

Scenario Testing

Cem Kaner, J.D., Ph.D.

Tutorial at the
Conference of the
Association for Software Testing

This work is licensed under the Creative Commons Attribution License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/2.0/> or send a letter to Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

These notes are partially based on research that was supported by NSF Grants EIA-0113539 ITR/SY+PE: “Improving the Education of Software Testers” and CCLI-0717613 “Adaptation & Implementation of an Activity-Based Online or Hybrid Course in Software Testing.” Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Abstract

Scenarios are credible stories about something that could happen in the future. Scenario tests involve scenarios that are likely to motivate a stakeholder with influence to demand that the product be fixed if it doesn't pass the tests. To achieve this, good scenarios convey human issues (why would people be unhappy, and how unhappy, if the program fails this test?) as well as the technical matters of software design.

The focus of this tutorial is how to design effective suites of scenario tests. The tutorial will start with a lecture that lays out several lines of analysis for creating scenarios. Each line will lead you to a different set of tests. Some are more productive for a given product (lead to more interesting tests) than others. The lecture will last about an hour. Then we'll practice (in small groups) applying lines of analysis to different software products, tying what we find back into a couple of general-group presentations and a summary at the end of the day.

One view of scenarios

In Rational Unified Process, a *scenario* is an instantiation of a *use case*. (Take a use case, specify the values of the data, get a scenario.)

Use cases generally specify the goal of the actor in terms of a desired state and a sequence of actions that achieve or fail to achieve the goal.

The scenario can be described with a *sequence diagram*. A scenario traces one of the paths through the use case. (See Collard, 1999).

There is no discussion of the underlying motivation of the user or the emotional significance of failure or success of the scenario. These are abstracted out when writing a use case, to avoid unnecessary descriptive complexity. (see Cockburn, p. 18).

One view of scenarios

- *Instantiated use case* is *NOT* what I have in mind when I say *scenario test*.
- But this idea, and other simplistic notions, are what my students come up with as they wrestle with applying the much richer concept of *scenario* to their own work.
 - We can do better than this.
 - This tutorial is about doing better than this.
 - But in the course of the tutorial, we'll look at this again because it will help us see the challenges of the scenario technique.

So what is a Scenario? A Scenario is....

Scenarios are attempts to describe in some detail a hypothetical sequence of events that could lead plausibly to the situation envisaged. Some scenarios may explore and emphasize an element of a larger problem, such as a crisis or other event that could lead to war, the process of "escalation" of a small war or local violence into a larger war, the spread or contraction of a limited war, the fighting of a war, the termination of the war, or the subsequent peace. The focus of such a scenario can be military events and activities, the internal dynamics of various countries, bargaining among enemies or inter-ally relations, and so on. Other scenarios can be used to produce, perhaps in impressionistic tones, the future development of the world as a whole, a culture, a nation, or some group or class. The scenario is particularly suited to dealing with events taken together—integrating several aspects of a situation more or less simultaneously. By the use of a relatively extensive scenario, the analyst may be able to get a feeling for events and the branching points dependent upon critical choices. These branches can then be explored more or less systematically or the scenario itself can be used as a context for discussion.

- (1) They serve to call attention, sometimes dramatically and persuasively, to the larger range of possibilities that must be considered in the analysis of the future. They are one of the most effective tools in lessening the "carry-over" thinking that is likely even when it is clear to all that 2000 cannot be the same as 1965 or even 1985. Scenarios are one way to force oneself and others to plunge into the unfamiliar and rapidly changing world of the present and the future:
- (2) They dramatize and illustrate the possibilities they focus on in a very useful way. (They may do little or nothing for the possibilities they do not focus on.)
- (3) They force the analyst to deal with details and dynamics that he might easily avoid treating if he restricted himself to abstract considerations. Typically no particular set of the many possible sets of details and dynamics seems especially worth treating, so none are treated, even though a detailed investigation of even a few arbitrarily chosen cases can be most helpful
- (4) They help to illuminate the interaction of psychological, social, economic, cultural, political, and military factors, including the influence of individual political personalities upon what otherwise might be abstract considerations, and they do so in a form that permits the comprehension of many such interacting elements at once.
- (5) They can illustrate forcefully, sometimes in oversimplified fashion, certain principles, issues, or questions that might be ignored or lost if one insisted on taking examples only from the complex and controversial real world.
- (6) They may also be used to consider alternative possible outcomes of certain real past and present events, such as Suez, Lebanon, Laos, or Berlin.
- (7) They can be used as artificial "case histories" and "historical anecdotes" to make up to some degree for the paucity of actual examples.

