Capturing Wild Requirements – For Testing

Silverpath Technologies Inc.
Trevor.Atkins@silverpath.com

Thinking Through Testing

Who Needs Requirements?

Silverpath Technologies

- Requirements information is a primary input to the majority of stakeholders' project activities.
  - **Customer:** Scope definition, description of business needs and users / roles, definition of acceptance criteria
  - **Marketing:** Promotion of system capabilities, competitive comparisons
  - **Technical Writing:** Creation of user manuals and tutorials
  - **Business Analysis:** Elicitation and capture of business logic and tasks from the customer and other stakeholders
  - **Project Management:** Scope management, risk planning, effort estimation, project goal setting, project and resource planning
  - **Development:** Design and coding
  - Testing: Verification and validation

Critical Communication

- *"The communication of functional requirements and specifications is the most difficult, critical, and error-prone task in IT projects."*

  Bill Walton, A Systematic Approach for More Effective Communication of Functional Requirements and Specifications

- *"The criticality of correct, complete, testable requirements is a fundamental tenet of software engineering. The success of a project, both functionally and financially, is directly affected by the quality of the requirements."*

  Theodore F. Hammer, Linda H. Rosenberg, et al., Doing Requirements Right the First Time!

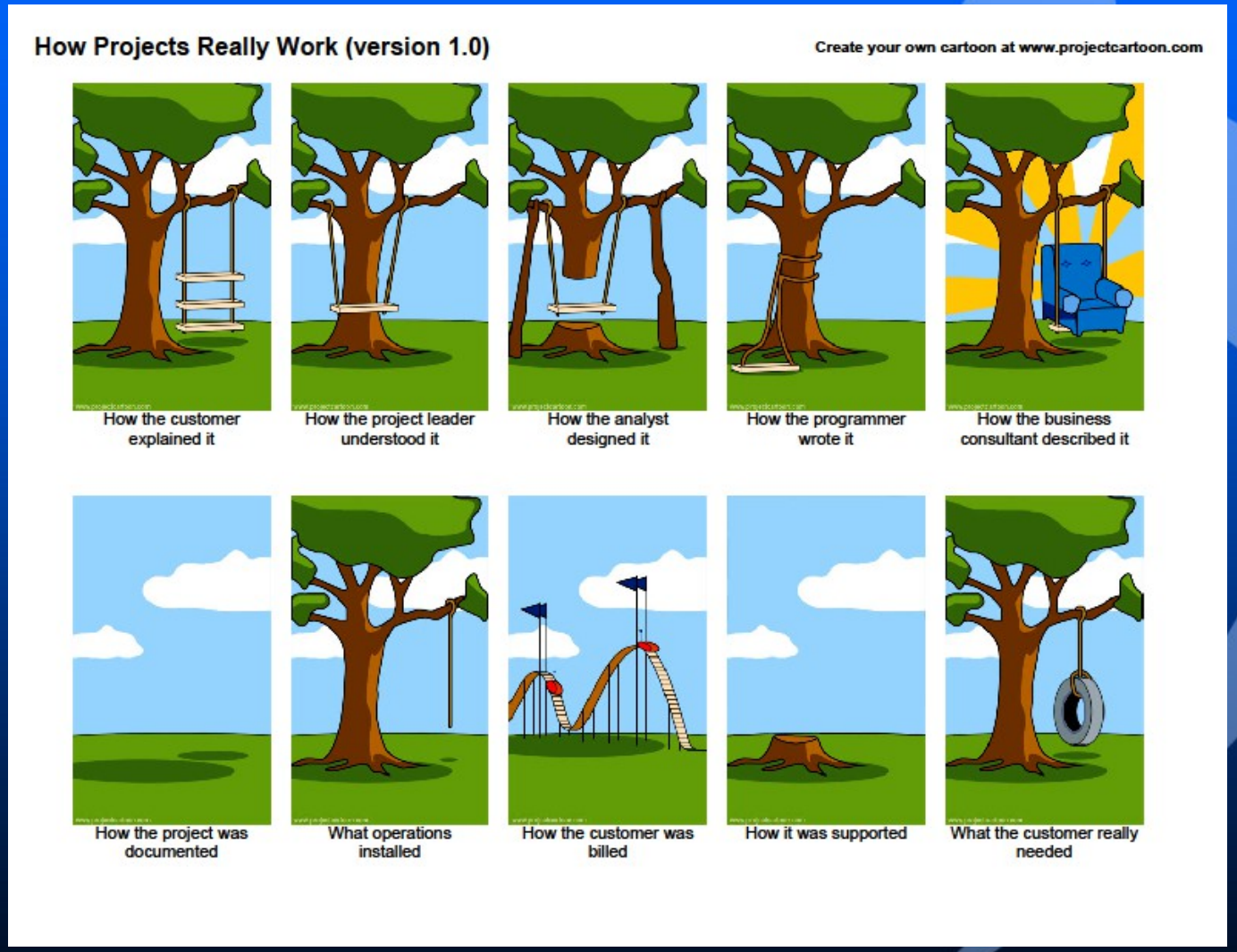
Wild Requirements

- Huge pressures to show tangible progress result in uncontrolled requirements that are often:
  - Incomplete
  - Lacking enough detail
  - Inaccurate
  - Unapproved
  - Out of date
  - Ambiguous
  - Just lacking / missing

