7 Process models

Process models

Introduction

In TIMe a process is an ordered series of actions that produce change or development of descriptions and systems. The particular states of descriptions and systems that are produces are called milestones, and the periods of time when they are developed are called phases. Hence a process is an ordered series of actions and milestones related to phases.

Objectives

The purpose of process models in TIMe are:

- to illustrate how TIMe may work in some typical development cases;
- to present a project management view on activities and milestones;
- to be a framework for adaptation to company process standards.

What

This theme is about how to carry out property oriented development projects. In the theme Activities and descriptions we deal with the types of descriptions and activities of TIMe. Here we shall focus on how they can be practically ordered and managed in development projects.

In activity models there is no notion of time, milestones or project phases. In a development project however, activities occur in a specific order, and produce specific results. From a management point of view the ordering of activities and results in time is a central issue.

Figure 7-1: A process overview

Open figure

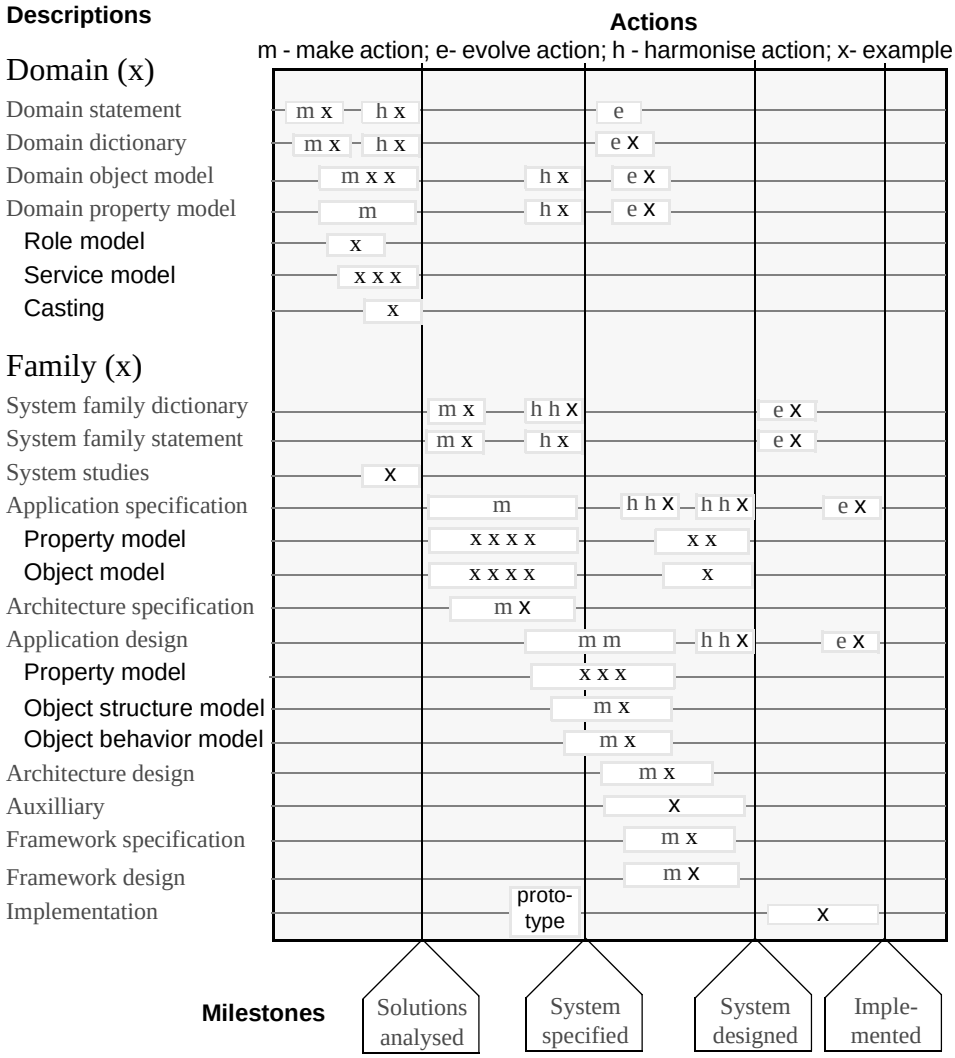

Figure 7-1 (p.7-3) illustrates how the various descriptions typically may be developed over time to produce milestone results. Each (white) bar represents an action on the description indicated in the left column. The letters inside the bars indicate the type of action:

- m - a make action;
- e - an evolve action;
- h - a harmonise action.

The general strategies and rules for these actions are described for each type of description in Activities and descriptions. If it is an “m” action, look under *making* that description, if it is an “e”, look under *evolving* and if it is a “h”, look under *harmonising*. For instance, the “m” in the leftmost top action stands for Making domain statement.

The “x” letters indicate that there is an example to be found in Example: Initial Development.

