Universität Stuttgart
Fakultät Informatik, Elektrotechnik und Informationstechnik
Institut für Visualisierung und Interaktive System (VIS)
Universitätsstraße 38
70569 Stuttgart

Fraunhofer IAO
Competence Center Human-Computer Interaction
Nobelstraße 12
70569 Stuttgart

Diplomarbeit Nr. 2370

Design, Implementation and Test of a Graphical Information and Interaction Interface for Process Control Systems

Xiaojun Li

Studiengang:	Informatik
Prüfer:	Prof. Dr. Thomas Ertl
Betreuer:	Dipl.-Inf. Thomas Schlegel (Fraunhofer IAO)
	Dipl.-Inf. Martin Rotard (Universität Stuttgart)
Beginn am:	11.07.2005
Beendet am:	10.01.2006
CR-Nummer:	D2.1, D2.2, H5.2, I.3.6, I.7.2

Erklärung

Hiermit versichere ich, diese Arbeit selbstständig verfasst und nur die angegebenen Quellen benutzt zu haben.

Unterschrift:

Stuttgart, 10.01.2006

Acknowledgement

I would like to thank Prof. Dr. Thomas Ertl (VIS) and my supervisor Dipl.-Inf. Thomas Schlegel (Frauenhofer IAO) for giving me the chance to do the work and learn many things from this.

I would like to thank my supervisor Dipl.-Inf. Martin Rotard for his support and advices during the work.

I would like to especially express my thanks to Dipl.-Inf. Thomas Schlegel for his well-founded and constructive discussions throughout the progress and final outcome of this manuscript sparing his valuable time for me.

Finally, I would like to thank all my colleagues of Fraunhofer IAO, Competence Center Human-Computer Interaction, for their support.

Contents

Deutsche Kurzfassung

Table of Figures

Deutsche Kurzfassung

Heutzutage werden die automatisierten technischen Produktionsprozesse von menschlichen Controllern überwacht und beobachtet. Eine wesentliche Aufgabe der Prozessleittechnik besteht im Realisieren der Mensch-Maschine Schnittstelle und Verwalten der Information von Maschinen, Prozessen und Produkten. Alle Informationen, die mittels Messtechnik erfassbar sind, können an den zentralen Server in Form eines aktuellen Prozess- und Maschinenzustands übermittelt werden. Mit Hilfe dieser Informationen können die Controller den Produktionsprozess durch direkte Eingabe von Daten and Befehlen steuern. Zur Qualitätskontrolle interessieren sich die Controller nicht nur die aktuellen Prozesswerte sondern auch für statistische Daten der Prozesse, die den Trend eines Produktionsprozesses zeigen und so für die Controller und Maschinenbediener alle möglicherweise eintreffenden Situationen besser vorhersehbar und beherrschbar machen.

Um diese Information nutzbar zu machen, wird eine Benutzungsschnittstelle benötigt, die alle Informationen sinnvoll integrieren und den Menschen mit allen Informationen gezielt und kontextspezifisch versorgen kann. Die Voraussetzungen dafür, eine solche Benutzungsschnittstelle für die Überwachung und Steuerung der Produktionsprozesse zu entwickeln, sind entsprechende Anwendungsmodelle für die Prozessleitfunktionen, ehre Klassifikation und Typisierung der Maschinendaten und Prozesswerte, Sowie die kontext-/situationsabhängige Repräsentation von Daten.

In dieser Arbeit wird ein Prototype einer modellbasierten Benutzungsschnittstelle für die Prozess-Überwachung und -Steuerung aufgebaut, basierend auf Anwendungsmodellen verschiedener Problem Domänen aus dem Bereich Prozesskontrolle und Produktionssteuerung. Die Anwendungsmodelle werden auf der Basis von Erkenntnissen aus der Prozessleittechnik aufgebaut. Für die weitere Arbeit, spielen die so aufgebauten Anwendungsmodelle eine wichtige Rolle als Informationsinfrastruktur des Produktionsprozesses.

Die Funktionen der Benutzungsschnittstelle bestehen aus zwei Teilen, eine ist Datenakquisition und die andere Datenrepräsentation. Um die Informationen von Prozessen zu visualisieren werden die in Anwendungsmodellen beschriebenen Daten ausgehend von ihren Datentypen in passende Präsentationsobjekte transformiert.

Aufgrund der zunehmenden Zahl sehr unterschiedlicher Systeme im Produktionsbereich, ist Plattformunabhängigkeit und Skalierbarkeit Idee wichtige Anforderung an ein modernes Prozessleitsystem. Daher findet die zunehmend gefragte vektorielle, SVG Technik, zur Implementierung der Präsentationsobjekte Verwendung.

Diese Arbeit bildet damit von Vorgehensweisen und Technologien der Modellierung von Anwendungsfällen für Fertigungssysteme bis hin zum Einsatz der Modelle zur Realisierung der Benutzungsschnittstelle für ein Prozessleitsystem alle Stufen der UI-Schnittstellenimplementierung für die Steuerungs- und Kontrollsoftware eines modernen Fertigungssystems.

Chapter 1

Introduction

1.1 Goals and Motivation

Technical production processes are initialized by the human and accordingly must be monitored and controlled by the human. For this goal the modern technologies, computer systems and information systems should cooperate in harmony. The production process system should hold the ability to build communication between central server and all information which is acquirable by the measurement technique. It is also necessary that the human controller is actively informed of the current process and machine state information in order to react on the remote processes by inputting data or giving commands using central computer or portable devices.

