SYS 5460: Use Cases, Scenarios , and Functional Requirements

Contents:

- Use Cases
- Scenarios
- Modeling Recommendations
- Class Mini Project

Department of Computer & Engineering Sciences

College of Engineering

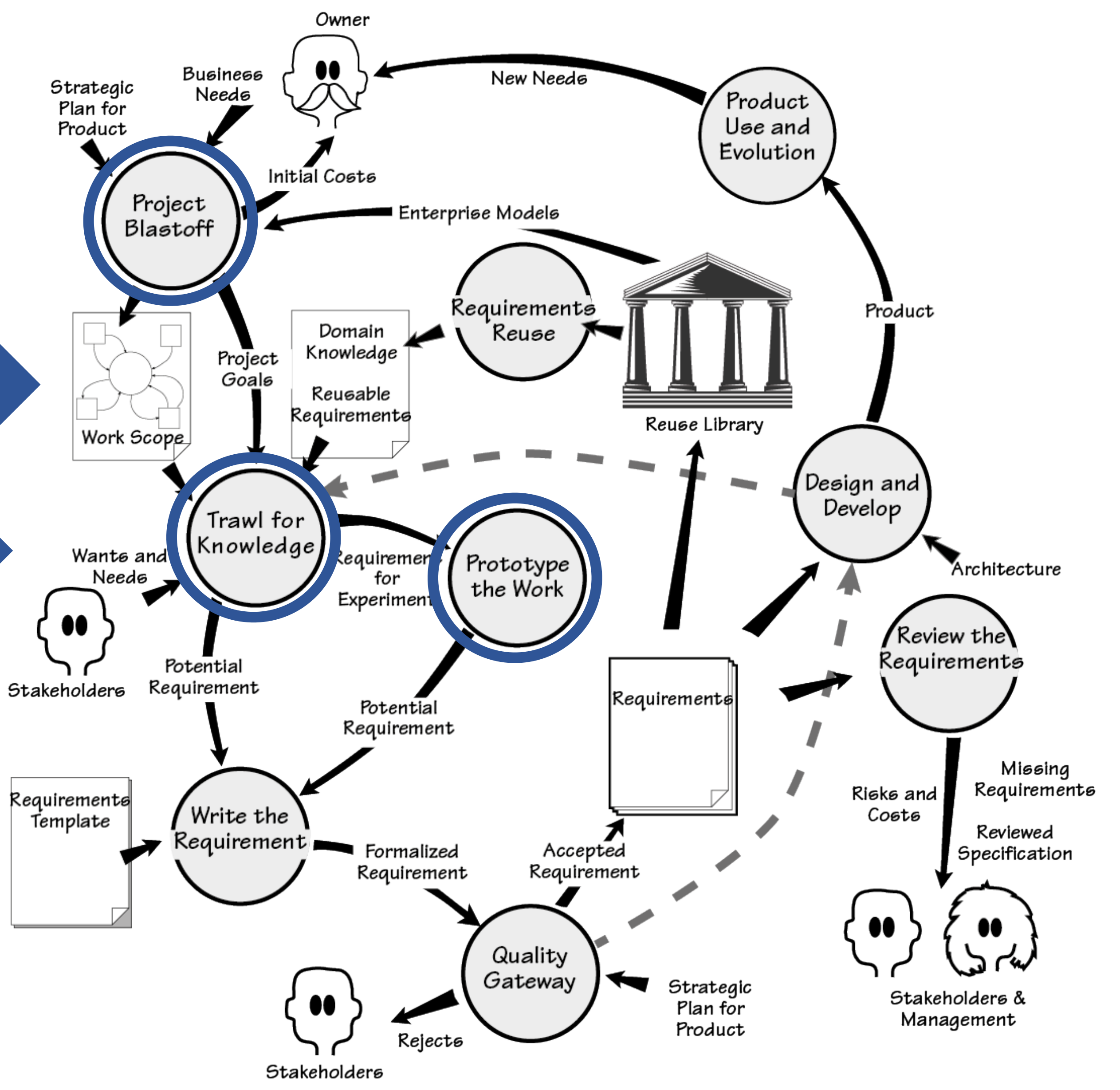
Florida Institute of Technology

FLORIDA TECH

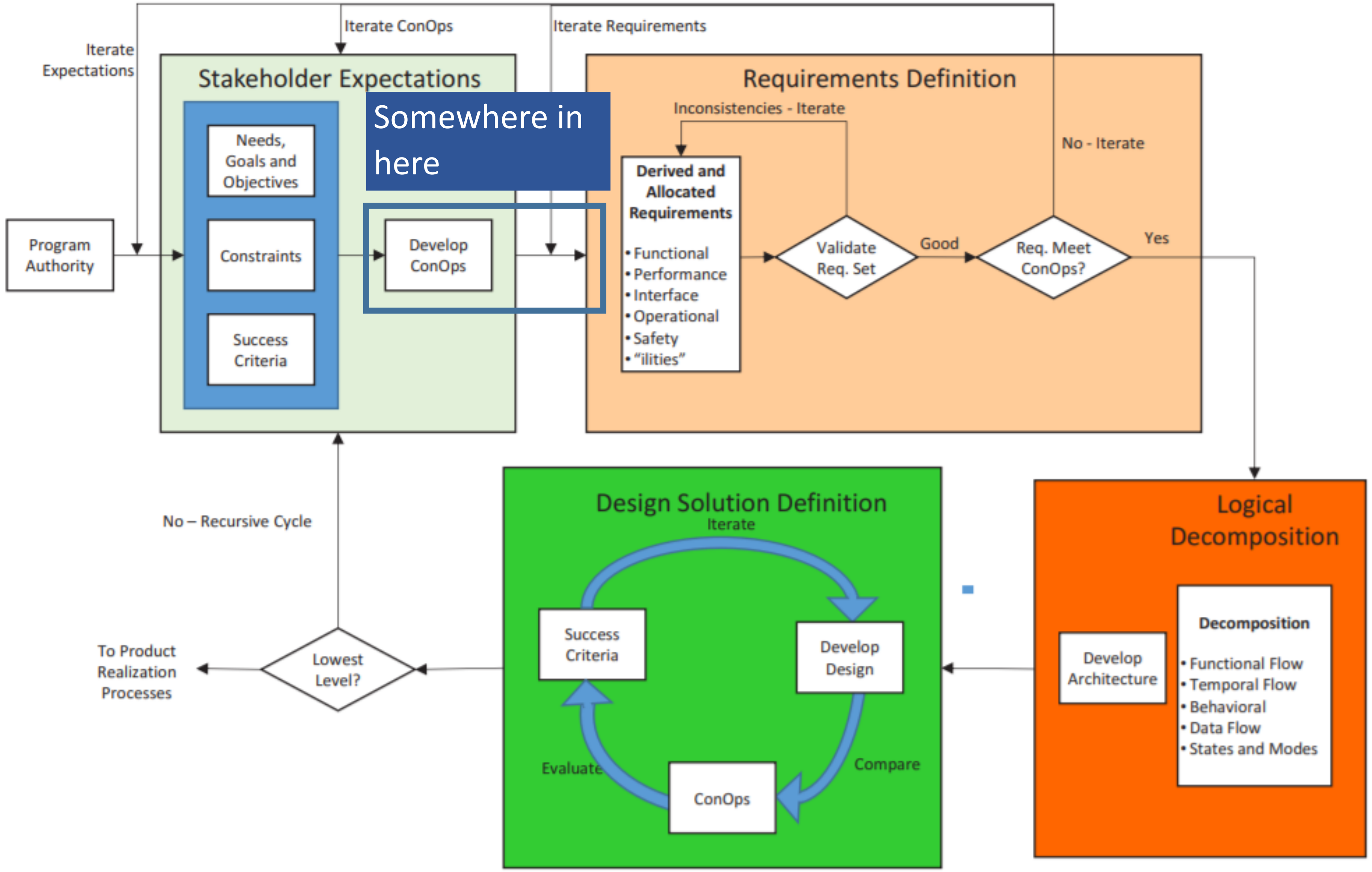
Volere Requirements Process (Robertson)