Process diagrams

While Figure 7-1 (p.7-3) has the advantage that it clearly shows how the descriptions evolve, it has the disadvantage that actions are not grouped in any way. We will use process diagrams as an alternative representation to show action groups represented as processes.

Process diagrams correspond to the management view needed to plan and follow up actual projects. They focus on processes and milestones, and allow processes to be decomposed until the basic actions represented in Figure 7-1 (p.7-3) are reached. (Plans will in addition consider resources and timing.) As an example, consider Figure 7-2 (p.7-5). It is a top level process diagram where we can see the processes, but loose the clear picture of how the descriptions evolve that we had in Figure 7-1 (p.7-3).

We use a notation for process diagrams where we show both processes and milestones. The processes and the basic actions will be instances of activities described in Activities and descriptions and follow the general strategies and rules given there.

Most companies will have some general process plans that specify major processes and milestones that every project should adhere to.

Process diagrams are like MSCs in the sense that they show typical orderings, but not all orderings that are possible. Hence, the process diagrams presented here may be seen as property models of the more general activity models presented in Activities and descriptions. This explains a major difficulty with process plans alone: they can only cover a limited set of cases. Therefore, to cover all the different cases that may be encountered in real life, we have to develop many different process plans.

The activity models, however, are far more general and cover most cases. They can be used with many other processes than those we present here. For instance the various strategies indicated in (Graham 89).

Process overview

In the following we will present some typical processes that together cover the lifespan of products. As illustrated in Figure 7-2 (p.7-5), two main cases are considered:

.

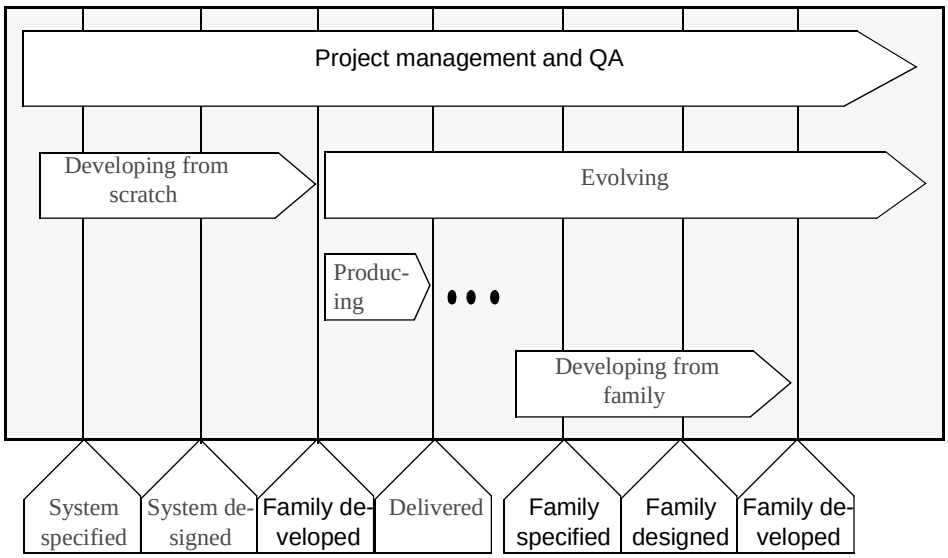

Figure 7-2: Processes described

- Developing from scratch (p.7-7) where we create an entirely new product in a new domain.
- Developing from family (p.7-16) where we create a new product based on existing families in an existing domain. This will be the normal case for most products.

In both cases incremental development is considered. We also consider:

- Producing (p.7-14). This task is in charge of specifying, engineering and producing the actual system instances to be delivered to customers.
- Evolving (p.7-15). This task takes care of the changes that are needed throughout the lifetime of the product. Errors reported from the marked is one important reason for change, but there are many other reasons, e.g.: new requirements, new technology, inconsistency between descriptions. Incremental introduction of new services are also covered by this process.
- Project management and QA. This task is concerned with project management issue such as project planning and follow up, milestones and milestone reviews. It is also concerned with the overall quality assurance tasks.

Note that every project is unique in one way or another. What information is available at start-up time varies a lot, the product complexity varies, the technological challenges varies, the risks and the schedules varies. In particular the ordering of tasks and the effort needed to perform them varies a lot. Therefore we will focus more on the milestones and the state of descriptions than on the tasks themselves.

In addition to the descriptions, most projects will develop a number of documents, such as:

- Project specifications that set forth requirements to the project itself, e.g. time constraints, management procedures, quality assurance plans, etc.
- Product specifications which contain the formal specifications supplemented with informal text and other matters.
- Feasibility study reports which are decision documents for the management.
- Installation manuals.
- User manuals.
- Test plans.

According to the ideas of Integrated Product development (IP), product development involves three main tasks which should run in parallel: product development, production development and market development. This means that the market, the product and the production is developed in parallel and that people from all three areas of concern cooperate all the way. We believe this to be a very sound principle that will help to reduce lead times and improve the quality.