Wild!

“How Projects Really Work”

- Consider:



Importance of Good Writing

Silverpath Technologies

- *“People cannot write well about things they are unsure of or do not understand. Therefore, if you find yourself unable to "translate" a requirement or feature, the developer may be fuzzy on the issue as well.”*

- *“This kind of inability to write about an area of the system under test should raise an immediate red flag. If you don't understand it, how can you test it? And if the developer is unclear about it, you've just found a potential nest of bugs.”*

Mike Alexander, Good Writing Leads to Good Testing

Example User Scenario

Example user scenario: The following provides an example outline of a user scenario:

Name: Making a Call – Originator Is Inside Caller

Actor(s): Inside Caller, (Outside Caller, Receptionist)

Basic Flow:

1.0 Inside Caller picks up receiver of handset
2.0 Inside Caller waits for dial-tone
3.0 Inside Caller dials a phone number on handset keypad. In the case of an outside destination the Receptionist's handset will show a new line is now in use.
4.0 The call is answered by the destination
5.0 Inside Caller and the destination are able to speak to each other
6.0 Inside Caller hangs up receiver of handset

Alternate Flows:

3.1 Inside Caller may dial '0' to reach the Receptionist station handset
3.2 Inside Caller may dial a 3-digit extension starting with the digit '1' to reach any other active extension within the corporate telephone system connected with a station handset
3.2.1 If Inside Caller dials an inactive extension an error message will be given
3.3 Inside Caller can dial an outside phone number to reach Outside Caller by first pressing the 'outgoing' button on the handset keypad
3.3.1 If Inside Caller dials a long distance number as the outside phone number, a 4 digit passcode will be required to complete the dialing
3.4 Inside Caller can connect to emergency services by dialing the 3 digits '911'

Pre and Post Conditions should also be included when significant.

- Find the errors & missing paths?

Draw Some Pictures

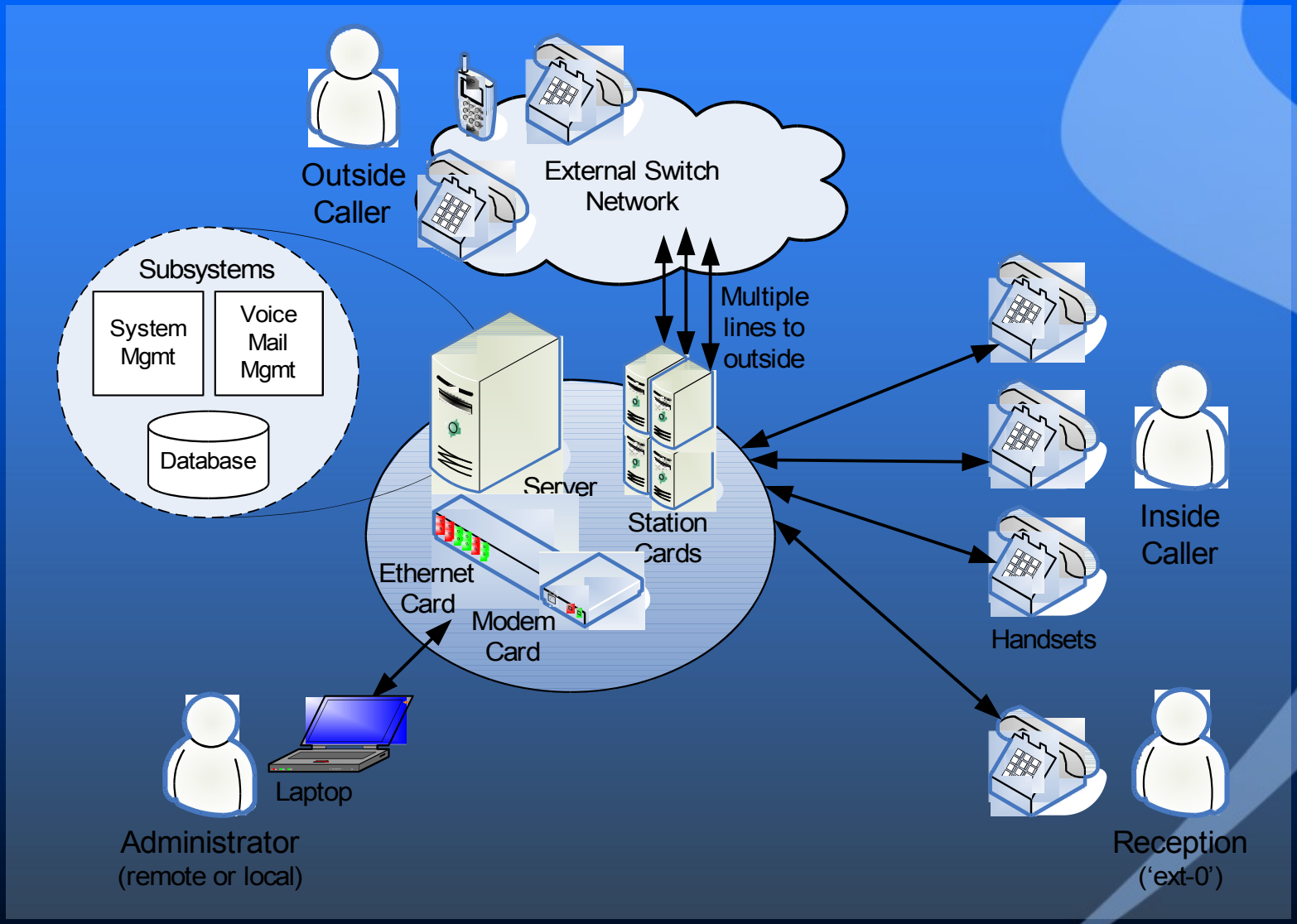
- A first step in capturing a useful understanding of the system to be tested is to think in pictures
- Every picture helps tell a story and stories or scenarios form a basis for analysis and testing

