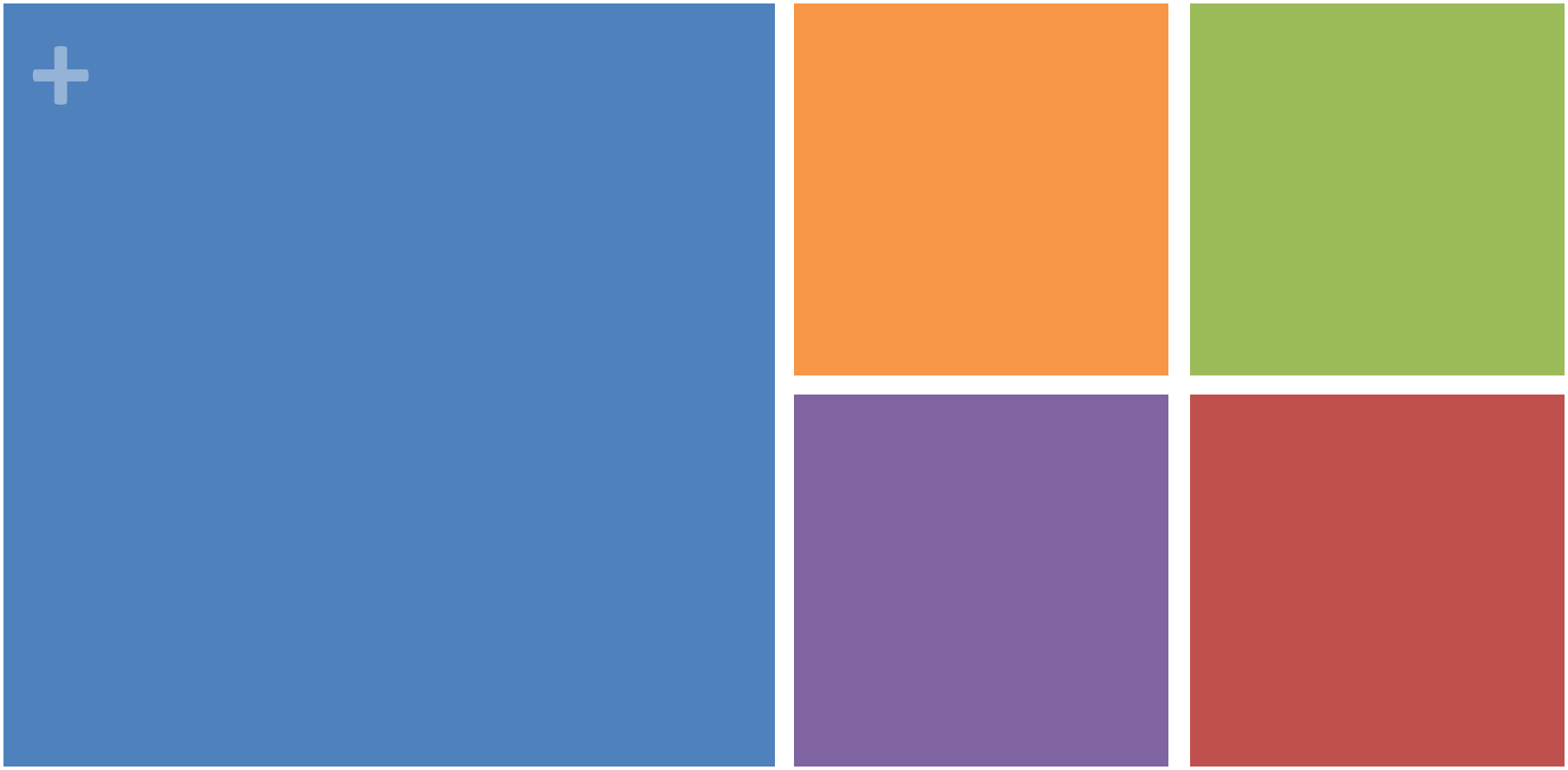

SWE3033
Software Processes

Descriptive Process Models
(Chapter 3 - Part 1)

