

# Chapter 4

## Explanations

# Additional Recommended Literature

- [Wooley 98] Wooley, Bruce A.: Explanation Component Of Software Systems. (1998).

Part 1

General Aspects

# Types of Explanations

- Explaining a term or concept:
  - Presents a description or definition and additional aspects like when and how to use the term.
- Explaining a procedure: How and why something was done.
- This can be done for two reasons:
  - To increase the trust of the user
  - To to give an insight into the main reasons for what happened with the intention to improve the process (interactively).
- We will be concerned with the last aspect.

# Explanations and Processes

- In an interactive processes it occurs often that the user does not understand certain results that are presented by some agent.
- Possible consequences:
  - Results are not accepted
  - Wrong or unwanted reactions
  - No feedback possible
- The last two points are in particular important if the results depend on one or more earlier decisions of the user:
  - The user cannot determine the influence of the decisions to the results and therefore no proper reaction (e.g. revising some decision) is possible.

# Explanations and Interactivity

- The purpose of an explanation is to give the user so much insight that undesired consequences do not occur.
- This means:
  - The explanation informs the user about the major influence factors for the results.
- Some of the influence factors are facts:
  - If the user is informed then the result is more likely to be accepted.
- Other influence factors depend on decisions (of the user):
  - These decisions are now subject to a revision
- Influence factors may also depend on estimates:
  - The estimates may be reconsidered.

# Explanations versus Proofs

- A proof is an argumentation that shows that the result is absolutely true in a logical sense.
- But a proof
  - does not provide evidence of what was more or less important
  - does not talk about causality
  - does not take care of the user:
    - What does the user understand?
    - What does the user want to know?
- An explanation takes care of these aspects:
  - It has a user model
  - It concentrates on major issues

# Explanation Component

- This is a special component (a software agent) that connects the user with other software agents in a special way.
- It is some kind of translator between the software and the user. Therefore
  - the translator has to have access to internal life of the software
  - has to have knowledge about the needs and desires of the user.

# User Models

- It is difficult to foresee what the user might want to know or which knowledge is missing, even if the general type of user (here: the project manager) is known.
- A way out: The user models him/herself.
- How can that be achieved:
  - The user investigates the result. If the user is satisfied: ok!
  - In case the user has a problem with the presented solution the the user can usually point to some part of the solution that gives rise to a difficulty in understanding.
  - As consequence, the user is allowed to ask a question.
  - Such a question will be answer, first on a very general level.
  - In a dialog follow-up questions can be raised.

# The Project Manager

- In this lecture the user will be the project manager of a software company.
- The manager makes certain decisions:
  - on the importance of tasks
  - on how to do something
  - on the allocation of resources
  - on estimates for certain magnitudes like effort needed for a task.
- These decisions are made with the intention to plan and execute the process in an optimal way.
- Because of the complexity of the processes and the fact that unforeseen events can occur some decisions may not be optimal and have to be revised.
- Our main application is scheduling.

# The Dialog Approach

- During the dialog two goals are achieved:
  - The user model is generated and completed
  - The user is iteratively informed.
- The user determined when the dialog is finished, in particular how many details are provided.
- This requires:
  - to model questions and answers;
  - a coupling of the dialog component and the problem solver (e.g. the scheduler).
- The dialog takes place between the project manager and the explanation component.

# The Dialog as a Recursive Process

- 1) The user has a choice of initial questions. This choice allows to select a question and to specialize it.
- 2) As a result, the user sees an answer. Now the user can select some item of the answer. As a result, a new but more detailed answer shows up.
- 3) This process can be iterated.
- 4) The user determines when to stop the dialog

# Questions and Answers

- The concept of *question* is related to the concept of *answer*.
- An answer can be a true or false statement.
- The main point is that an answer has to be *answer to the question*.
- Examples:
  - What is a prime number?
  - Meaningful answers: A definition or an example
  - Question: How many persons work in project P?
  - Meaningful answers: 1, 2, 3, ..., many, few, nobody, all, ...
  - But not: Project P needs 3 persons (even if this is true)
  - Question: Why is Bill not in his office today?
  - Meaningful answers: Because he is sick, he visits a customer, ...
  - But not: Bill is an expert in data bases (even if this true)

# How to Formulate Questions and Answers

- There are many ways to express questions and answers:
- Natural language:
  - Needs a natural language processor
- Frequently Asked Questions (FAQs):
  - Allows only finitely many questions. In order to increase flexibility tools like thesauri etc. are required
- Menues:
  - Very restricted
- Question and answer types:
  - We will follow this paradigm. It is sufficiently flexible and efficient for the use in an explanation component for software developing processes.