Editorial

Documents will not be elaborated (at least not in this version) as they tend to be very product and company dependent.

In this version only development from scratch is elaborated, and even that only partially, just in order to give the reader an idea about what this theme can be like when fully developed.

Developing from scratch

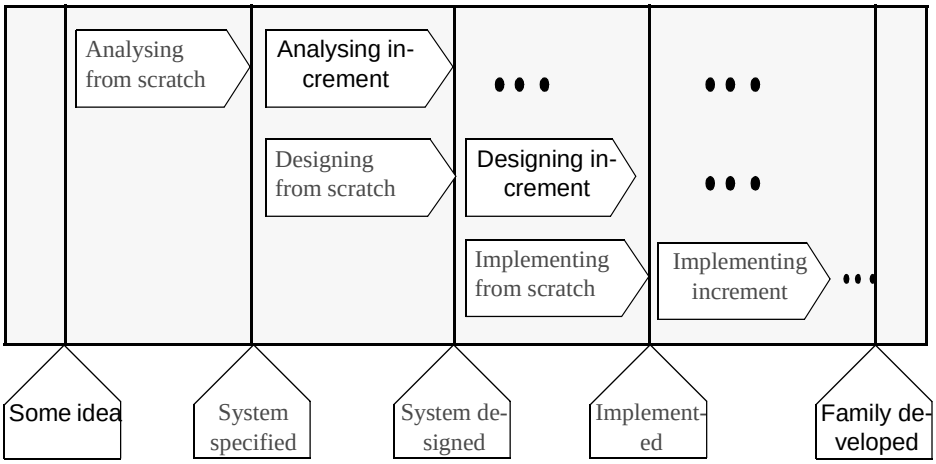

Figure 7-3: Developing from scratch

This process covers the development of a new system family from initial domain analysis to implementation. An incremental development strategy is assumed where the first increment involves the following sub-processes:

- Analysing from scratch (p.7-8) which ends in the milestone System specified (p.7-19). The goals here are to analyse the domain, to analyse possible system solutions and to specify the family in sufficient detail to start the first design increment. It contains an important internal decision point where a go/no-go decision is made for the remaining development.
- Designing from scratch (p.7-11) which ends in the milestone System designed (p.7-21). Here the goal is to develop design models for the new system family. The design models must be sufficiently detailed that implementation may start, and the remaining design increments can be performed with limited impact on the first increment.
- Implementing from scratch (p.7-13), which ends in the milestone Implemented (p.7-23) where the first implementation increment is finished.

After the first increment, the development continues until a milestone is reached where the family is well defined and production may start. After that, developing from scratch ends and Evolving (p.7-15) takes over.

It is assumed that the analysing increment actions also take care of harmonising the domain descriptions with the family descriptions.

Strategy

The ordering of these processes is not arbitrary. It is essential that analysis, designs and implementations follow each other in that order for each part of the product. But it is not necessary that every part of the product is developed at the same time. Therefore it is possible to start design before all parts are specified, and to start implementation before all parts are designed, as indicated in Figure 7-3 (p.7-7).

Each increment should follow the main sequence indicated in Figure 7-3 (p.7-7).

Analysing from scratch

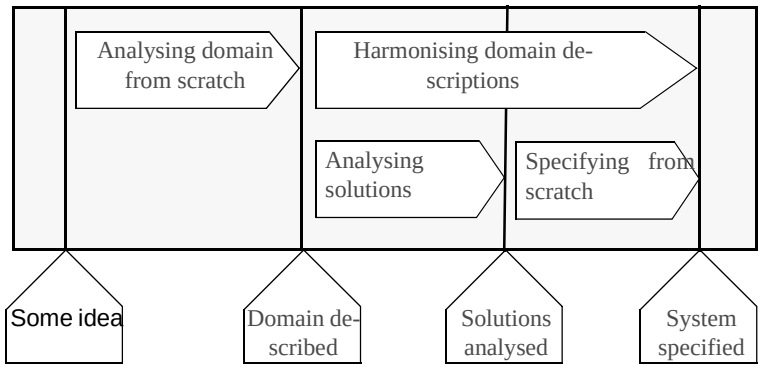

Figure 7-4: Analysing from scratch

Activity model

This process is an instance of the analysing activity and it follows the analysing from scratch strategy.

What

Since it is the first analysis increment, the results need not be complete, only be sufficiently complete to assess the feasibility and to start the first design increment.

The process contains an internal milestone Solutions analysed (p.7-18), which is used to decide whether one should continue the development or not. If it is decided to go ahead, a number of specification actions are performed that result in the milestone System specified (p.7-19). The goal is to develop requirement specifications for the first increment of a system family that will satisfy a range of needs in a market. The two most important outputs are specification of functional properties and non-functional properties. Once the first increment is specified, the design activity may start while the specification continues with further increments.