Chapter's Objectives

- Descriptive Process Modeling Goals
- Creating Descriptive Process Models
- Process Elicitation Guidelines
- Assess and manage the basic risks associated with a descriptive process modeling effort

Today

- Introduce Goals of Descriptive Process Modelling
- Discuss Challenges of Process Engineers
- Introduce Systematic Approach for Creating and Validating a Descriptive Process Model
- Introduce a Systematic Approach to Creating a Descriptive Process Model for an Organization

Descriptive Process Modeling

- As its name implies, this discipline is concerned with producing an explicit and accurate representation of an organization's actual processes, for purposes such as documentation, dissemination, analysis, and improvement.

Goals of Descriptive Process Modeling

- Stable and Accurate Process Execution
- Process Understanding
- Process Propagation
- Process Measurement
- Process Administration
- Process Automation

G1: Stable and Accurate Process Execution

- Software processes can be very complex, involving large numbers of interrelated activities and complex interdependencies
- There is a high risk that some activities are not planned or are skipped because of external pressure. Also, there is always the risk that people will change the process over time without even noticing it, maybe eliminating beneficial activities.
- A documented process helps to mitigate these risks :
  - both project planners and project performers can make sure that they are performing all important and expected activities
  - Also, a documented process makes it easier to introduce changes in a controlled fashion, which leads us to our next point: understanding the process.

G2: Process Understanding

- Software development processes must change for a number of reasons.
  - Ex. external factors—such as changes in laws and regulations, or technological progress, introduce improvements
- **Risk**: changes that are originally perceived as improvements end up having unintended, detrimental consequences, which may negatively affect the overall process performance.
- Understanding the process is fundamental for managing this risk. An explicit process representation makes it much easier to assess the overall impact of process changes, thus allowing better management of process changes.

G3: Process Propagation

- Different development groups in large organizations follow different processes for the same task.
- These process differences are related to the fact that different groups may have different needs.
- By explicitly describing the processes followed by various organizational units, it is easier to achieve a unified process.
- A unified process is important because it allows for practices that are widely accepted as being beneficial, to be propagated to all units in a large organization.

G4: Process Measurement

- As an organization reaches a higher degree of process maturity, it becomes increasingly important to measure process performance.
- Aspects to be measured: its productivity (amount of output produced vs. resources invested into producing it), its ability to deliver results on time, and its ability to detect and correct product defects as early as possible.
- Measuring a process involves acquiring quantitative information about the actual activities, products, and resources involved.
- This is a hard task requiring high levels of discipline in order to yield satisfactory results, and which therefore must be carefully planned and executed.
- Proper process description greatly facilitates such planning by making it easier to identify appropriate measurement objects and to specify the procedures necessary to measure them.

G5: Process Administration

- In the long term, one of the main reasons for explicitly describing an organization's development processes is to be able to define goals for the process and work systematically on achieving them.
- Long-term goals for a process are often related to improvement (e.g., increase product quality, reduce time dedicated to product rework, etc.)
- E.g. many companies are forced by legal regulations, to comply with existing norms and regulations regarding the way their products are produced and validated.
- For these organizations, one important reason for modeling their processes explicitly is to guarantee that they comply with whatever norms and regulations are considered relevant.

G6: Process Automation

- It happens quite often that once a process has been properly described and stabilized, opportunities for supporting parts of it with automated tools become evident.
- Analyzing a detailed process description is probably one of the best ways to determine the exact requirements of new process-supporting tools.
- This type of support is especially useful in modern development environments, where developers are frequently distributed geographically, often across several time zones.