Model system functionality

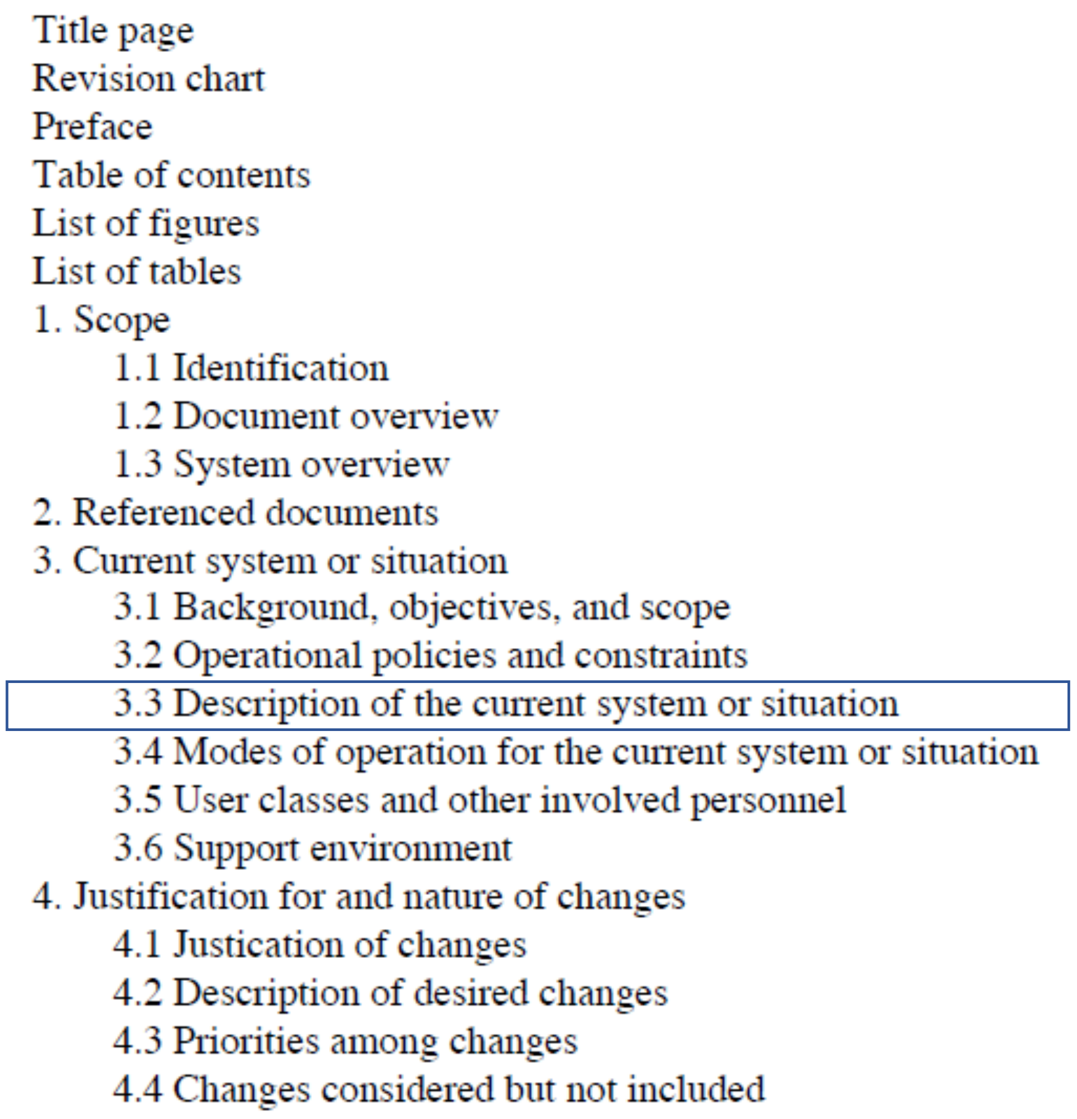
Concept of Operations



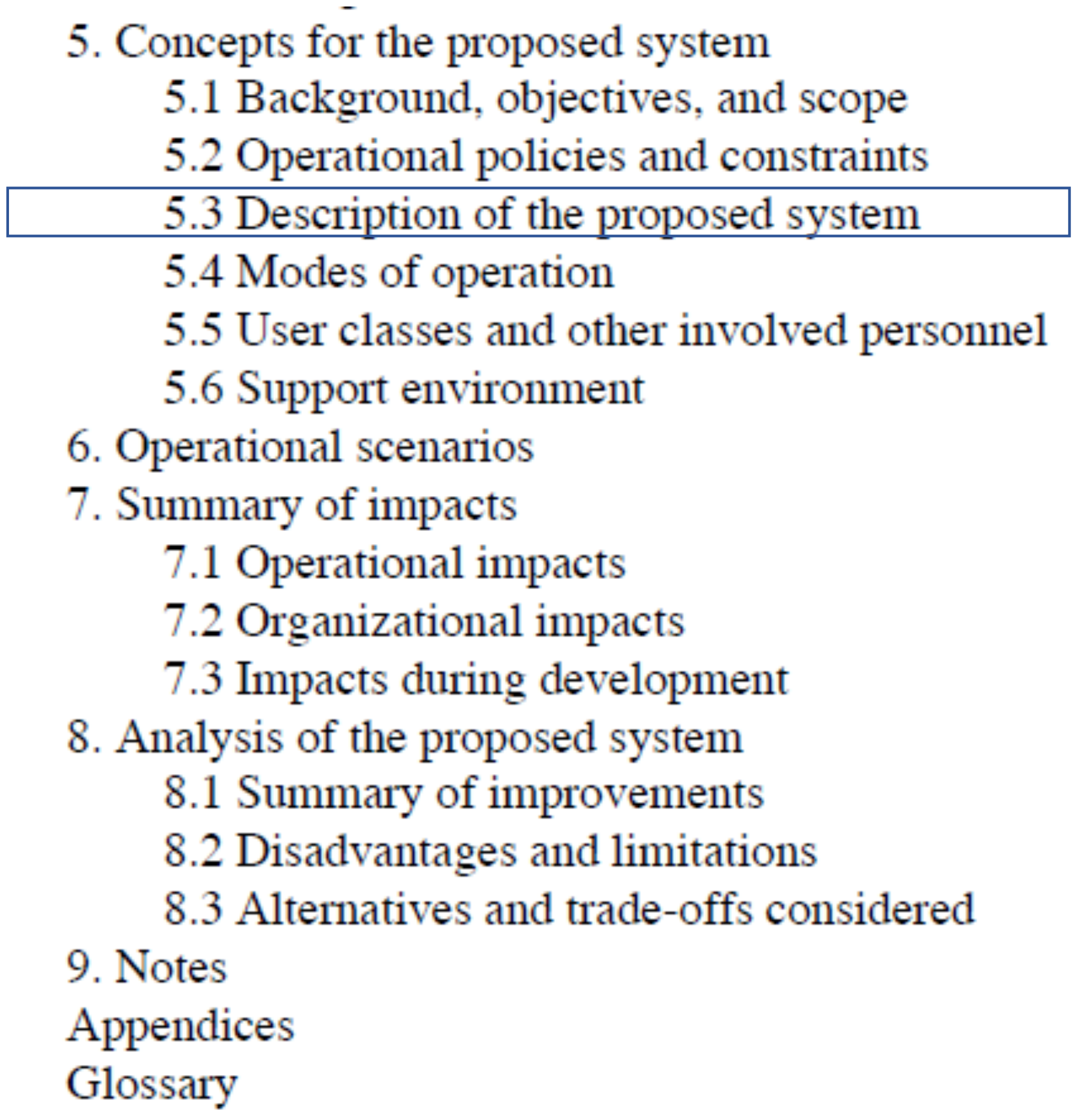
Systems Engineering Process NASA Rev 2



Concept of Operations Document (IEEE 1362-1998)



Title page
Revision chart
Preface
Table of contents
List of figures
List of tables
1. Scope
    1.1 Identification
    1.2 Document overview
    1.3 System overview
2. Referenced documents
3. Current system or situation
    3.1 Background, objectives, and scope
    3.2 Operational policies and constraints
    3.3 Description of the current system or situation
    3.4 Modes of operation for the current system or situation
    3.5 User classes and other involved personnel
    3.6 Support environment
4. Justification for and nature of changes
    4.1 Justication of changes
    4.2 Description of desired changes
    4.3 Priorities among changes
    4.4 Changes considered but not included



5. Concepts for the proposed system
    5.1 Background, objectives, and scope
    5.2 Operational policies and constraints
    5.3 Description of the proposed system
    5.4 Modes of operation
    5.5 User classes and other involved personnel
    5.6 Support environment
6. Operational scenarios
7. Summary of impacts
    7.1 Operational impacts
    7.2 Organizational impacts
    7.3 Impacts during development
8. Analysis of the proposed system
    8.1 Summary of improvements
    8.2 Disadvantages and limitations
    8.3 Alternatives and trade-offs considered
9. Notes
Appendices
Glossary

Scenarios

Scenario

A **scenario** is a **scene** that illustrates some interaction with a proposed system.

A **scenario** is a tool used during requirements analysis to describe a specific use of a proposed system. Scenarios capture the system, as viewed from the outside, e.g., by a user, using specific examples.

Note on terminology

Some authors restrict the word "scenario" to refer to a user's total interaction with the system.

Other authors use the word "scenario" to refer to parts of the interaction.

In this course, the term is used with both meanings.

Describing a Scenario

Some organizations have complex documentation standards for describing a scenario.

At the very least, the description should include:

- A statement of the purpose of the scenario
- The individual user or transaction that is being followed through the scenario
- Assumptions about equipment or so(ware
- The steps of the scenario

Developing a Scenario with a Client

Example of how to develop a scenario with a client

The requirements are being developed for a system that will enable university students to take exams online from their own rooms using a web browser.

