Editor: Neil Maiden ■ City University, London ■ n.a.m.maiden@city.ac.uk

In Search of the System Concept

Olly Gotel

Requirements documents are often tedious to construct, difficult to manage, and poor for mediating communication between stakeholders during a software development project. Using *system concepts* can simplify the way you determine, handle, and communicate requirements in a busy world. They could even help you design more innovative software systems.

Requirements at warp speed

In New York—a city where speed daters decide about potential partners in just three minutes and a moment's hesitation will lose you a taxi ride—what chance do requirements engineers have to secure user interest in their work? You want how many stakeholders in your workshop and for how long? Who has the patience to fill out your perfectly crafted questionnaire? Do you really think your imaginative use-case scenarios or snappy Extreme Programming stories can compete with the latest Broadway show? If you want to exchange information in this city, you must do it in a New York minute.

Over in Hollywood, working writers have around 10 minutes to make a pitch. You might not be making a multimillion-dollar Hollywood movie, but you still have to tell your customers and business sponsors what you're trying to do, why you're trying to do it, and how you plan to do it. You must find a succinct way to explain the project to the development team and users, engage their interest, and secure their participation. How do you get people on the same page and engage disinterested parties in the time it takes to eat two bites of a bagel?

Explaining system concepts

A system concept is an explicit, compelling vision of what a project is all about. One concise description synthesizes the problem's essence, requirements, projected solution, cost, and return on investment. A well-articulated concept will help you steer through difficult development activities because it provides a framework for systemic thinking and discussion (see the sidebar, "Ten Reasons to Use System Concepts").

A concept could look as crude as this:

The New York transit authority wants to introduce a more convenient payment system for its users. The idea is to give travelers a smart card that can store unlimited-monthly-travel tickets and prepaid single-use tickets to allow complete flexibility with just one card. The customer can add credit to the card in ticket machines, at ticket booths, on the Internet, or over the phone. Unlike the existing system, customers can get a refund for lost or stolen cards. The cards will use RFID technology, so no swipe is necessary. The cost of replacing all the current card readers and upgrading the card vending machines, in addition to developing the controlling computer system, will be covered by the savings of not having to send engineers to clean and maintain the current card readers.

You could therefore consider the system concept as a project's gist. It might sometimes help to provide an analogy or metaphor to capture the idea. Basically, the system concept can be

Ten Reasons to Use System Concepts

1. Want to run a requirements workshop but need a tool to explain it and focus discussions? Use a concept.
2. Can't get customers and users to sign off on requirements? They'll probably sign off on a concept.
3. Need to tell your mother what your system is all about? Use a concept.
4. Want to make sure new requirements are in scope? Check whether they support the concept.
5. Want to compare two systems' requirements? Compare the concepts, and you'll know whether doing more work is worth the effort.
6. Having difficulty getting started with requirements? Use a concept to explore the problem space and solution space simultaneously—you often need to synthesize solutions to analyze and frame problems.
7. Getting bogged down in requirements details? Revisit the concept to make sure you haven't gone off course.
8. Got changing requirements? Examine the concept to decide whether the changes will have a significant effect. It's much easier to keep the concept up-to-date than a bulky requirements set, and the concept is possibly more stable.
9. Need to scale back on your requirements? Check that each still directly supports the concept and prioritize from there.
10. Overwhelmed by complexity? Work at a higher level of abstraction with multiple concepts.

anything you want or need, even a poster or a movie trailer (to belabor the Hollywood theme), as long as it conveys and renders visible the salient aspects of what you're trying to achieve. However, don't underestimate how difficult it can be to discover a good concept.

A highly-packaged, stylish, and well-designed concept example is a company or product tag line. You might consider "All the News That's Fit to Print" a mere marketing slogan for the *New York Times*, but it says everything you'd hope to expect from a newspaper: even-handedness, correctness, and attention to detail. You couldn't grasp what this newspaper claims to be about any better from a comprehensive requirements document.

An elegant, powerful example from another domain would be Jørn Utzon's shell-inspired sketches (www.arcspace.com/studio/utzon/pages/11.htm) that later became the design for the Sydney Opera House.

In many disciplines, concepts are such familiar tools that whole processes surround their development. In urban planning, for example, a preconcept design *charrette* is an intensive effort in which interested parties come together to agree on goals and requirements and envision creative solutions to urban design problems. The resulting concepts tend to be highly visual, capture the project's core design information, and inform the master plan. The charrette is therefore a mechanism to explore the interaction between the problem, requirements, and possible solutions.

Go find that concept

I'm not suggesting anything new. The system concept is the starting point for engineers in many fields. Rather, I want to emphasize the value of concept studies and suggest that searching for the system concept must be integral to any software development project. Not only is there value in treating requirements work as part of the wider design process, but rapidly developing the system concept can be crucial to a commercial project's feasibility or bid phase.

One of my past projects required understanding a system proposed for a problem in one defense environment and evaluating whether it would be a viable solution for another. The system dealt with the managed dissemination of information in a dynamic, fast-moving, and time-critical domain. Would reverse-engineering all the system requirements have been feasible? The system was still being developed, and we didn't have the full details. Would this have uncovered the stakeholders' true needs in the second environment, anyway? Conducting a full requirements-gathering exercise in the second environment simply wasn't practical owing to the domain's nature. Furthermore, enumerating exhaustive requirements before synthesizing possible design concepts would have likely destroyed the second system's feasibility. You don't always have the luxury of pure design freedom when dealing with complex systems of systems.

Where you have time pressure and incomplete information, it can be more effective to discover and compare system concepts. At a high level of abstraction, this project was about mediated supply and demand. Some necessary roles must be present (that is, a supplier, demander, and intermediary) and some basic responsibilities must be fulfilled (for example, supply awareness and demand awareness). Each possible configuration of roles and responsibilities defines a specific supply-and-demand concept. These configurations lead to distinct system architectures and support different types of information exchange. The problem then becomes one of examining whether the particular concepts identified in the two environments align (or can be made to align) in any reasonable way. The more interesting question becomes whether the concepts underpinning the developing system align. Concept studies can give an economical approach in such situations and highlight the bigger-picture issues that might not have surfaced by digging into the involved and changing requirements details.

The system concept both binds the problem, requirements, and possible solutions and functions as the seed for their future concepts. If you don't have time for comprehensive requirements work, try to figure out the system concept. Creating something simple and tangible to help you understand and communicate what your project is all about is often the shortcut that you need. This should be a skill in any requirements engineer's repertoire.

Olly Gotel is an assistant professor of computer science at Pace University in New York City. Contact her at ogotel@pace.edu.