The following sub-processes are involved:

- Analysing domain from scratch (p.7-9) which results in Domain described (p.7-17). The main outcome is the first domain descriptions for a new domain.
- Analysing solutions, which develops system studies and ends in the milestone Solutions analysed (p.7-18). Based on this milestone a go/no-go decision is made for the continuation of the development. Hence this sub-process ends in a major control point. It is a first investigation into design possibilities of the new system, that explores the major solution alternatives and seeks to answer if a development project can be profitable or not. Prototyping may be necessary to determine the feasibility of an idea. Positioning properties (properties that gives a competitive advantage), and critical technical aspects must be in focus.
- Specifying from scratch (p.7-10), which develops the first specification increment called System specified (p.7-19).
- Harmonising domain descriptions which seeks to keep the domain descriptions aligned and consistent both internally and with the specification.

In successful companies, the project planning which is part of this process, is a more or less continuous process.

After the first analysis, described here, domain descriptions will exist that can be used directly to plan product evolution and new products in the same domain.

Analysing domain from scratch

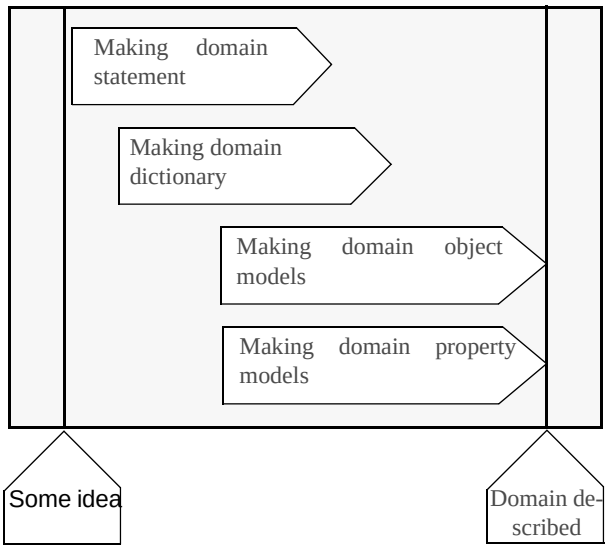

Figure 7-5: Analysing domain from scratch

Activity model

This process follows the Analysing Domain activity using the new domain strategy. A general description of the actions can be found there.

What

In the first domain analysis, described here, it is assumed that the domain is not described before. Therefore al the domain descriptions are developed for the first time here.

Specifying from scratch

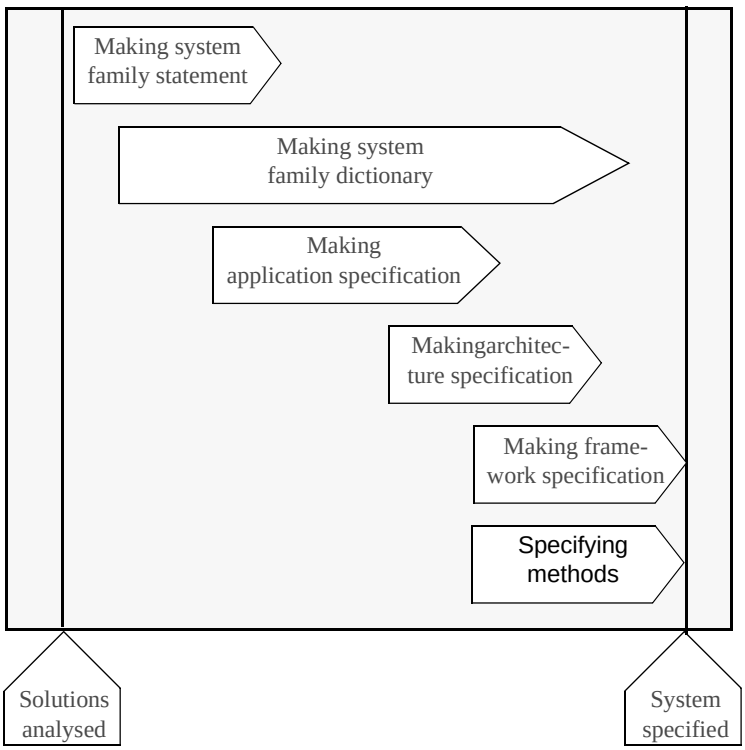

Figure 7-6: Specifying from scratch

Activity model

This process follows the Analysing requirements activity. The general strategies and rules can be found there.

It may well happen that specification of the framework and the methods are postponed until after the architecture has been designed.