Create a scenario for how a typical student interacts with the system.

In the next few slides, the questions in blue are typical of the questions to ask the client while developing the scenario.

Developing a Scenario with a Client: a Typical Student

Purpose: Scenario that describes the use of an online Exam system by a representative student

Individual: *[Who is a typical student?]* Student A, senior at FIT, major in computer science. *[Where can the student be located? Do other universities differ?]*

Equipment: Any computer with a supported browser. *[Is there a list of supported browsers? Are there any network restrictions?]*

Scenario:

1. Student A authenticates. *[How does a Florida Tech student authenticate?]*
2. Student A starts browser and types URL of Exam system. *[How does the student know the URL?]*
3. Exam system displays list of options. *[Is the list tailored to the individual user?]*

4. Student A selects CS 1234 Exam 1.

5. A list of questions is displayed, each marked to indicate whether completed or not. *[Can the questions be answered in any order?]*

6. Student A selects a question and chooses whether to submit a new answer or edit a previous answer. *[Is it always possible to edit a previous answer? Are there other options?]*

7. *[What types of question are there: text, multiple choice, etc.?]* The first question requires a wri]en answer. Student A is submi^ng a new answer. The student has a choice whether to type the solution into the browser or to a]ach a separate file. Student A decides to a]ach a file. *[What types of file are accepted?]*

Developing a Scenario with a Client (continued)

8. For the second question, the student chooses to edit a previous answer. Student A chooses to delete a solution previously typed into the browser, and to replace it with an a]ached file. *[Can the student edit a previous answer, or must it always be replaced with a new answer?]*

9. As an alternative to completing the entire exam in a single session, Student A decides to saves the completed questions work to continue later. *[Is this always permiMed?]*

10. Student A logs off.

11. Later Student A log in, finishes the exam, submits the answers, and logs out. *[Is this process any different from the initial work on this exam?]*

12. The Student A has now completed the exam. The student selects an option that submits the exam to the grading system. *[What if the student has not aMempted every question? Is the grader notified?]*

Developing a Scenario with a Client (continued)

13. Student A now wishes to change a solution. The system does not permit changes once the solution has been submi]ed. *[Can the student still see the solutions?]*

14. Later Student A logins in to check the grades*. [When are grades made available? How does the student know?]*

15. Student A requests a regrade. *[What are the policies? What are the procedures?]*

Developing a Scenario with a Client (continued)

- Developing a scenario with a client clarifies many functional requirements that must be agreed before a system can be built, e.g., policies, procedures, etc.
- The scenario will (clarify the requirements for the user interface, but the design of the user interface should not be part of the scenario.

Although this scenario is quite simple, many details have been laid out.

Scenarios for Analyzing Special Requirements

Scenarios are very useful for analyzing special requirements.

Examples

- Reversals. In a financial system, a transaction is credited to the wrong account. What sequence of steps are used to reverse the transaction?
- Errors. A mail order company has several copies of its inventory database. What happens if they become inconsistent?
- Malfeasance. In a voting system, a voter has houses in two cities. What happens if he a]empts to vote in both of them?

Scenarios for error recovery

Murphy's Law: "*If anything can go wrong, it will*". Create a scenario for everything that can go wrong and how the system is expected to handle it.

Modeling Scenarios as Use Cases

Models

Scenarios are useful in discussing a proposed system with a client, but requirements need to be made more precise before a system is fully understood.
This is the purpose of requirements **modeling**.

A **use case** provides such a model.

Describing a Use Case

Some organizations have complex documentation standards for describing a use case.

At the very least, the description should include:

- The name of the use case, which should summarize its purpose
- The actor or actors
- The flow of events
- Assumptions about entry conditions

Outline of Take Exam Use Case

Name of Use Case: Take Exam

Actor(s): ExamTaker

Flow of events:

1. ExamTaker connects to the Exam server.
2. Exam server checks whether ExamTaker is already authenticated and runs authentication process if necessary.
3. ExamTaker selects a exam from a list of options.
4. ExamTaker repeatedly selects a question and either types in a solution, a]aches a file with a solution, edits a solution or a]aches a replacement file.

Outline Specification of Use Case (continued)

Flow of events (continued):

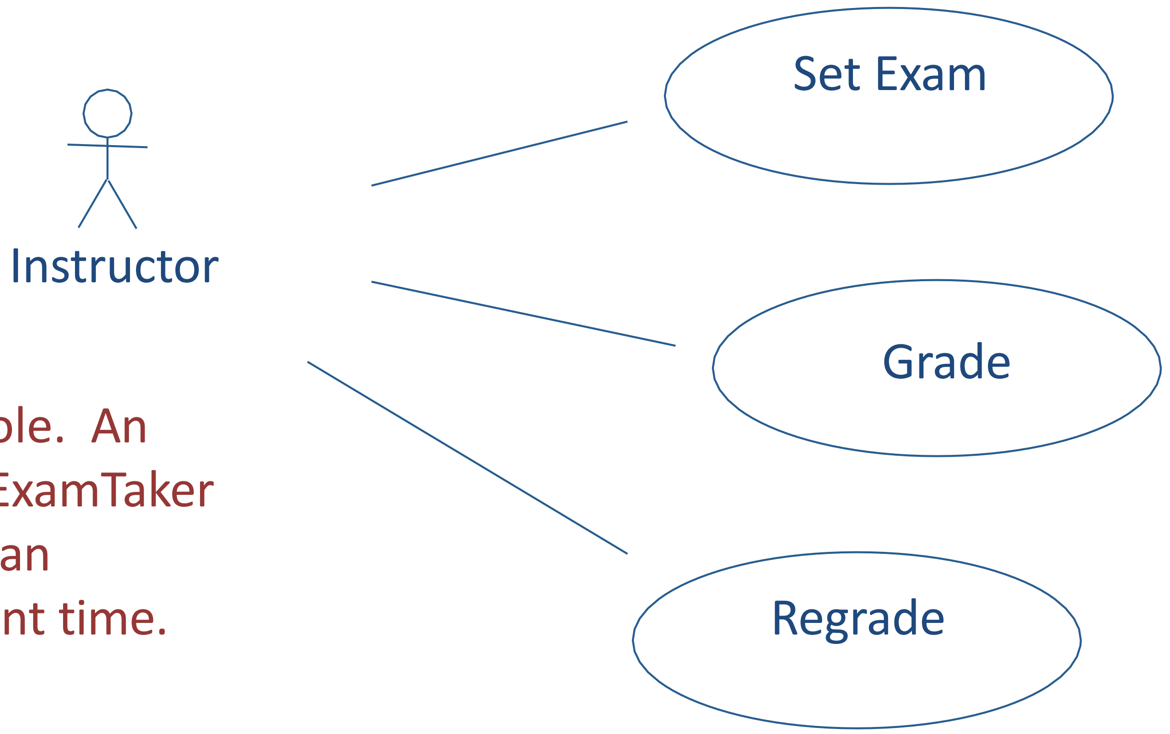
5. ExamTaker either submits completed exam or saves current state.
6. When a completed exam is submi]ed, Exam server checks that all questions have been a]empted and either sends acknowledgement to ExamTaker, or saves current state and notifies ExamTaker of incomplete submission.