A Scenario is....

Scenario methodology for teaching principles of emergency management

Scenarios are discussed in terms of their various uses in emergency planning and management. Their function in teaching programs is assessed with respect to various sorts of curricula for training emergency personnel. The format of scenarios is discussed and the potential for using the methodology creatively is explored. It is concluded that scenario methods are useful in developing such skills as time management, cognitive mapping, mediation, team management, and decision making under stress. A brief example of an emergency training scenario is presented and evaluated in terms of its teaching potential. Finally, scenarios are assessed in relation to other forms of modeling and simulation, such as table-top games and field exercises, which are commonly used for training emergency managers.

Alexander (2000), p. 89.

A Scenario is ...

“The New Basel Capital Accord (Basel II) valid since January 2007 for international active banks newly introduced a capital requirement for operational risk (in addition to credit and market risk)... In this paper we focus on modeling and stress testing of economic and regulatory capital set aside to cover unexpected losses of an anonymous Central European bank...” (p. 2)

One of those methods is the scenario analysis or, generally, the stress testing. This method is supposed to examine whether a financial institution would be able to undergo exceptional risk losses. Stress testing can be defined as “the examination of the potential effects on a bank’s financial condition of a set of specified changes in risk factors, corresponding to exceptional but plausible events. (p. 16)

“The scenarios can be divided into two groups based on the type of event they define. The first group uses historical events like 9/11 terrorist attacks or unauthorized trading that happened in Societe Generale in 2007. Risk managers study a potential impact of those events on the financial institution. The second group, which is more widely used in practice, uses hypothetical scenarios. The scenarios are based on some plausible risk events that have not happened yet, but a non-zero probability of their occurrence exists. A scenario can also be based on an analysis of a new product a bank is going to implement.

“A typical scenario consists of the description of a complex state of the world that would impose an extreme risk event on a financial institution, including probabilities and frequencies of occurrence of the particular state of the world, business activities impacted by the event and maximum internal and external loss amounts generated by occurrence of such event, possible mitigation techniques including insurance against such an extreme event. Even though such a scenario claims to be realistic, it is not possible to comprise all possible risk factors and features. However, risk managers are trying to define the scenarios, so that they correspond to the reality as much as possible. It is clear that “the generation of relevant scenarios is a time-consuming process that requires quantitative skills as well as good economic understanding of the factors” financial institution are exposed to. “ (p. 16-17)

A Scenario is..

“The use of the term *scenario* as an analytical tool dates from the early 1960s, when researchers at the RAND Corporation defined states of the world within which alternative weapons systems or military strategies would have to perform. Since then their use has grown rapidly, and the meanings and uses of scenarios have become increasingly varied.... For the purposes of this paper, I propose the following definition of a crisis management scenario...

“A description of the conditions under which the crisis management system or crisis management policy to be designed, tested, or evaluated is assumed to perform...” P. 1-2

Walker (1994)

A Scenario is...

Computers are more than just functionality. They unavoidably restructure human activities, creating new possibilities as well as new difficulties. Conversely, each context in which humans experience and act provides detailed constraint for the development and application of computer technologies. In analyzing and designing systems and software we need better means to talk about how they may transform and/or be constrained by the contexts of user activity: this is the only way we can hope to attain control over the “materials” of design. A direct approach is to explicitly envision and document typical and significant user activities early and continually in the development process. Such descriptions, often called “scenarios”, support reasoning about situations of use, even before those situations are actually created.

Scenarios are stories. They are stories about people and their activities. For example, an accountant wishes to open a folder on the system desktop in order to access a memo on budgets. However, the folder is covered up by a budget spreadsheet that the accountant wishes to refer to while reading the memo. The spreadsheet is so large that it nearly fills the display. The accountant pauses for several seconds, resizes the spreadsheet, moves it partially out of the display, opens the folder, opens the memo, resizes and repositions the memo, and continues working.

This is about as mundane a work scenario as one could imagine. Yet even this scenario specifies window management and application switching functionality vividly and pointedly: People need to coordinate information sources, to compare, copy, and integrate data from multiple applications; displays inevitably get cluttered; people need to find and rearrange windows in the display. Scenarios highlight goals suggested by the appearance and behavior of the system, what people try to do with the system, what procedures are adopted, not adopted, carried out successfully or erroneously, and what interpretations people make of what happens to them. If the accountant scenario is typical of what people want to do, it substantively constrains design approaches to window management and switching.

Carroll (1999, p 1-2)

A Scenario is...