Designing from scratch

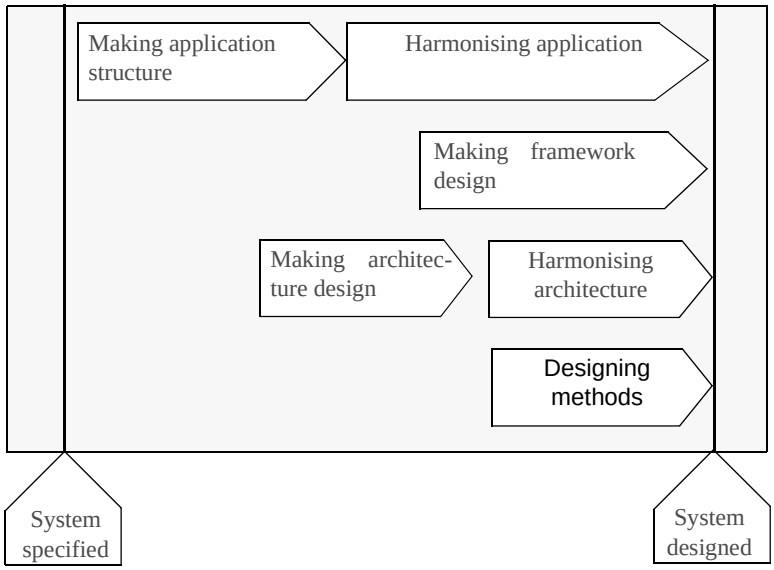

Figure 7-7: Designing from scratch

Activity models

This process is instance of the Designing activity, which explains the general approach.

What

The purpose of this process is to design the first increment of the system family. This includes application design, architecture design and framework design.

This is the main design task for a new system family. The main actions are:

1. Making application design which makes object models and property models for the (first increment of) application design. This defines the implementation independent service functionality of the system family. UML/OMT+- is used at the system level to give a unified view of the system and its component types. SDL is used to formally define behaviour and component types having SDL type of reactive behaviour. It shall be complete with respect to the services selected for the first increment, and provide a scheme for adding services in later increments.
2. Making architecture design seeks to design an implementation architecture that satisfies the non-functional properties. Hardware and software architectures will be defined to a level of detail from which implementation is well defined. The architecture shall separate between (implementation) support mechanisms, such as an operating systems, and applications. It should be a framework definition from which configuration and building of system instances shall be as easy as possible.
3. Making framework design takes the implementation dependent functionality into account, e.g. distribution support, error handling and configuration. It seeks to adopt a layered approach which separates between the application (defined in action 1 above) and the implementation dependent infrastructure. The infrastructure part shall be nearly complete, and the rules for mapping applications to the framework shall be well defined.

4. Designing methods is an action that results in well defined methods for evolution, code generation and system instantiation, see family auxiliary.

While Analysing from scratch (p.7-8) has focus on objects in the domain and environment of the system, this sub-process has focus on the objects in the system.

Implementing from scratch

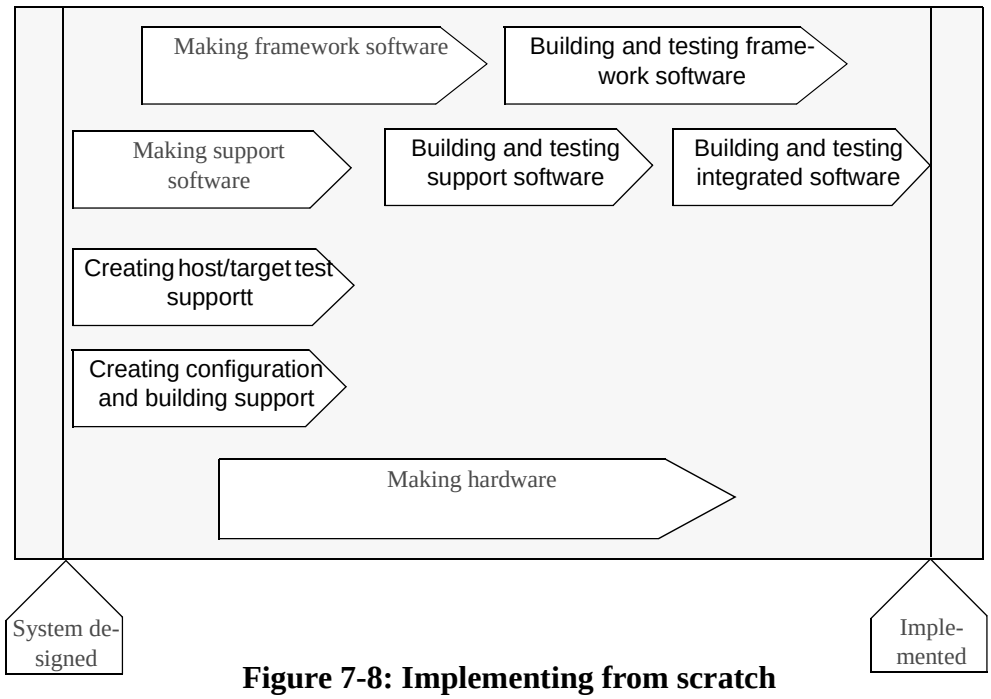

Figure 7-8: Implementing from scratch

What

The goal is to implement the first increment of the product family. This includes hardware, support software and application software as well as support for testing, configuration and building on host and target.