7.ExamTaker logs out.

Entry conditions:

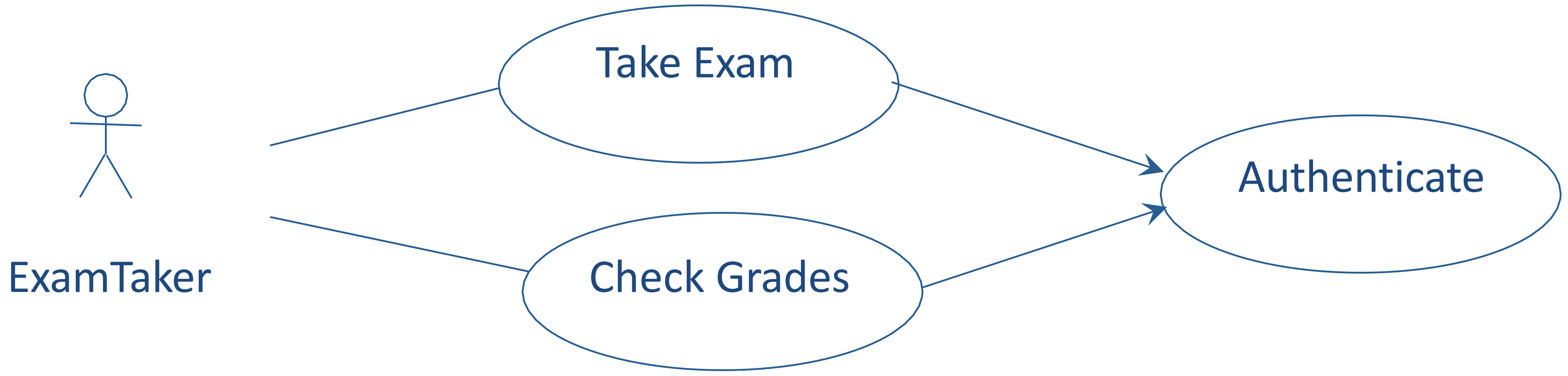
1. ExamTaker must have authentication credentials.
2. Computing requirements: supported browser.

Use Cases for Exam System (continued)



Note that actor is a role. An individual can be an ExamTaker on one occasion and an Instructor at a different time.

Relationships Between Use Cases:



The Authenticate use case may be used in other contexts

Scenarios and Use Cases in the Development Cycle

- Scenarios and use cases are both intuitive –easy to discuss with clients
- Scenarios are a tool for requirements analysis.
  - They are useful to validate use cases and in checking the design of a system.
  - They can be used as test cases for acceptance testing.
- Use cases are a tool for modeling requirements.
  - A set of use cases can provide a framework for the requirements specification.
  - Use cases are the basis for system and program design, but are o(en hard to translate into class models

Why do we do scenarios?