# Question Types

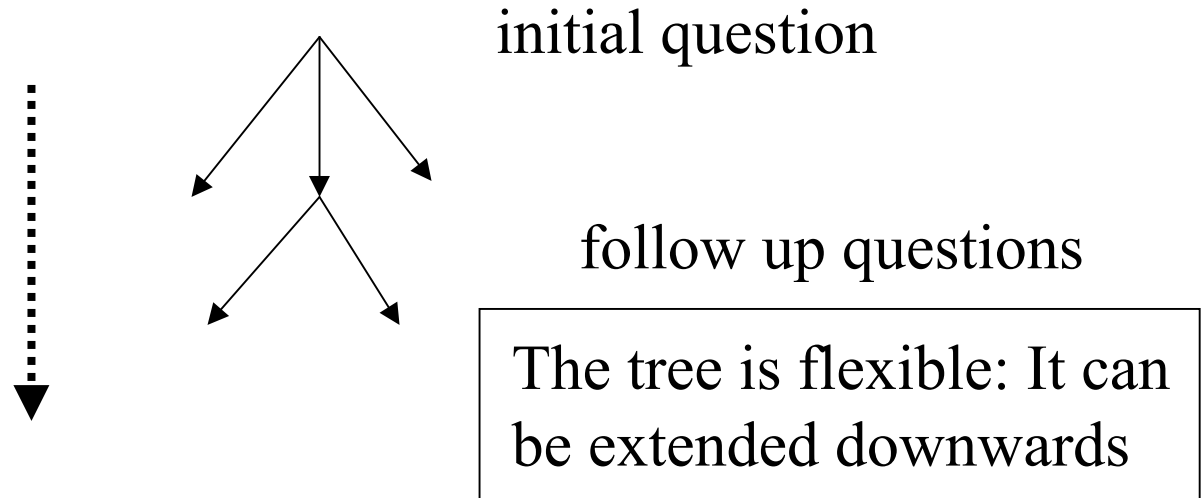
- The two most important question types are:
  - Why was task X not in release i?
  - Why was task X in release i?
- Related is:
  - Why was task X scheduled before task Y?
- These question types are called *initial questions*.
- Here X and Y are variables for tasks and i ranges over natural numbers.
- Instantiating X, Y and i yields a specific question.

# Presenting Question Types

- The user sees a window that presents the question types.
- The user is asked
  - either to terminate the dialog or no (further) explanation is wanted
  - or to select a type and to instantiate the variables.
- The a question is generated and sent to the explanation component.
- In this way the user can precisely formulate what kind of explanation is wanted.

# The Question Hierarchy

- The question types are organized in a question tree:



- The role of the question tree is twofold:
  - The dialog follows the directions of the tree
  - The answering machine follows the direction of the tree

# Answers

- To each question the set of *meaningful answers* is associated. This is also called the *set of alternatives* or the *contrast class*.
- Example:
  - Question: Why was the graphics program G123 not used?
  - Set of alternatives: {G123 is no longer available, G123 is too slow, G123 could have been used but is not better than our choice}.
- One of the alternatives was selected and was the reason for the decision.
- Gthis does not mean that this alternative is true: The decision maker might have believed that it was true but made an error.

# Answer Types

- The answer types contains in addition to question variables other variables that are useful form the explanation.
- The answer type is defined in such a way that the instances cover all explanations that are meaningful answers.
- The task is to select those instances
  - that apply in the specific situation
  - that are true in the specific situation
- The answers can be given on different levels of detail.
- The dialog strategy should go iteratively from very general to more specific explanations.

# Example

- Question type:
- Why is task X not in release i ? (X is in release j,  $j > i$ ).
- Answer type:
- X was preassigned to release j
- There is coupling constraint between X and Z and Z is in release j
- There is a precedence constraint between Y and X and X is in release j
- The majority of stakeholders considered all tasks in release i as more important than task X

# Rule Based Explanation

- Rules are of the form  $\phi_1 \wedge \phi_2 \dots \wedge \phi_n \rightarrow \psi$  ; the  $\phi_i$  are the premises and  $\psi$  is the conclusion of the rule.
- A rule system also contains facts.
- A solution is obtained by applying rules iteratively to facts.
- The explanation follows the parsing paradigm:
- The inference process is reconstructed by applying the inference steps backwards. This explains how the result was obtained.
- This method works if not too many rules are involved.