Scenarios have characteristic elements. They include or presuppose a *setting*: *The accountant scenario* explicitly describes a starting state for the described episode; the relative positions of the folder and spreadsheet, and the presence of the accountant. The scenario implies further setting elements by identifying the person as an accountant, and the work objects as budgets and memos.

Scenarios also include *agents or actors*: *The accountant* is the only agent in this example, but it is typical of human activities to include several to many agents. Each agent or actor typically has *goals or objectives*. *These are* changes that the agent wishes to achieve in the circumstances of the setting. Every scenario involves at least one agent and at least one goal. When more than one agent or goal is involved, they may be differentially prominent in the scenario. Often one goal is the defining goal of a scenario, the answer to the question “why did this story happen?” Similarly, one agent might be the principal actor, the answer to the question “who is this story about?”

In the accountant scenario, the defining goal is displaying the memo in such a way that both the memo and budget can be examined. A subgoal is opening the folder in which the memo is located, and a further subgoal is moving the budget to allow the folder to be opened. Scenarios have a plot; they include sequences of *actions* and *events, things that actors do, things that happen to them*, changes in the circumstances of the setting, and so forth. Particular actions and events can facilitate, obstruct, or be irrelevant to given goals. Resizing the spreadsheet and moving it out of the display are actions that facilitate the goal of opening the folder. Resizing and repositioning the memo are actions that facilitate the goal of displaying the memo so that it can be examined with the budget. Pausing is an action that is irrelevant to any goal, though it suggests that the accountants goal-oriented actions were not completely fluent. Notably, actions and events can often *change the goals --even the defining goal --* of a scenario.

Representing the use of a system or application with a set of user interaction scenarios makes that *use explicit*, and in doing so orients design and analysis toward a broader view of computers. It can help designers and analysts to focus attention on the assumptions about people and their tasks that are implicit in systems and applications. Scenario representations can be elaborated as prototypes, through the use of storyboard, video, and rapid prototyping tools. They are the minimal contexts for developing user-oriented design rationale: a given design decision can be evaluated and documented in terms of its specific consequences within particular scenarios. Scenarios and the elements of scenario-based design rationale can be generalized and abstracted using theories of human activity, enabling the cumulation and development of knowledge attained in the course of design. Thus, scenarios can provide a framework for a design-based science of human-computer interaction.

Carroll (1999, p 1-2) continued

A Scenario is...

A scenario is a coherent story about how someone uses (or tries to use) the program.

The elements (Carroll, 1999)

- Setting
- Agents or actors
- Goals or objectives
- Plot (sequences of actions and events)
- Actions and events can change the goals

A *scenario test* checks whether the program behaves appropriately as the tester attempts the scenario.

Attributes of scenario tests

The ideal scenario has several characteristics:

The test is *based on a coherent story* about how the program is used, including information about emotions of the people involved.

The *story is credible*. It not only could happen in the real world; stakeholders would believe that something like it probably will happen.

The *story is motivating*. A stakeholder with influence would advocate for fixing a program that failed this test.

The story involves a *complex* use of the program or a complex environment or a complex set of data.

The test results are *easy to evaluate*. This is valuable for all tests, but is especially important for scenarios because they are complex.

Benefits of scenarios

Learn the product.

- Work through control flow of the tasks (track through use cases).
- Explore the diversity of the product's potential uses and relations with other products or systems.

Connect testing to documented requirements.

- Verify ability to achieve specific tasks (including basic error handling) (use cases)
- Verify expected benefits (“requirements”) rather than implementation-oriented specifications

Benefits of scenarios

Expose failures to deliver desired benefits.

- Bring requirements-related issues to the surface, which might involve reopening old requirements discussions (with new data) or surfacing not-yet-identified requirements.
- Good scenario tests go beyond the simple uses of the program to ask whether the program is delivering the benefits it should deliver.

Scenarios & Requirements Analysis

Designing scenario tests is much like doing a requirements analysis, but it is not requirements analysis.

They rely on similar information but use it differently.

- The requirements analyst tries to foster agreement about the system to be built. The tester exploits disagreements to predict problems with the system.
- The tester doesn't have to reach conclusions or make recommendations about how the product should work. Her task is to expose credible concerns to the stakeholders.
- The tester doesn't have to make the product design tradeoffs. She exposes the consequences of those tradeoffs, especially unanticipated or more serious consequences than expected.
- The tester doesn't have to respect prior agreements. (Caution: testers who belabor the wrong issues lose credibility.)
- The scenario tester's work need not be exhaustive, just useful.