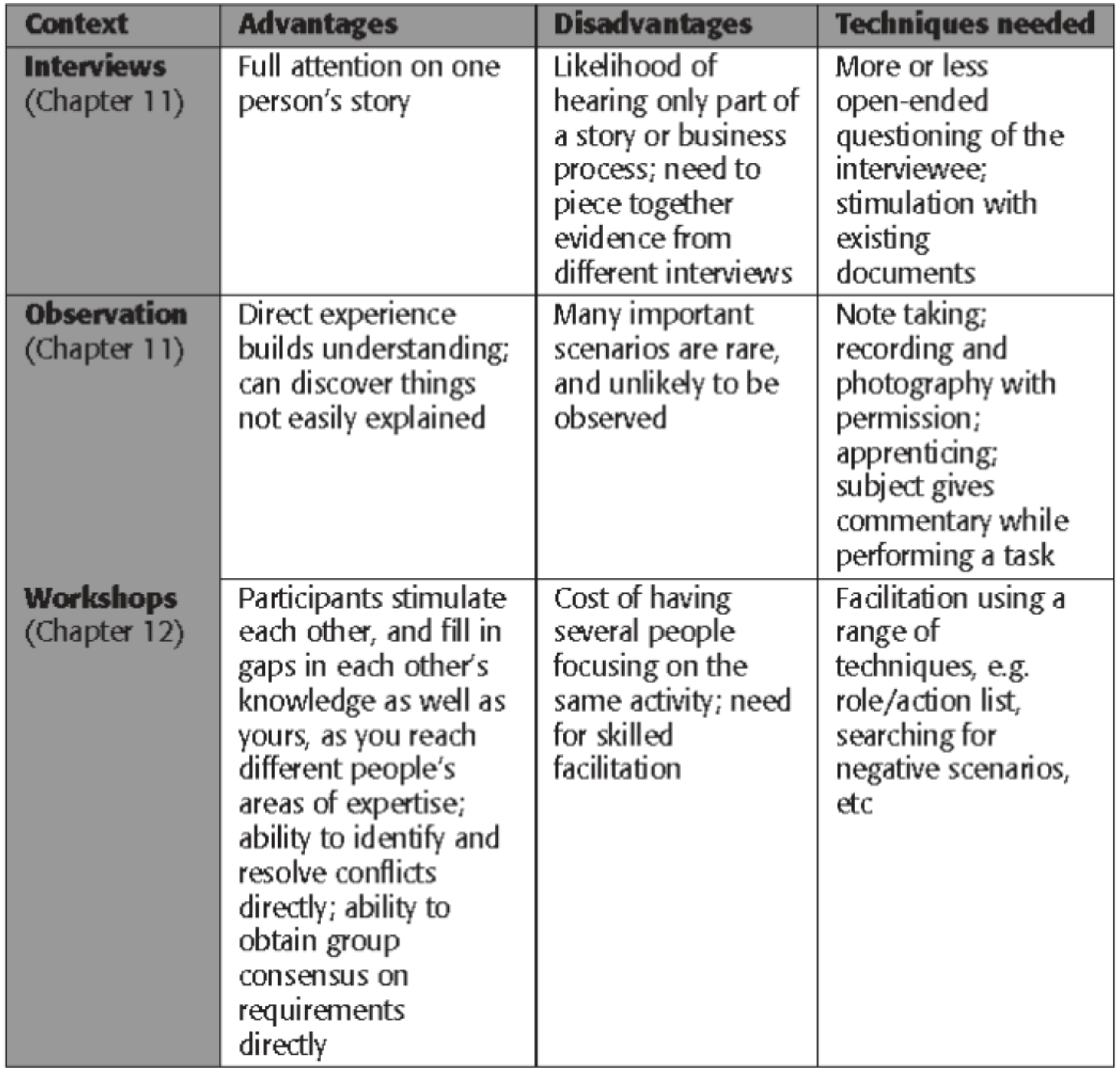
- To communicate a situation as it evolves trough time in a series of steps
- Scenarios communicate requirements very effectively
- When scenarios describe interactions of humans and products they refer to an interface

What Scenarios to Write?

- Stakeholders’ comments in interviews
- Goals in workshops
- known scenarios, e.g. problems with existing systems
- Study of the system context and scope
- Standard scenario patterns
  - Day in the Life of (DILO)
- Generally, the right level of scenario to start with is **a complete, end-to-end story** that deals with how a human operator uses a product to get results.
  - For instance, in an online bookstore, ‘buy a book’ is a complete story, whereas ‘add item to cart’ is a minor detail.
- It is also generally best to start with **normal, ‘happy day’** scenarios

Contexts for Discovering Scenarios (Alexander)

- Interviews
- Observation
- Workshops



Context	Advantages	Disadvantages	Techniques needed
Interviews (Chapter 11)	Full attention on one person's story	Likelihood of hearing only part of a story or business process; need to piece together evidence from different interviews	More or less open-ended questioning of the interviewee; stimulation with existing documents
Observation (Chapter 11)	Direct experience builds understanding; can discover things not easily explained	Many important scenarios are rare, and unlikely to be observed	Note taking; recording and photography with permission; apprenticing; subject gives commentary while performing a task
Workshops (Chapter 12)	Participants stimulate each other, and fill in gaps in each other's knowledge as well as yours, as you reach different people's areas of expertise; ability to identify and resolve conflicts directly; ability to obtain group consensus on requirements directly	Cost of having several people focusing on the same activity; need for skilled facilitation	Facilitation using a range of techniques, e.g. role/action list, searching for negative scenarios, etc

Interviewing Stakeholders

- Set the interview in context
- Limit the duration of the interview
- Have business use cases/diagrams as anchor for the interview
- Ask questions , listen to the answer and **feed back your understanding**
- Draw models and encourage the stakeholder to change them
- Use stakeholders terminology and artifacts
- Keep sample or copies of artifacts and log them for future reference
- Thanks stakeholders for their time and them what you learned and why it was valuable