# Constraint Based Explanation

- Constraints are n-ary predicates  $P(x_1, \dots, x_n)$ ; in the solution the variables are replaced by constants  $a_i$ .
- The definition of a solution to a constraint system is that it satisfies all constraints.
- Each constraint system has its own inference strategy.
- The explanation follows again the parsing paradigm.
- Often it is possible to name constraints that exclude possibilities for plausible solution candidates.

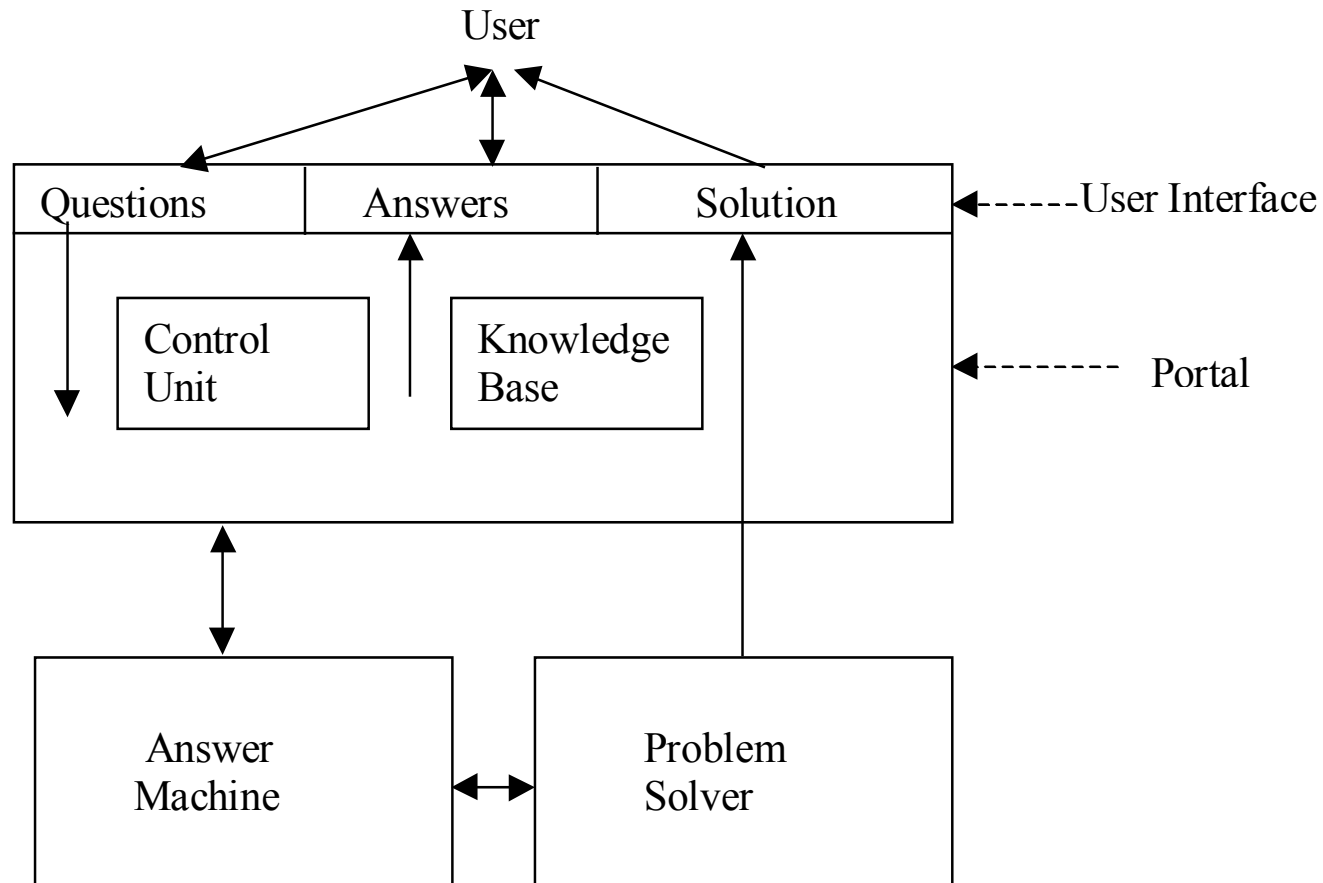
# Optimization Based Explanation

- An optimization procedure is a complex black-box algorithm; the problem is coded into the algorithm.
- Therefore the parsing paradigm is not applicable.
- Our explanation strategy is as follows:
  - The solution is presented to the user.
  - In case the solution is unsatisfactory the user has to name what should have been obtained.
  - This is enforced as a new constraint and given again to the optimization procedure.
  - The two results, the old and the new one, are compared and analyzed by the explanation component and presented to the user.

