, they quickly grasped the event concept and provided helpful feedback.

Selecting the Modeling Techniques

In Project Pluto, the events served as a springboard for selecting the appropriate analysis techniques in the following ways:

- For events initiated by a human, we used use cases (e.g., "Customer orders a product"). In turn, the details of actor actions and system responses served as the basis for user interface analysis and design.
- For events initiated by time, a device, or another system, we used stories, and for those with complex logic, activity diagrams (e.g., "Time to notify of delinquent payment" and "Delivery provider notifies of delivered date.")
- For events that produced a report, whether initiated by time or by a user, we used report prototypes (e.g., "CFO requests Sales by Region report.")

The event construct provided a bonus: It was a natural antidote to the too-big or too-small use case or story. For example, the business folks did not think that the use case "Manage customer data" was initiated by a single event. Instead, they enumerated several business events that resulted in customer data being created or updated.

Multiplying the Benefits of Events

The team rebuilt the project plan using the events as the foundation. Initially, the project manager, the system architect, and the managers of the test and development teams didn't need to know whether the events were going to be detailed in use cases or in stories. To build their piece of the plan, all they needed was clarity on the scope of the system behavior, which was captured in the events.

The events were also important when the team planned the releases and iterations. And because we elicited all events--not just those initiated by a user--the planning scope was both accurate and comprehensive. Analysts used the events' dependencies (as defined in their pre- and post-conditions) when "chunking" the system into increments that would deliver value to the customer.

The team also set up a system to trace each event to its business requirements and track the events through the life of the project. The events gave everyone—

business as well as technical people—a common vocabulary that eliminated a lot of potential misunderstanding.

The Case for Events

Software is built to respond to events, so events serve well as the foundation for virtually any domain application. They work for projects of any size, whether complex or simple, risk driven or change driven. They are useful for planning new development, building enhancements, and evaluating off-the-shelf software.

Events helped rescue Project Pluto from a painful situation. But it's so much better to avoid the pain in the first place. To set the stage for success, start your project with event modeling.

Appreciation My thanks go to Susan Burk, Ellen Gottesdiener, and Karl Wiegers for their helpful review of this column.

Further Reading

Requirements by Collaboration: Workshops for Defining Needs, by Ellen Gottesdiener

The Software Requirements Memory Jogger: A Pocket Guide to Help Software and Business Teams Develop and Manage Requirements, by Ellen Gottesdiener

Essential Systems Analysis, by Steve McMenaminand John Palmer

More About Software Requirements: Thorny Issues and Practical Advice, by Karl Wiegers

About the Author

Mary Gorman, senior associate at EBG Consulting, assists project teams to explore, analyze, and build robust business and system requirements models. Mary serves on the International Institute of Business Analysis' Body of Knowledge (IIBA BOK) committee and is the leader of the Requirements Elicitation sub-committee. She can be reached at mary@ebgconsulting.com and www.ebgconsulting.com.