8 Examples of TIMe

Introduction 2
Scenarios and Methodology 3
First introduction to the example 5
List of figures 6

Examples

Introduction

This theme covers the application of the methodology on a concrete example by means of a set of *scenarios*.

Currently only one scenario is available:

- initial development of a new product starting from scratch.

In future version of this book we plan to present scenarios for:

- creating a product family/ application framework based on some existing products (reuse)
- evolving the application framework to cover new properties.

All scenarios will be illustrated by the same example. In order to have a general introduction to this, have a look at First introduction to the example (p.8-5).

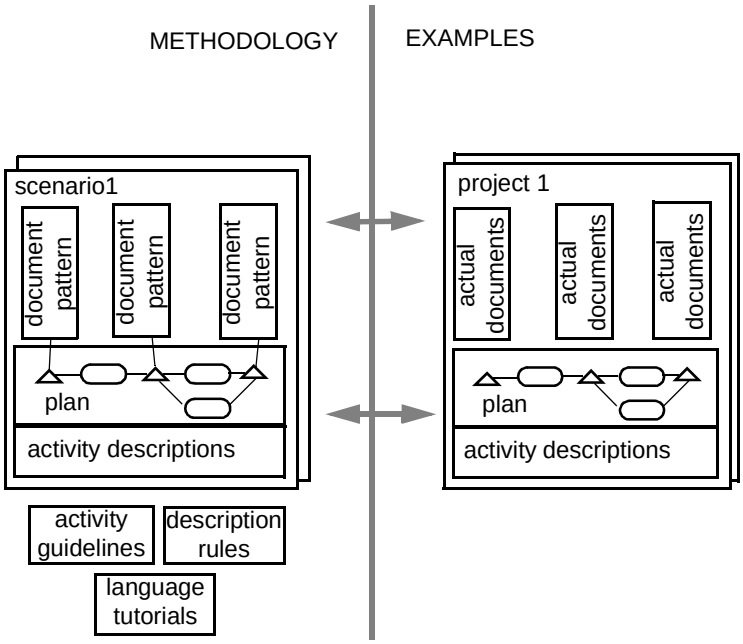
The relation to the general methodology is given in Scenarios and Methodology (p.8-3).

Scenarios and Methodology

The scenarios deal both with the overall *process* (providing the context) and the *specific activities* which must be performed in order to produce the various models and descriptions. Applications of the guidelines of the various activities will be demonstrated.

Figure 8-1: Scenarios and project examples

Open figure



In the Process theme a number of development processes are defined. The scenarios are examples of exactly these processes. Each scenario is a project with concrete descriptions and activities.

The material covered here is complementary to the Descriptions theme, which deals with the (static) categories of descriptions and documents covered by TIMe. Here the focus is on the (dynamic) activities that will produce (sequences of) the descriptions.

The Configuration Management theme describes how the very flexibility of software can lead to problems if you don't have a good overview of exactly which variants of which items you have, their different characteristics, the differences between them etc. It discussed various ways of expressing this information, to help maintain a clear picture of the items you have.

However, the Configuration Management theme addresses only the *static* part of the problem, i.e. representing what variants you actually have, now. It does not say anything about what activities you must carry out in order to *develop* new variants.

So: just as it is necessary to have orderly ways of representing what variants you have, it is important to have orderly procedures for how to *produce* the variants.

We propose an overall *software development process*. It shows the main activities that should take place as a product evolves, and is constructed in such a way as to support flexible adaptation, configuration and re-use. The process is described in prose and in diagrammatic form.

There are a few important things to note about this development process (or indeed any development process other than one tuned to a specific company and project):

- it represents the overall sequencing of activities that should take place; it does not tell you “what to do” to solve a specific task;
- it can be viewed as a kind of “road map” showing which activities/results lead to which other activities/results;
- it would certainly need customization before being adopted in a specific company or development project;
- the process encompasses the entire (hopefully long) lifetime of the product: it is unlikely that any one individual would ever navigate his/her way through the entire process.

So: the role of the development process is to provide an essential *context* for organising the day to day activities of systems engineers. Its impact on their day to day work is to show how completion of a specific activity relates to other activities and, ultimately, to an effective way of producing new system instances.

System engineers who really use a methodology want to do so on a more or less day to day basis; they want something that tells them *how to proceed* under various different circumstances. While the development process gives a context, it does not answer this more immediate need.

For this reason, we also include a number of scenarios illustrating how the process works in various situations that the software developing organization must tackle. Each scenario is described in terms of:

- *background:* the typical conditions that would hold beforehand.
- *task specification:* the objective that the company wants to achieve
- *where to start and where to go:* specific guidelines about how to proceed - related to the overall development process

Building on the analogy that the development process is like a road-map, the scenarios are akin to *directions*.

First introduction to the example

This is an informal introduction to the example used to illustrate the methodology.

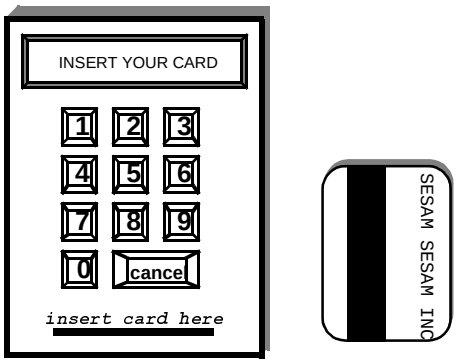
The purpose of the Access Control System is to control the access to some service (i.e. entrance through a door into a zone) open only to trusted people with known identity. Ideally, users should be recognised by the system and granted or denied access depending on some unique physical attribute.

Each card holds a unique Card-code that identifies the card. To grant access the system will read the Card-code and then check the corresponding access right. For additional authentication, the user may be asked to enter the secret personal number (PIN).

The card is a plastic card with a magnetic strip holding a card code and possibly an encrypted PIN code. The encrypted PIN will not be used in our system initially. At a later stage it may be used for local authentication in the Local Stations. The physical appearance of the panel and the card is shown in Figure 8-2 (p.8-5). Each panel represents an Access Point.

Figure 8-2: Panel and card of the access control system

Open figure



The main service demanded by the user is to gain access when the card is presented to the system. But access should only be given to authorised users. Hence, a user will be rejected if an attempt is made to enter at an access point where the user is not authorised to pass.

A typical access control system will consist of a number of access points and a central unit where validation is performed. Some access points are so-called blocking access points, that is access points that may be blocked by an operator, so that access is denied even with a valid card and code, until the access point is enabled again. Other access points may have the property that they log what is going on at the point.

List of figures

Scenarios and project examples 3
Panel and card of the access control system 5