## Part 2

## Architectures

# An Architecture



# The Answer Machine

- The purpose of the answer machine is to generate the answer.
- It accepts the instantiated question and
  - communicates with the problem solver (e.g. the scheduler)
  - analyzes the situation
  - simplifies the results and maps it to an instance of the answer type.

# Explanations for Scheduling and Release Planning (1)

- Purpose: To inform the project manager about reasons for results of the planner that seem to be unsatisfactory.
- There are two possible reactions of the manager:
  - The manager becomes convinced that the result is good and will no longer object.
  - The manager sees that some reasons for the unsatisfactory result rely on his/her own decisions/estimates which should be revised.
- Example: The question is:  
Why is task X not in release  $R_i$  ? (X is in release j,  $i > j$ ).

# Explanations for Scheduling and Release Planning (2)

- As a result of this question a new problem will be presented to the planner.
- It differs from the old problem by adding the preassignment

$X$  in release  $R_i$

as a new constraint.

- As a result a new plan will be generated.
- Both plans, the old and the new one will be given to the answering machine.

# A Simple Answering Machine

- Input: Two plans,  $\text{plan}_{\text{old}}$  and  $\text{plan}_{\text{new}}$ .
- The answering machine will analyze and comment the differences between these plans.
- These comments will constitute the explanation.
- The analyzer needs therefore access to certain details that will be described in the sequel. These details are, however, not concerned with the interior of the optimizer.

# The Analyzer (1)

- There are two releases involved,  $\text{release}_i$  and  $\text{release}_j$ . Both releases have two versions:  $\text{release}_{i_{\text{old}}}$  and  $\text{release}_{j_{\text{old}}}$ ,  $\text{release}_{i_{\text{new}}}$  and  $\text{release}_{j_{\text{new}}}$ . The analyzer performs the following steps:
    - 1) Compute differences for  $\text{plan}_{\text{old}}$  and  $\text{plan}_{\text{new}}$  :
      - tasks out of  $\text{release}_{i_{\text{old}}}$  and  $\text{release}_{j_{\text{old}}}$
      - tasks in to  $\text{release}_{i_{\text{old}}}$  and  $\text{release}_{j_{\text{old}}}$ .
      - The totality of these tasks is called  $\text{diff}(i, j, \text{old}, \text{new})$ .
    - 2) List all
      - pre-assignments
      - coupling constraints
      - precedence constraint
- in which any of the tasks from  $\text{diff}(i, j, \text{old}, \text{new})$  are involved.

## The Analyzer (2)

- 3) List all votes of stakeholders for tasks in  $\text{diff}(i, j, \text{old}, \text{new})$ .
- Investigate first the constraints, e.g.:
  - Were they responsible for other elements out of  $\text{task}_{\text{old}}$
  - Are normative constraints involved?
- Investigate the votes, present the majorities and the importance of the stakeholders.

# The Reaction of the User

- First possibility: The user is now convinced that the solution should be accepted (e.g. because the user was not aware of the impact of some constraints).
- Second possibility: The user realized that some of his/her opinions are responsible for the unwanted elements of the solution. The user may not have given them as input if the consequences would have been obvious.
- As consequence, these opinions (e.g. votes or pre-assignments) will be revised.
- This makes scheduling an interactive process.

# Discussion

- When the user formulates a question to the explanation component this is some kind of user feedback.
- The question expresses that the objective function seems not to reflect the interest of the user in an adequate manner.
- The explanation is used to improve the solution.
- But: Only the specific solution is improved. The explanation has no further consequences for future problems.
- In particular, the explanation and their consequences are not recorded as experiences.

# Outlook: Learning

- A very general form of feedback is to rank two or more possible solutions to a problem: This means there is a partial ordering („better“) on these solutions.
- The user has no definition of this ordering but can decide in each particular case which solution is better.
- This is the classical situation where learning takes place: One has to learn this partial ordering from the feedback of the user.
- The learning can be done by humans, by machine learning methods or by a combination of both.