Start with “W5”

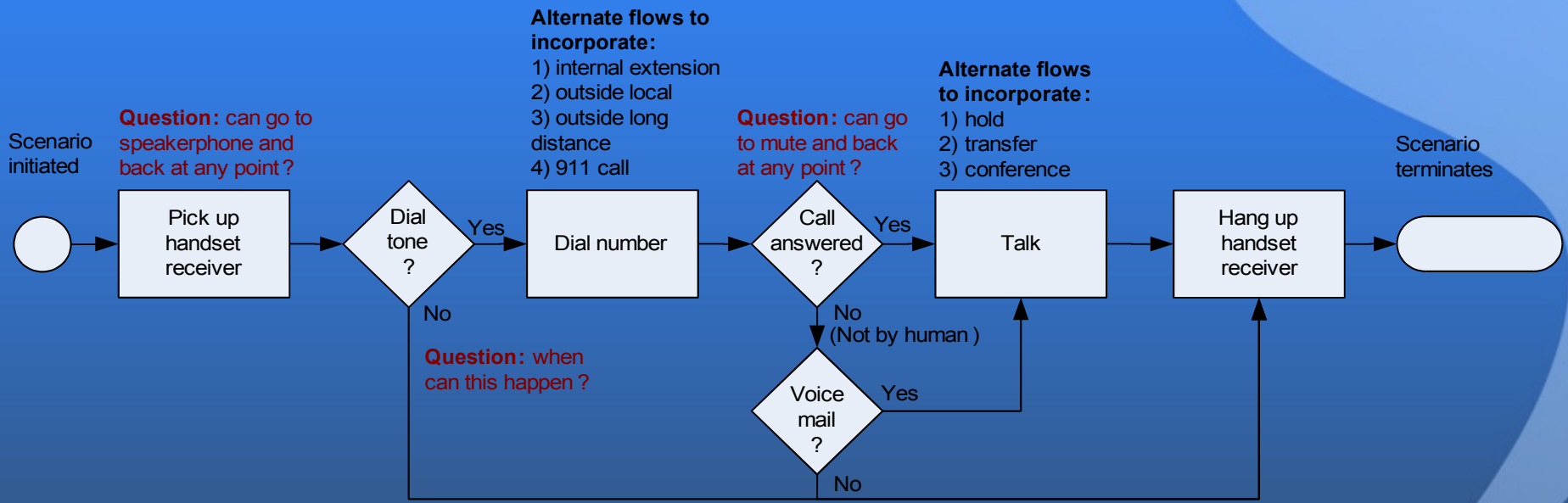
Silverpath Technologies

- Who are the end users of the system?
- What are the tasks to be performed?
- Where will the system be deployed?
- When are there interactions between modules?
- Why is the system being built?

