Risk Analysis & Management

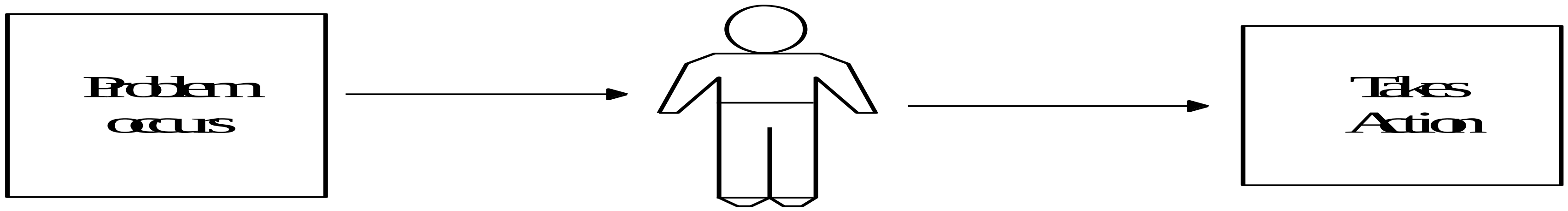
By: Dr. Ali Rehan Syed.

Risk strategies

- According to Webster’s Dictionary,
  - ***Risk*** is the possibility of suffering loss.
- Any decisions that we take generally carries some amount of risk.
- Any decision we take may result in a gain or loss for us. In both cases, we must bear the consequences.
- Risk involves change in some form like a change in place, a change in profession and change in opinion.
- Software engineering also entails many risks.
- At every step, we must make a choice be it the selection of people for the project or the selection of equipment to be used.

Risk strategies

- Two commonly used strategies to deal with risks in SE.
  - ***Reactive strategies***
  - ***Proactive strategies***.
- There is a phrase “cross the bridge when you come across it”, which when applied to a software project would mean taking care of problems as and when we come across them without planning for them in advance.
- This is termed as “**Reactive strategy**”, which means we react to the problem.
- With reactive strategies, the team does nothing about risks until something goes wrong.
- When a problem arises, the team goes into action and takes initiative to solve the problem.
- Reactive strategy can be depicted as Figure.

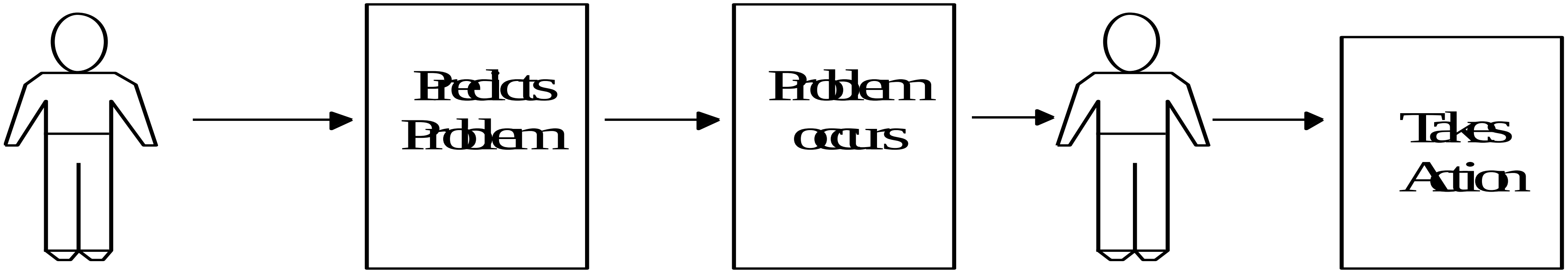


Reactive strategy

- Assume that there was a software project in progress with three software engineers involved in all kinds of responsibilities.
- The firm never accounted for the fact that any one of them may leave. It simply took for granted the continuance of the staff. But, all of a sudden, two of the engineers leave the project.
- In such a situation, the entire burden falls upon the lone software engineer.
- Due to shortage of time, it may so happen that the firm is unable to hire new staff immediately and the project may have to be abandoned or delayed.
- In the given example, reactive strategy would mean hiring and training new staff after the earlier staff has quit.
- Had the problem been foreseen, the engineers who plan to quit could be made to train the new staff and then leave.

Proactive strategy

- Proactive means taking the initiative.
- **Proactive strategy** with respect to a software project would mean acting in advance, predicting that a problem may arise and taking precautionary measures for it.
- Identification of risks that may occur, prioritizing them and planning solution strategies to deal with these risks are part of proactive strategy.
- Figure illustrates this strategy.



Characteristics Of Risks

- Any kind of risk possesses two characteristics –
- ***Uncertainty:***
  - Uncertainty is probability estimation about whether the foreseen risk will actually occur.
- ***Loss:***
  - If the risk becomes a reality, unwanted consequences or losses will occur.
  - Loss is the direct consequence of the risk actually taking place.

Software risks

- Risks can be categorized into several categories

- ***Project risks:***
  - These affect or threaten any project plan that has been laid.
  - Project risks can cause time delays or increase in cost.
  - These risks include personnel, customer, resource, budgetary, scheduling and requirement risks.

- ***Business risks:***
  - Business risks affect the viability of the software to be built.
  - Business risks can come in various forms:
    - building a product that the marketing team does not know how to sell,
    - building a product that does not fit into the business strategy of the company,
    - building a superlative product that no one wants to use.
    - Sometimes, business risks are unpredictable too.

Software risks

- ***Technical risks:***
  - These risks can have an effect on the creation and completion of software.
  - In case such a risk becomes real, implementation could become difficult and even impossible.
  - Technical risks can occur within any of these stages: implementation, interfacing, verification, design, and maintenance.
  - The causes for technical risks could be software obsolescence, technical uncertainty, and ambiguity in specifications.
- ***Predictable risks:***
  - Certain risks like unrealistic delivery date, lack of documented requirements or software scope can be predicted by evaluating the project plan and the environment under which the project will be developed.
  - Predictable risks can be identified from past project experience.
- ***Unpredictable risks:***
  - These are the risks that may occur but cannot be predicted in advance.

Risk identification

- Risk Identification is a systematic attempt to specify factors that may affect the project plan.
- It is important to identify the known and predictable risks, can be avoided or controlled.
- Two types of risks -
  - ***generic risks***
  - ***product-specific risks***.
- Generic risks can affect any software project.
- whereas product specific risks can be identified by those who are familiar with the technology, the people and the project at hand.
- Detailed examination of the project plan is necessary to find out and identify the product specific risks.

Risk identification

- A ***risk item checklist*** is generally created to help in identification of risks as well as focus on different categories of risks.
- The checklist will consist of the following parameters:
  - **Product size**
  - **Customer characteristics**
  - **Process definition**
  - **Business Impact**
  - **Development environment**
  - **Technology**
  - **Risk associated with staff size and experience**

Assessing Overall Project Risk

1. Have top software and customer managers formally committed to support the project?

2. Are end-users enthusiastically committed to the project and the system/product to be built?

3. Are requirements fully understood by the software engineering team and their customers?

4. Have customers been involved fully in the definition of requirements?

5. Do end-users have realistic expectations?

6. Is project scope stable?

7. Does the software engineering team have the right mix of skills?

8. Are project requirements stable?

9. Does the project team have experience with the technology to be implemented?

10. Is the number of people on the project team adequate to do the job?

11. Do all customer/user constituencies agree on the importance of the project and on the requirements for the system/product to be built?