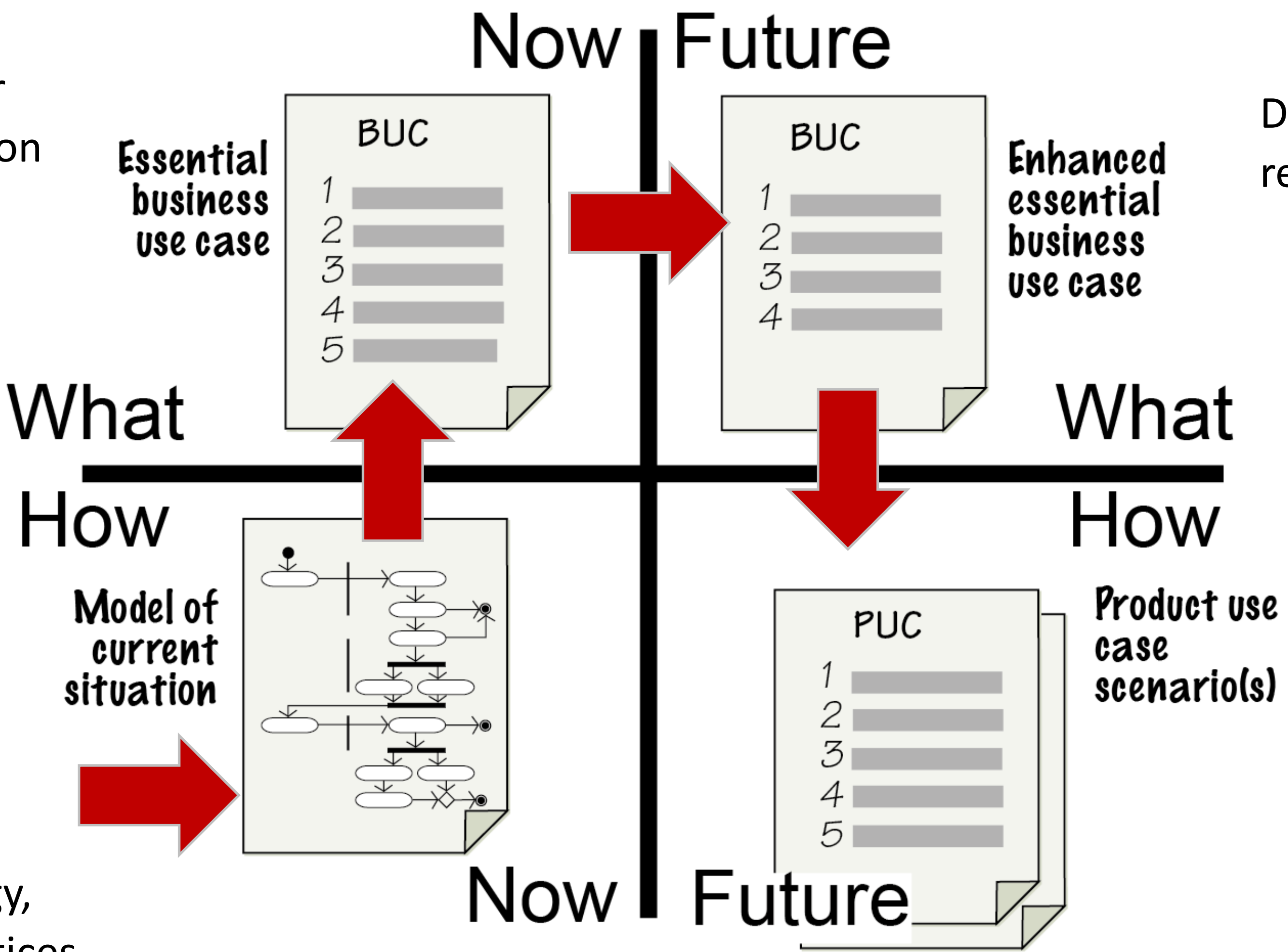
Where are we going? (brown cow model…)

Remove implementation or tech references and focus on what the system should do

Desired system without reference to implementation

System design in the solution space

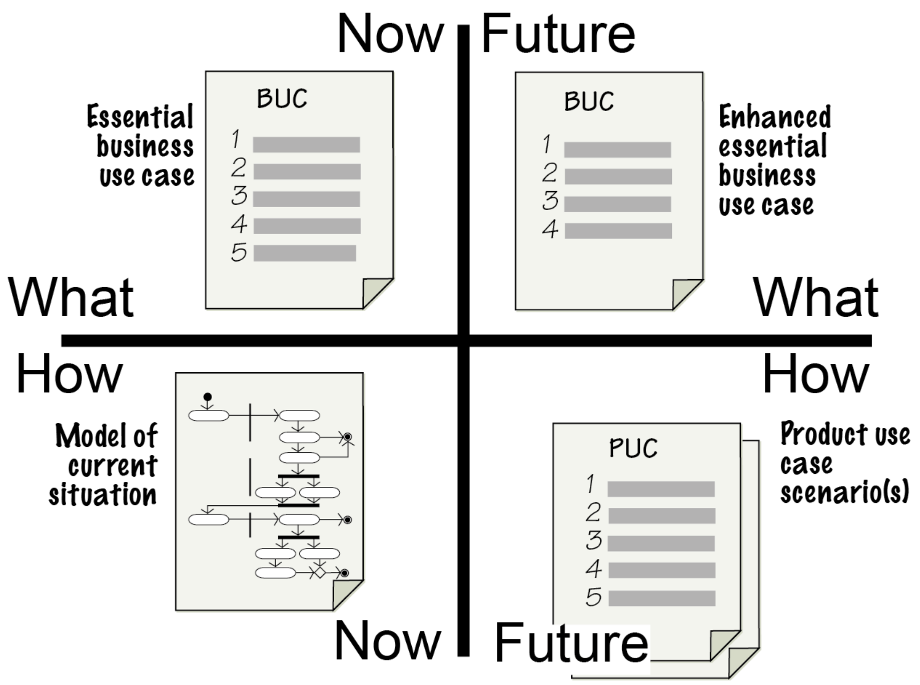
Start Here, include all references to technology, implementations , practices


Now
Future
BUC
Essential business use case
1
2
3
4
5
BUC
Enhanced essential business use case
1
2
3
4
What
What
How
How
Model of current situation
PUC
Product use case scenario(s)
1
2
3
4
5
Now
Future

https://en.wikipedia.org/wiki/How_now_brown_cow

Use the How-Now to How-Future to Guide the Information Gathering

- Ask for What , How, When, Where, Why, Who to encourage description as opposed to Y/N
- Start with questions on the current situation. Ask for artifacts and methods.
- Then transition to what future "Are there other facts about<ti that would be useful to you"


Now
Future
Essential business use case
BUC
1
2
3
4
5
BUC
1
2
3
4
Enhanced essential business use case
What
What
How
How
Model of current situation
PUC
1
2
3
4
5
Product use case scenario(s)
Now
Future

Scenario Workshops

- Making role/action lists
- Making operator actions/machine actions lists
- Asking about standard patterns
- Acting scenes

Things to Avoid

1. Making design decision without noticing
2. Model swim lanes in real time (it make take to point #1)
    1. Step back and provide a curated list at the end of the workshop

Summary of

- Start with happy day scenarios
- Capture each story as simple as possible
- Take into consideration scope:
  - Normal transaction
  - DILO
  - Maintenance, retirement
- Move to special cases
  - Exceptions detection and handling