System Overview

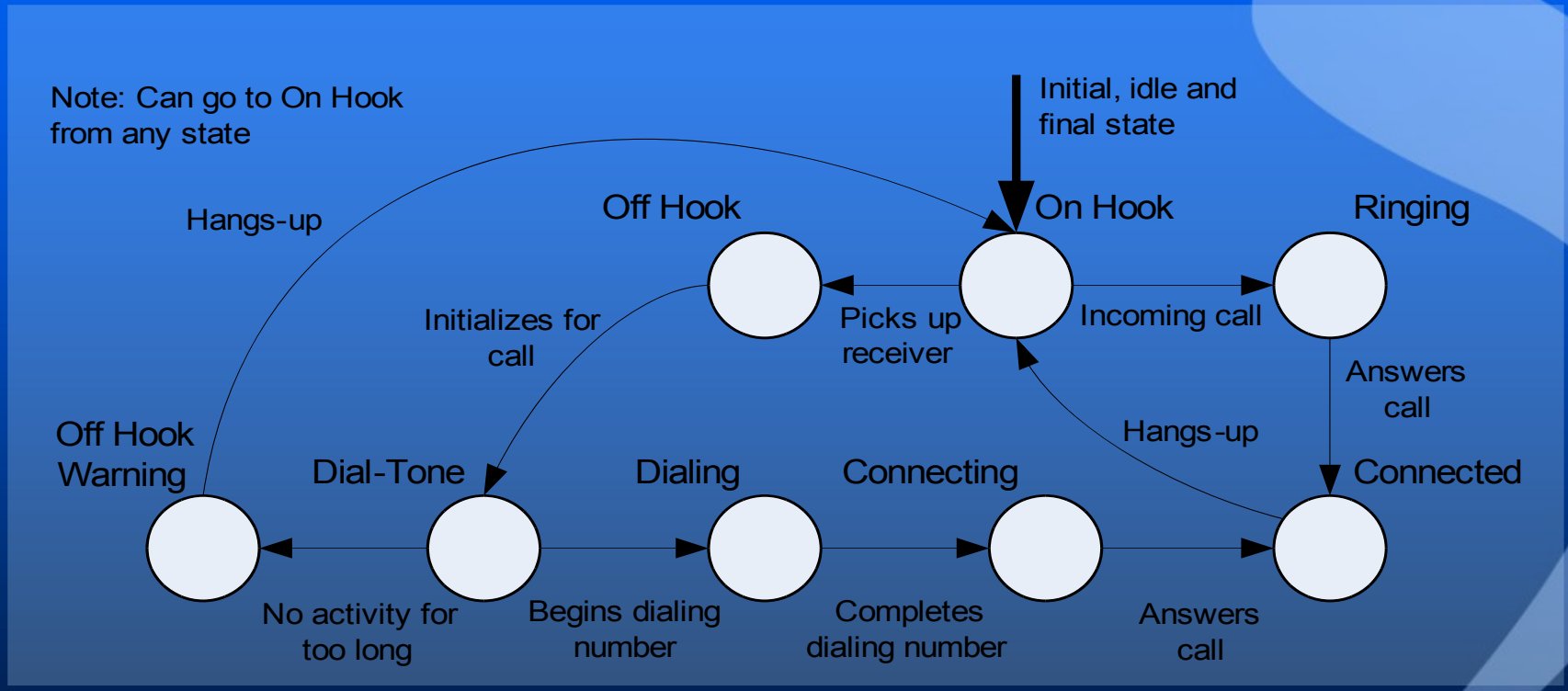


Flowchart



- What's still missing?

State Diagram



- What's incorrect?

Requirements Taming Tips

- Flowcharts and state diagrams are powerful visualization tools
  - Capture the behaviours and functionality of a system
  - Establish path coverage and ideas for error path or negative testing
  - Annotate the diagrams to record cases and ideas
- Add scenario-oriented narratives to complement and expand upon the pictures
- Use checklists and matrices to track the progress of that testing

Value of Pictures

- Humans think easily in terms of pictures → use them to document fast and review faster

- *"Pictures can pack a great deal of information into a small space. They help us to see connections that mere words cannot."* Elizabeth Hendrickson, A Picture's Worth a Thousand Words

- What other visual representations could you use for capturing requirements?

Thinking Through Testing

Silverpath Technologies

For our latest updates:

- Visit ThinkTesting.com
- Follow @ThinkTesting

Discussing "right-fit" approaches for software testing

We are always sharing our ideas on crafting “right-fit” approaches to software testing. We are sure you will find something you can apply to your own projects and organizational environment.

Thinking Through Testing