Challenges of Process Engineers

- Process engineers working on describing an organization's processes are confronted with two main challenges:
  - Process elicitation
  - Storing the process knowledge for future use

Challenges of Process Engineers - Process elicitation

- It is about collecting information about how the process is actually performed. This can be done by:
  - Working together with the actual process performers
  - Observing their work
  - Analyzing the results of their work such as the documents produced during past development projects
  - Interviews

Challenges of Process Engineers - Storing the process knowledge for future use

- The process-related information must be expressed in a clear and unambiguous way that is amenable to the goals of the modeling effort (e.g., process measurement, improvement, analysis,…)
- A number of notations and modeling tools are available to facilitate this task
- Notations allow expressing processes using a predefined, standardized set of concepts. They can be formal or graphical.
- Tools help process engineers to deal with the complexity of process models

Creating a Descriptive Process Model - Approach Overview

- The approach consists of **two** major phases:
  - **set-up phase** configuring the modeling approach for the organization, the modeling goals and context;
  - **execution phase** performing the actual modeling
- **Set-up** phase consists of **four** steps:
  - 1. State objectives and scope
  - 2. Select or develop a process modeling scheme
  - 3. Select (a set of) process modeling formalisms
  - 4. Select or tailor tools

Creating a Descriptive Process Model - Approach Overview

- **Execution** phase consists of **four** steps:
  - 5. Elicitation
  - 6. Create the process model
  - 7. Analyze the process model
  - 8. Analyze the process

Step 1: State Objectives and Scope

- This includes specifying
  - Who will use the process model
  - What these people expect of the model
  - Which processes should be covered
  - Which processes are explicitly excluded from the model
- With this information as a basis, it is possible to specify the scope more concretely.
- For example, knowing the general goals of the effort as well as people's concrete expectations makes it possible to determine which processes to model in which detail in order to better fulfill the stated goals and meet the stated expectations.

Step 2: Select or Develop a Process Modeling Schema

- This step consists of identifying the set of concepts that will be used to describe processes in the organization:
  - Activity
  - Work product
  - Role
  - Product flow between activities and work products, i.e., which products are inputs or outputs to which activities
  - Assignment of roles to activities, i.e., which roles are responsible for performing which activities

- A set of concepts such as this used to model processes is called a *process schema*

Step 2: Select or Develop a Process Modeling Schema

- Following requirements should be considered while choosing a schema:
  - The concepts must cover those aspects of a process that are relevant for the current modeling effort. E.X, if reducing time to market for new products is a stated modeling goal, modeling the amount of time required by different activities will be fundamental.
  - The concepts must allow for an adequate level of detail and formalism. E.X, a model that is intended for personnel training may be more detailed and informal than one that is intended to provide fine-grained support for process enactment.
  - The concepts must facilitate modeling. For this reason, it is important to test modeling concepts in advance before committing to them for a large modeling effort.
  - Modeling results must be accessible for their intended audience.

Step 3: Select (a Set of) Process Modeling Formalisms

- Process modeling requires a concrete notation in which these concepts can be expressed.
- The notation chosen for a particular modeling effort must be able to express the concepts identified in the previous step.
- Process notations can be characterized along a number of dimensions:
  - Graphical vs. textual
  - Formal vs. informal
  - Fine grained vs. coarse grained
  - Predetermined vs. extensible
- For instance, if the main modeling goal is to support automated enactment, a formal and fine-grained notation would be desirable. If the main purpose is to support personnel training, a more informal and coarser-grained notation would likely be enough.

Step 4: Select or Tailor Tools

- Tool support is usually necessary to effectively model complex processes.
- Tools are useful for:
  - Supporting the use of the chosen modeling notation. The more complex the chosen modeling notation, the higher the need for an editing tool that supports it explicitly.
  - Storing large models or model collections in a convenient and safe way.
  - Producing alternative representations of models. In many cases, different process stakeholders have different needs regarding the way a process model is presented. For instance, project managers may require an overview of the activities involved in a process, whereas the people working on specific activities will need to see the details of how such activities must be performed.

Step 5: Elicitation

- The elicitation step is intended to collect all information necessary for actually describing the target software process. Necessary information comprises:
  - *The process entities*. These include activities, roles, work products, tools, etc.
  - *Relationships between entities*. For example, information about which activities produce or consume which products, which tools are necessary for performing a task, or which tasks compose an activity is expressed through relationships.
  - *Behavioral properties of the process entities*, e.g., which conditions must hold in order for an activity to be started (preconditions), or which criteria a work product must fulfill in order to be accepted.