These actions are not elaborated further in this version, but Figure 7-8 (p.7-13) shows a possible process diagram.

Producing

What

This task is concerned with configuration and building system instances according to the needs of specific customers.

It is not elaborated in this version.

Some topics that might be covered are how to:

- specify (the configuration of) a system instance based on a system family;
- build the instance;
- test the instance.

Evolving

All successful products undergo evolution. In fact, the more successful they are, the stronger the push for new features seems to be. The integrated methodology, TIMe, has flexibility and modularity as important goals, and encourages planned evolution.

The notion of system families is introduced to help in that respect, and so are the system reference models. Service modelling using roles and close ties between roles and application objects also helps.

If one want to achieve service flexibility and to be able to introduce new services incrementally, at high speed, with little impact on existing services, then some careful design is needed. If one succeeds, it will be possible to introduce and change services mainly by working with property models on the application level.

Under the various evolving activities described in Activities and descriptions some guidelines are given.

Developing from family

This should be the normal case of product development. After using TIMe for some time, the company have some system families defined and use them as the basis for new developments. In that way new developments come to resemble evolution.

The important difference compared to Developing from scratch (p.7-7), is that existing families are used extensively. Reuse becomes the rule rather than the exception.

Domain described

In this milestone, domain descriptions have been made for the first time

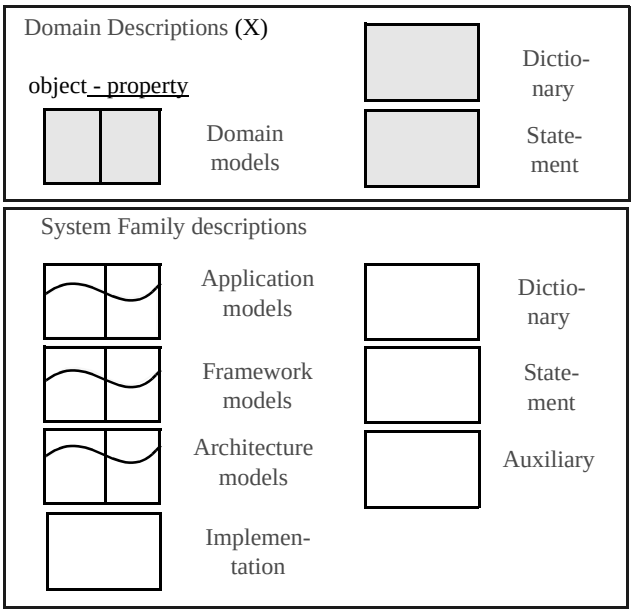

Figure 7-9: Milestone: domain described

.The domain descriptions shall be sufficiently complete that one can start analysing solutions:

Domain descriptions

(see example: domain description of access control)

Domain dictionary The dictionary shall cover every term used in the domain statement and the domain models.

Domain statement The domain statement shall be nearly complete and cover every entity and service that may be supported by the forthcoming system family.

Domain object model The domain object model shall define the overall domain structure and also contain a (component) model for (most of) the object types identified.

Domain property model Every domain service shall be defined using role structures, MSC and text.

Solutions analysed

In this milestone alternative solutions has be investigated. As a result there will be one or more partial family descriptions, and there will be an assessment of technical and economical feasibility.

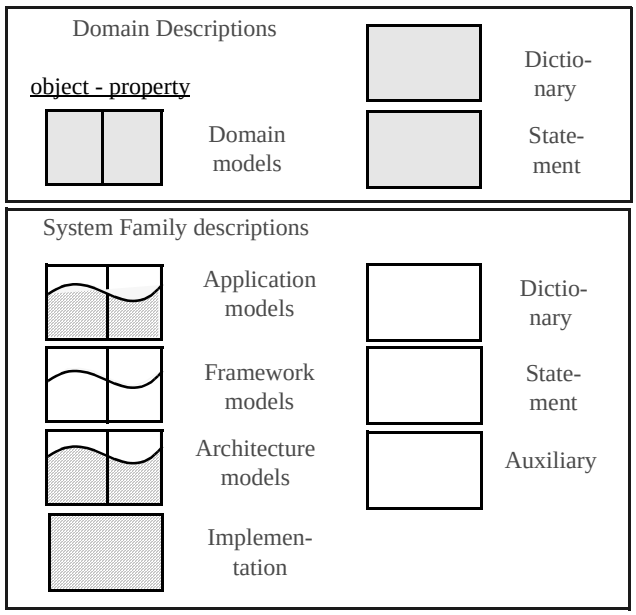

Figure 7-10: Milestone: domain described

The partial family descriptions will not necessarily be maintained. They will rather be used as inspirations for the next step.

System specified

At this milestone the domain descriptions are nearly complete, the specification is finished for the first increment, and design may start.