Benefits of scenarios

Scenario testing provides an approach to combination testing that is focused on the value of the product. (Combinations can involve sequences as well as data.)

- Mechanical
 - Such as random or combinatorial
- Risk-based
 - Combinations that are perceived as more likely to yield failure or yield consequences that are more serious if failure occurs
- Scenario-based
 - Combinations that can provide insight into the value of the product

Benefits of scenarios

Explore *expert* use of the product.

- Many test techniques tell us how the program will behave in the first few days that someone uses it. Good scenarios often give us insight into frustrations that an experienced user will face—someone who has used the program for a few months and is now trying to do significant work with the program.

Improve the motivational quality of a bug report.

- Scenarios are powerful tools for building a case that a deferred bug should be fixed. Inability to develop a strong scenario around a failure may be a signal that the failure is not well understood or not important.

Stories about scenario-based analysis

- Hans Buwalda, banks and soap operas
- Using failure scenarios to strengthen the argument for fixing a bug

16 lines of inquiry for suites of scenarios

1. List possible users. Analyze their interests and objectives.
2. Work alongside users (or interview them) to see how they work and what they do.
3. Interview users about famous challenges and failures of the old system.
4. Look at the specific transactions that people try to complete, such as opening a bank account or sending a message. What are all the steps, data items, outputs, displays, etc.?
5. Look for sequences: People (or the system) typically do task X in an order. What are the most common orders (sequences) of subtasks in achieving X?
6. Consider disfavored users. Analyze their interests, objectives, capabilities, and potential opportunities.

16 lines of inquiry

7. What forms do the users work with? Work with them (read, write, modify, etc.)
8. Write life histories for objects in the system. How was the object created, what happens to it, how is it used or modified, what does it interact with, when is it destroyed or discarded?
9. List system events. How does the system handle them?
10. List special events. What accommodations does the system make for these?

16 lines of inquiry

11. List benefits and create end-to-end tasks to check them.
12. Read about what systems like this are supposed to do. Play with competing systems.
13. Study complaints about the predecessor to this system or its competitors.
14. Create a mock business. Treat it as real and process its data.
15. Try converting real-life data from a competing or predecessor application.
16. Look at the output that competing applications can create. How would you create these reports / objects / whatever in your application?

We're creating suites, not single tests

- We are taking an inventory of some aspect of the product (much like the ideas of touring that Bach & Kelly were describing in the same period)
- Given an item in the list, we ask scenario-building questions:
 - What is important about this?
 - Why do people care about it?
 - Who would care about it?
 - What does it relate to that modifies its importance?
 - What gets impacted if it fails?
 - What does failure look like?
 - What are the consequences of failure?
 - Does it ever take on urgency?

- Coherent story
- Credible
- Motivating
- Complex
- Easy to evaluate

We're creating suites, not single tests

- Given an item in the list, we ask the scenario questions:
 - What would make a story about this be credible?
 - When would this come up, or be used?
 - Who would use it?
 - What would they be trying to achieve?
 - Competitor examples?
 - Spec / support / history examples?

- Coherent story
- Credible
- Motivating
- Complex
- Easy to evaluate

We're creating suites, not single tests

- Given an item in the list, we ask the scenario questions:
 - How do we increase complexity?
 - What does this naturally combine with?
 - What benefits involve this and what collection of things would be required to achieve each?
 - Can we make it bigger? Do it more? Work with richer data? (What boundaries are involved?)

- Coherent story
- Credible
- Motivating
- Complex
- Easy to evaluate

We're creating suites, not single tests

- Given an item in the list, we ask the scenario questions:
 - How do we design an evaluatable test?
 - Self-verifying data sets?
 - Automatable partial oracles?
 - Known, predicted result?

- Coherent story
- Credible
- Motivating
- Complex
- Easy to evaluate

We're creating suites, not single tests

- Given an item in the list, we ask the scenario questions:
 - How do we create a story that people will listen to?
 - Setting
 - Agents or actors
 - Goals or objectives
 - Plot (sequences of actions and events)
 - Actions and events can change the goals
 - Emotions

- Coherent story
- Credible
- Motivating
- Complex
- Easy to evaluate

Some suggestions

- Test each feature in isolation before testing scenarios, to efficiently expose problems as soon as they appear.
- Don't run prematurely complex scenarios. If the first feature in your scenario is broken, the rest of the test can't be run. Once that feature is fixed, the next broken feature blocks the test.
- Wiser to follow one line than to combine them.
- Some lines will be more productive than others for some products / contexts.
- You can't guarantee high code coverage from this (but you might guarantee high requirements coverage or high feature coverage)
- Pay careful attention to evaluatability. Many failures have been exposed by a good scenario but missed by the tester