Step 5: Elicitation

- The information can be obtained from a number of sources:
  - Individual interviews with process performers
  - Group-oriented workshops with process performers
  - Direct observation of people at work
  - Analysis of preexisting work products
  - Analysis of other data left by the execution of a process, e.g., meeting minutes, electronic discussions, issue/bug reports, software version history, etc.

Step 6: Create the Process Model

- The information collected during the elicitation step can now be used to create a model using the chosen modeling notation.
- It is advisable to collect the information in the following general order:
  - Start by modeling products, because they are usually the easiest process entities to identify compared to activities, which are often only implicitly defined
  - Continue with the activities and the product flow
  - When activities are modeled, attach associated roles to the activities or the products
  - Model further entity types, such as resources, as needed
  - As a last step, model behavioral restrictions associated with entities, such as the preconditions that must be met before an activity can start

Step 7: Analyze the Process Model

- As process models become complex, there is an increasing risk of defects being inadvertently introduced into the model by the process engineers.
- The purpose of the process model analysis step is to detect such defects and correct them.
- Depending on the notation used, there are many possible types of defects that may arise in practice (as stated in the next slide):

Step 7: Analyze the Process Model

- *Dangling reference*: An entity references another, undefined entity
- *Inconsistent precondition*: The start conditions stated for an activity are contradictory and cannot be fulfilled in practice.
- *Inconsistent references*: The same entity is mentioned in the model using different, inconsistent names.
- *Orphan entity*: An entity is defined in the model, but is not referenced by any other entity in a meaningful way.
- *Dependency cycle*: An activity depends on a particular input product in order to start, but, according to the model, this input product cannot be produced unless the activity has already been finished.
- *Incomplete descriptions*: Important entity attributes, such as activity or product descriptions, are missing or empty in some or all of the process entities.

Step 8: Analyze the Process

- The final step of descriptive process modeling is perform a process analysis, i.e., to use the process model to track or analyze process performance, depending on the process modeling objectives stated in step 1.
- One possibility is to track the process by asking process performers to log, for example, the start and finish times for their activities.
- Another option is to make “snapshots” by asking people about their current activities at regular intervals.
- The resulting data can be used for qualitative or for quantitative analyses.

Step 8: Analyze the Process

- *Qualitative analyses* aim at identifying weaknesses of the processes.
  - e.g., when too many responsibilities are pinned to a single role, or when too many roles are assigned to a single person, or when feedback loops become so excessive that they consume more time than productive project work.

- *Quantitative analyses* aim at identifying correlations between process attributes.
  - e.g., when the number of requirements is more than 30% higher than average, the number of defects rises by 70%.

Step 8: Analyze the Process

- Both kinds of analysis results can then be used to:
  - Modify the process (in order to remove the weaknesses)
  - Modify the process model (in case it does not fit the process)
  - Influence project planning (e.g., allocate more resources to requirements reviews when the number of requirements is more than 30% higher than average).

Example: The 8-step approach is illustrated by a case study - DocVault.

- The Defect Management Process at DocVault
  - DocVault is a small software development company, with about 50 employees and 6 years in the market. DocVault's main product is an Internet-based document management system that is made available to clients through a subscription-based model (software as a service). The company already has more than 30 corporate customers and good prospects for acquiring several more in the upcoming months.
  - Since DocVaults's product is already quite large and complex, it is relatively common for clients to be affected by defects in the system that were not detected by the quality assurance procedures used by the company (mainly testing). Until now, the company has relied on email as the main mechanism for clients to report problems they might have found. This simple mechanism has served the company well so far, but with the number of customers quickly increasing, the need for a more elaborate and potentially automated solution is becoming urgent.
  - For this reason, DocVault wants to start by modeling its defect management process. This process involves not only the actual problem reporting by users, but also the classification of problem reports by company staff, the assignment of potential defect-fixing tasks to developers, and the follow-up of defect fixes until they are released to clients, among other tasks.