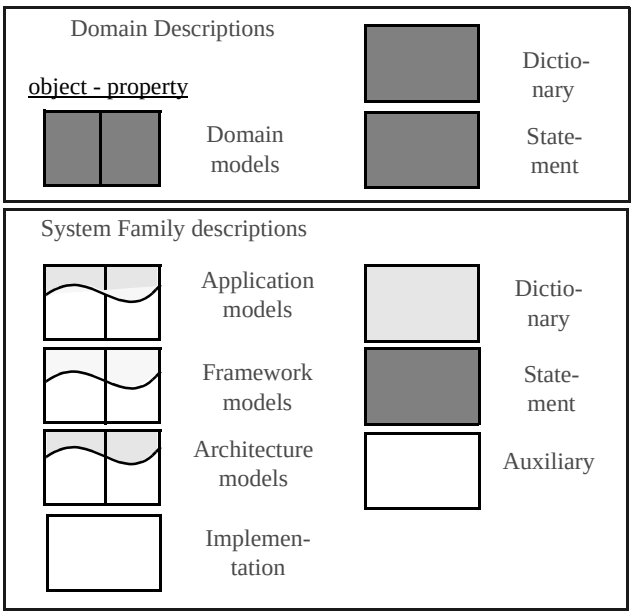

Figure 7-11: Milestone: system specified

It is recommended to specify properties as completely as possible in this first increment. Why? Because properties have impact on design. Thus, when new properties are added later on there is a danger that redesign will be required.

Design object models are just outlines, and the property models focus on high level description of essential properties and positioning properties considered as success criteria for the system family.

Domain descriptions

At this stage it is recommended that the domain descriptions are nearly complete. What can be missing is possible feedback from design activities.

Checklist: Is every stake holder described?

Is every task described?

Are the models in harmony (dictionary, problem statement, object models and property models)?

Family

The design space shall still be as open as possible, while the specifications shall be as closed as possible.

Dictionary The dictionary defines every system specific term used in the family statement and the specification models.

Statement The statement is nearly complete.

Application Both the domain given and the system given environment are completely described as well as the domain given services and some of the system given services. Interface given parts are specified to the extent they can be. However, as some interface specific issues may depend on architectural design, they may be open at this stage.

Variability issues relating to the application level shall be specified.

A design outline will identify domain given objects and possibly some interface given objects.

Framework Little is normally said about the framework except high level properties. Interface properties may be stated if required. Goals for the framework, i.e. flexibility shall be stated if relevant.

Architecture The architecture context shall be described in object models and the non-functional properties shall be specified (e.g. capacity, physical size, technology, power consumption). Variability in the physical environment shall be specified, as any other requirement to architectural variability.

Checklist Are every positioning property defined?

Is every duty property described?

Are the specifications harmonised with the domain?

Implementation Normally no implementation will exist, unless a prototype has been developed to test the feasibility of some critical parts.

System designed

At this milestone a first increment design is ready

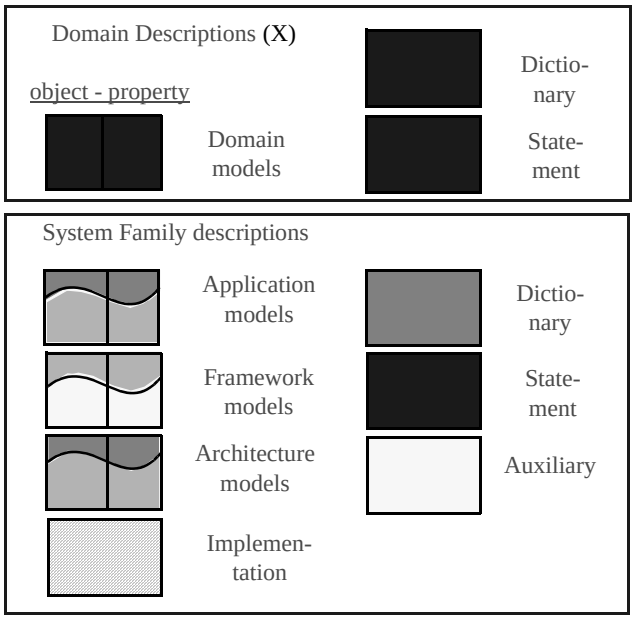

Figure 7-12: Milestone: system designed

Domain

At this stage the domain descriptions are complete.

Family

Family statement The family statement is complete.

Dictionary The family dictionary is nearly complete in the sense that all terms used so far are defined.

Application The application services are complete in the following sense:

- Services for the first increment are completely defined in object and property models.
- Considerations has been taken for future increments and how they shall be added. Design outlines has been made for critical or difficult services.
- Property models for future increments are as complete as possible and at least sketched.

The application object models are complete with respect to the first increment, and a method for incremental evolution is defined.

Framework	The framework specification is complete, and the design is complete for the first increment. However it may happen that the framework design is postponed to a later increment.
Architecture	The architecture design is complete for the first increment.
Implementations	Some prototype implementations may exist at this milestone.