Reusing scenarios may lack power and be inefficient

- Documenting and reusing scenarios seems efficient because it takes work to create a good scenario.
- Scenarios often expose design errors but we soon learn what a test teaches about the design.
- Scenarios expose coding errors because they combine many features and much data. To cover more combinations, we need new tests.
- It might be more effective to do regression testing with single-feature tests or unit tests, not scenarios.
- Unless you can substantially automate creation / assessment, you are better off using these for exploration than regression
- You can't easily automate this in the traditional GUI way (but keyword frameworks have had success – Buwalda / Logigear)

Applications

- Mind Manager
- Google docs
- Open Office
- What else would you like?

Some References

Anggreeni, I. and van der Voort, M. (2007) Tracing the Scenarios in Scenario-Based Product Design: a study to support scenario generation. Technical Report TR-CTIT-07-70, Centre for Telematics and Information Technology, University of Twente, Enschede. ISSN 1381-3625. <http://eprints.eemcs.utwente.nl/11231/01/TR-CTIT-07-70.pdf>

Alexander, David (2000). Scenario methodology for teaching principles of emergency management. *Disaster Prevention & Management*, Vol. 9(2), pp.89 - 97

Alexander, Ian & Maiden, Neil. *Scenarios, Stories, Use Cases: Through the Systems Development Life-Cycle*.

Berger, Bernie (2001) "The dangers of use cases employed as test cases," STAR West conference, San Jose, CA. www.testassured.com/docs/Dangers.htm.

Bolton, Michael (2007). *Users We Don't Like*. Developsense.com. <http://www.developsense.com/articles/2007-08-UsersWeDontLike.pdf>.

Buwalda, Hans (2000a) "The three holy grails of test development," presented at EuroSTAR conference.

Buwalda, Hans (2000b) "Soap Opera Testing," presented at International Software Quality Week Europe conference, Brussels. <http://www.logigear.com/resource-center/software-testing-articles-by-logigear-staff/246-soap-opera-testing.html>.

Some References

- Carroll, John M. (1995). *Development, Scenario-Based Design: Envisioning Work and Technology in System*. Wiley.
- Carroll, John M. (1999). Five reasons for scenario-based design. Proceedings of the 32nd Hawaii International Conference on System Sciences, <http://www.massey.ac.nz/~hryu/157.757/Scenario.pdf>
- Cockburn, Alexander (2000). *Writing Effective Use Cases*. Addison-Wesley.
- Cohn, Mike (2004). *User Stories Applied: For Agile Software Development*. Pearson Education.
- Collard, R. (1999, July) “Developing test cases from use cases”, *Software Testing & Quality Engineering*, available at www.stickyminds.com.
- Heijden, Kees van der (1996). *Scenarios: The Art of Strategic Conversation*. Wiley.
- Jacobson, Ivar & Bylund, Stefan. *The road to the unified software development process*
- Kaner, Cem (2003). *An introduction to scenario testing*. <http://www.kaner.com/pdfs/ScenarioIntroVer4.pdf>.
- Kahn, Herman (1967). The use of scenarios. In Kahn, Herman & Wiener, Anthony (1967), *The Year 2000: A Framework for Speculation on the Next Thirty-Three Years*, pp. 262-264. http://www.hudson.org/index.cfm?fuseaction=publication_details&id=2214

Some References

Ringland, Gill (1998). *Scenario Planning: Managing for the Future*. Wiley.

Rippel, Milan & Teply, Petr (2009). *Operational Risk -- Scenario Analysis*. Working Papers IES 2008/15, Charles University Prague, Faculty of Social Sciences, Institute of Economic Studies, revised Sep 2008.

http://ideas.repec.org/p/fau/wpaper/wp2008_15.html

Robertson, Suzanne & Robertson, James C. (2006). *Mastering the Requirements Process*. Addison-Wesley Professional.

Rosson, Mary Beth & Carroll, John M (2002). *Usability Engineering*: Morgan Kaufmann.

Walker, Warren E. (1994). The use of scenarios and gaming in crisis management planning and training. Presented at the conference, *The Use of Scenarios for Crisis Management*, Netherlands Ministry of Home Affairs, at the Netherlands Institute for Fire Service & Disaster Mgmt, Arnhem, November 16-18.

Weber, Bob. (2008) *Information Commerce 1997 – Scenario Mapping Changes Beliefs*. http://www.strategykinetics.com/scenario_planning/