Case study - DocVault
Step 1: State Objectives and Scope

- The modeling effort at DocVault is restricted to the defect management process. This means that the scope is relatively narrow:
  - Modeling is limited to the defect management process. It includes taking problem reports from users, classifying them, assigning accepted reports to responsible people in the company, and following them up until fixes are released. Technical processes related to the actual fixing of defects, such as determining the causes of particular failures, designing and implementing code fixes, and testing fixed code, are excluded, however.
  - The main users of the description are the developers of the automated defect management system. Notice that these are likely to be not the same people developing DocVault's product, since DocVault will probably subcontract the defect management system to another company. DocVault's clients, as well as the software developers and quality management team at DocVault, are considered secondary users, as they may use the process description for their own purposes, such as guidance while actually handling problem reports.
  - The developers of the defect management system(primary users) expect a detailed description they can use as an initial set of requirements for implementing the process automation. The secondary users, on the other hand, expect a clear description that can be used for guiding internal users at DocVault.

Case study – DocVault

Step 2: Select or Develop a Process Modeling Schema

- Given the narrow scope of its modeling effort, DocVault can live with a small set of modeling concepts: task, artifact, and role. Also, modeling more specific dependencies among entities of these types is not necessary, because the involved work flows are relatively simple. For this reason, a produces and a consumes relationship, together with a relation to associate roles to tasks are enough for this modeling effort.

Case study – DocVault

Step 3: Select (a Set of) Process Modeling Formalisms

- Since DocVault's aim is to automate the defect management process, they require a process description that is essentially free of imprecision and ambiguity. For this reason, the process modeling team decides to define a formal, yet simple process notation. This textual notation allows for defining instances of the concepts mentioned in the last section. While defining a process, entities must be given unique identifiers that can later be used for referring to them from other elements, such as relations. Additionally, a formula notation can be used to express pre- and postconditions for tasks. These conditions can later be used by the programmers automating the process.

Case study – DocVault
Step 4: Select or Tailor Tools

- The relatively small size of DocVault's defect management process as well as the selected, text-based notation make it possible to conduct the modeling without using advanced modeling tools. For this reason, DocVault decided to use standard programming text editors for creating and maintaining the model.

Case study – DocVault
Step 5: Elicitation

- DocVault used three information sources for elicitation:
  - Interviews with members of the quality assurance team
  - Interviews with staff members of one large customer, who had reported several problems over the years
  - Archived e-mail messages from various problem reports

Case study – DocVault
Step 6: Create the Process Model

- The model creation work at DocVault was conducted by two process engineers, who prepared detailed step sequences in a regular programming text editor using the defined formal notation. In order to review the resulting model, a workshop was conducted where selected members of DocVault's Quality Assurance team worked together with the process engineers. The process engineers presented the model, while the QA members were encouraged to criticize it and make suggestions. Issues were written down during the workshop and corrected afterwards.

Case study – DocVault
Step 7: Analyze the Process Model

- Since DocVault's model ended up being relatively small, most checks were conducted by manually inspecting the model. However, the process engineering team also created a simple syntax checker to make sure that a set of minimal syntactic requirements was fulfilled by the text-based specifications.

Case study – DocVault
Step 8: Analyze the Process

- Although the defect management process at DocVault had been handled manually for the most part, it actually left traces in a number of data repositories. First, a good portion of the communication and software defects happening between customers, QA people, and developers were reported via email, and those messages were archived. And, second, changes made to the software because of defects found were always registered in the version management system, together with a log entry explaining the reason for the change.

- The process group then proceeded to isolate a number of defect-fixing cases and to identify email messages and entries in the version management system that were relevant to each case. By looking at this data, the process engineers were able to create detailed profiles for these cases, identifying the people involved, the activities that were conducted, and the time for each event.

- With these profiles at hand, the process group then proceeded to check their models. The basic question was to determine whether the process model could be instantiated for each one of the particular cases, and how difficult this would be. Together with the primary users (developers of the defect management system) and the secondary users of the model (developers of the defect management system and DocVault's clients, as well as the software developers and quality management team at DocVault), they decided that the model adequately describes the process and proceeded with the implementation of the automated defect management system.

Self-Test:

- Suppose now that the selected goal was “process guidance.” What would be the consequences for step 3 in this case?