Implemented

In this milestone the first implementation increment is finished, and several specification and design increments may have been performed since the previous milestone

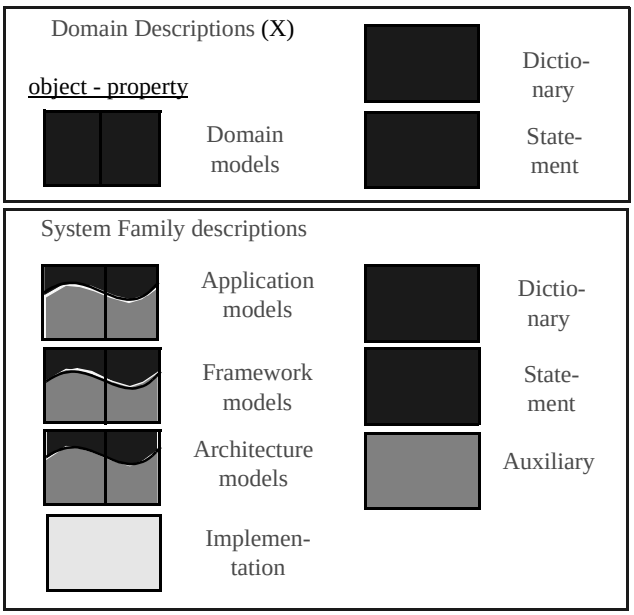

Figure 7-13: Milestone: system implemented

Domain

As complete as can be.

Family

Application The application design is at an advanced stage of development. What remains are possible increments that have not been performed as yet.

Auxiliary The method for incremental application evolution is now well defined, and so is the method for (automatic) code generation. The method for system instantiation is outlined.

Framework Sufficiently complete to support the early application increments that are implemented in this milestone.

Architecture Sufficiently complete for this implementation increment.

Implementation The first increment is made and tested. This means that hardware is finished, support software is finished and framework implementations are finished for that increment. The first implementation increment will often aim to get the platform in place, and only implement some limited application functionality.

Delivered

In this milestone, a concrete system has been produced and delivered.

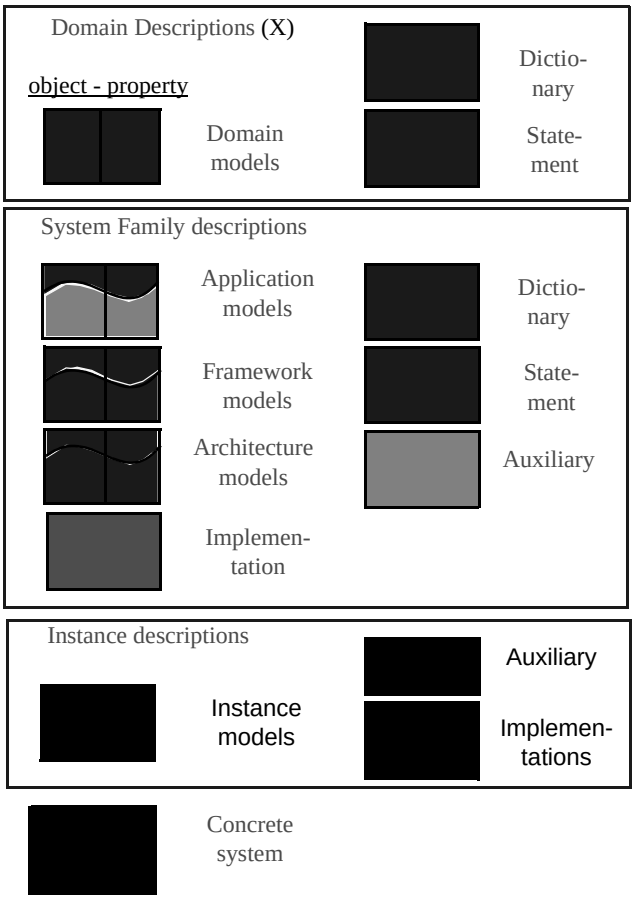

Figure 7-14: Milestone: system delivered

The delivered instance is complete as agreed with the customer. Family descriptions have reached a stage where the family is defined, and the future development is considered as evolution.

List of figures

List of definitions

Process

According to Collins (1986) a process is:

1. a series of actions which produce a change or development;
2. a method of doing or producing something.

In TIMe a process is an ordered series of actions that produce change or development of descriptions and systems. The particular states of descriptions and systems that are produces are called milestones, and the periods of time when they are developed are called phases. Hence a process is an ordered series of actions and milestones related to phases.

Action

According to Collins (1986) an action is:

- something done, such as an act or deed.

In TIMe an action is seen as an occurrence of an activity during some development process. It takes a specific state of input (a milestone) and produces a specific state of output (another milestone). The general rules and guidelines that should be followed during an action are described for the activity it is an occurrance of.