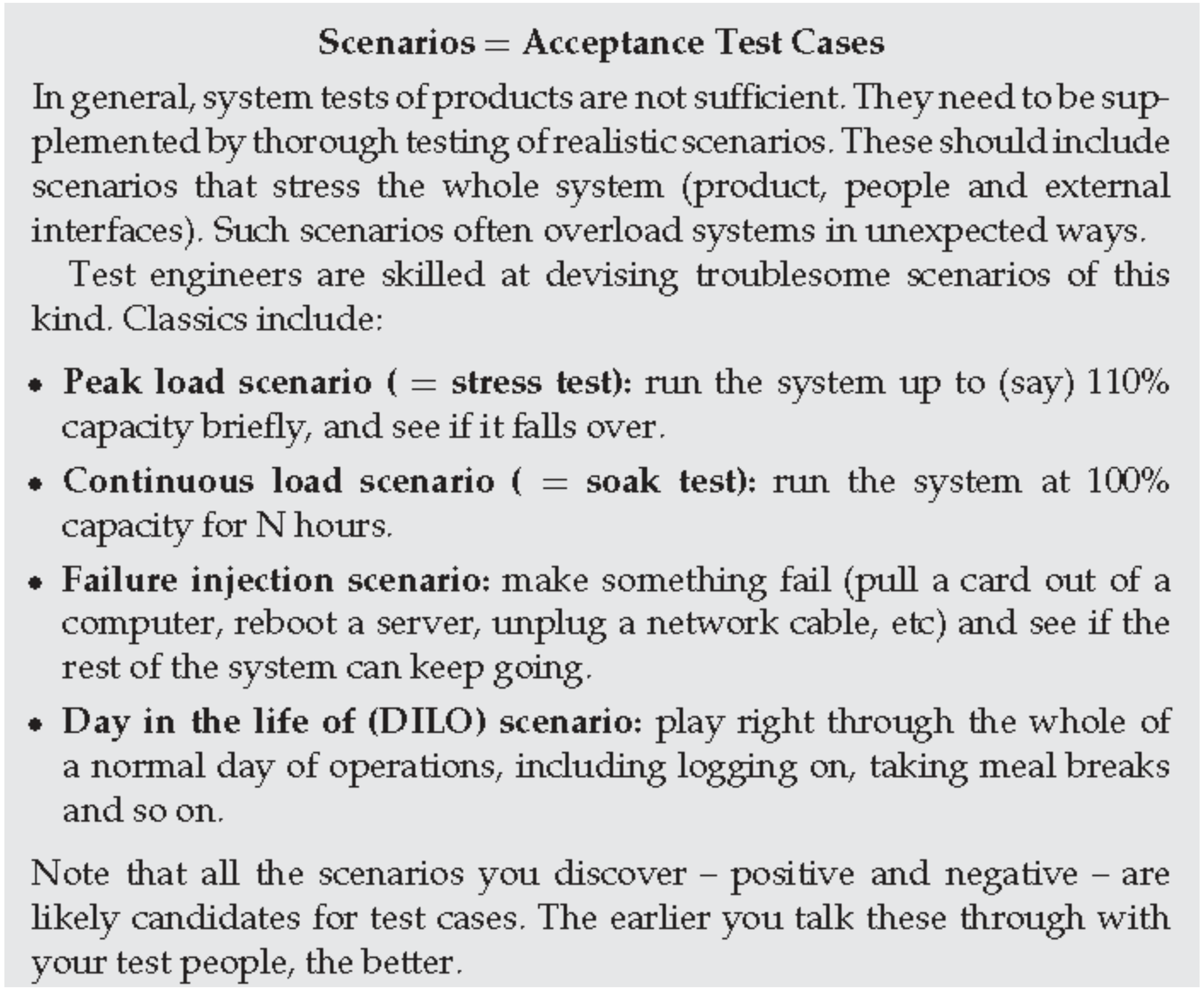
Negative Scenarios

- After the normal flow we need to test for things that should not happen (goal)
  - Exceptions: Events that interrupt normal progress
  - Intentional threats: Actions of hostile stakeholders which may cause exceptions
  - Unwanted scenarios: Undesirable combination of correct (allowed) behaviors
- Discovering negative scenarios
  - List events/triggers at the system interface
  - Define scope by agreeing upon all the events that will be covered by the product
  - Normal scenarios are complete when all the events are covered
  - List and agree on exceptions on the normal scenarios
  - Then what?
- Exception handling scenarios shall be simple, unambiguous and short
- Ignoring an exception is equivalent to accept a risk of failure
- For hardware components physical exceptions become important

Tips for Documenting Scenarios

- Decide what style will be most appropriate (colorful stories or something more analytic)
- Tell your story simply and directly
- Supply enough context to enable readers to see where the scenario fits in

Scenarios and Acceptance Tests



Scenarios = Acceptance Test Cases

In general, system tests of products are not sufficient. They need to be supplemented by thorough testing of realistic scenarios. These should include scenarios that stress the whole system (product, people and external interfaces). Such scenarios often overload systems in unexpected ways.

Test engineers are skilled at devising troublesome scenarios of this kind. Classics include:

- **Peak load scenario (= stress test):** run the system up to (say) 110% capacity briefly, and see if it falls over.
- **Continuous load scenario (= soak test):** run the system at 100% capacity for N hours.
- **Failure injection scenario:** make something fail (pull a card out of a computer, reboot a server, unplug a network cable, etc) and see if the rest of the system can keep going.
- **Day in the life of (DILO) scenario:** play right through the whole of a normal day of operations, including logging on, taking meal breaks and so on.

Note that all the scenarios you discover – positive and negative – are likely candidates for test cases. The earlier you talk these through with your test people, the better.

CLASS MINI PROJECT #1

- Identify a Problem that can be resolved with the use of an ARDUINO device. (EG next page)
- Perform a series of discussions and interviews to identify project goals (todays lecture time)
- Define project scope (1 slide)
- Create a context diagram (1slide)
- How-Now to How-Future (1 slide)
- Describe 2 scenarios (up to 2 slides)

EVERY TEAM MUST HAVE AN APPROVED PROJECT BY NEXT TUESDAY 5PM
EACH TEAM MUST PRESENT THEIR SYSTEM PROPOSAL AND SCOPE (THIS CLASS OR NEXT)
CLASSMATES AND GUESTS WILL VOTE FOR PROJECT ACCEPTANCE