For a secure and economic production process only currently measured process values are not enough. For example, for a safe production, maximum and minimum values of machine's running state must be considered. For quality control the statistical process values assist the human controller to follow the trend of processing. Furthermore, with the knowledge of process information and statistical process values the machine operators can better handle the machines in all possible situations.

In order to achieve the above described goals, besides technical production process system an additional information system should be built which provides for communication between human controller and process information. Sometimes the usage of information by the human is difficult because he must control multiple function systems and computers at the same time. The method to resolve such a problem is to develop a human-machine interface which can integrate all helpful process information and allow to an efficient communication between humans and production systems.

In the early phase of developing a human-machine interface for a production process system the main focus consists of capturing requirements for process control engineering. The requirements of process control engineering are based on engineering functions and technologies for process control systems. The basis of developing human-machine interfaces is to build various task models by means of task analysis and means analysis.

In process control system information plays the most important role throughout the whole production process. In the final analysis, all actions are carried out by data exchange. As a consequence, task analysis of process control engineering can be integrated into the information-oriented infrastructure. In the last years, the most popular and flexible format of data exchange and representation is XML which makes it possible to exchange data between different machines and platforms. Due to the advantage of XML the task information modelled in XML format can be transformed into a user interface with different interpreter languages or generators [Schlegel 2003].

1.2 Structure of This Work

The goal of this work is to analyze the requirements of data information in a production process system using case studies and to model the use cases for implementing a human-machine interface for process control systems. To implement such a model-base user interface, at the early phase, task models of process control system are built as the kern of interface software, afterwards presentation models are built for graphical representation of data information to end users.

At first, in chapter 2 the fundamentals of technologies are introduced. The approaches of models for task analysis and software architecture are studied as the basis of building models for the human-machine interface. EU project INMAS addresses the problems of today's manufacturing by developing intelligent networked manufacturing system. The human-machine interface for production process control is a part of the project and will be introduced in the chapter 3. Furthermore, in chapter 3 the tasks and proceeding of this work are described.

The core of human-machine interface is task model. The primary work for modelling tasks is to determine needs of the human-machine interface for process control system which is studied in chapter 4. Needs of such system are analyzed with use cases in different problem domains. Beginning with monitor and control system, it describes which actual process values should be displayed for the control personnel. Afterwards, statistical process control is analyzed which focuses on the trend of processing. Besides process values the overall phase model describes the process elements of the whole

production process and the connections between the components that will be used in the future work for modelling navigation system of user interface.

After needs analysis in chapter 5 the task models of human-machine interface for process control system are built based on the use cases analyzed in chapter 4. At first, the models are built for the individual entities such as product and machine, and then with the relationships between the entities the model for entire production process is built step by step.

In chapter 6 a function model is employed for transformation of data information to graphical representation. The function model separates the function of data into two blocks, the one is acquisition and the other is representation. The two blocks are communicated by the interactive elements. Additionally, in this chapter the presentation models and navigation models of human-machine interface are depicted.

To implement human-machine interface for process control system based on previously built task models and presentation models, the SVG (Scalable Vector Graphic) technique is adopted to represent the information and to realize interactions with JavaScript. The choice of SVG for implementing user interface has the advantages that the modern SVG technologies are capable of describing effective graphical representation and still under active developments.

At the end of the work, the approach of modelling human-machine interface in this work is evaluated in chapter 8. The conclusion of this work and discussion of the future works conclude the thesis.

Chapter 2

Fundamentals

2.1 Principles of User Interface Design

User Interface is the design of the interaction between user and computer. The user communicates to the computer via commands menus and other graphical elements on the user interface. User interface is also the connection and interaction between hardware, software and users. Graphical User Interfaces have been a revolution in the user interface domain that enables non-expert users to operate computers without obstacles. With graphical user interface the user can interact with a computer through a metaphor of direct manipulation of graphical images and widgets in addition to texts.

To design a good user interface several important principles should be considered. In order to enable the software developers to design user interface based on the principles. There are various standards which support the software ergonomic and layout principles. In standards ISO 9241 and ISO 14915 the principles for user interface design are described in detail. Many expert books summarize also the most important principles for user interface design such as [Shneiderman 1998]. The following important principles for user interface design are summarized from the standards ISO 9212, ISO 14915 and [Shneiderman 1998]:

1. User focus:
Before the developer designs the user interface a question must be answered at first, that is, for whom the user interface is designed. The outline of the possible users should be sketched, for example:
- What are the user's needs?
- What are the user's goals?
- What are the user's average skills and experience?

2. Clear Structure
Clear and consistent mental models are the prerequisite of the user interface design. The related things should be summarized together and the unrelated things should be separated.

3. Consistency
This rule is the most sensitive rule for designing a harmonious user interface. There are many forms of consistency. For example, the sequences of actions should be consistent in similar situations; in prompts, menus and help screens the rule of consistency should also be followed.

4. Simplicity
The tasks in a user interface should be simple and easy to be understood. The communication between users and computer should be clear and simple in the user's own language.

5. Visibility
The design should keep all needed options and materials visible for a task. A good design doesn't trouble the users with extraneous and redundant information.

6. Feedback
The users should always be informed about actions or interpretations, changes of state or condition and errors or exceptions. The feedback should be transmitted quickly and clearly to users.

7. Tolerance
The user interface should be flexible and tolerant, that is, the actions of users are reversible. To reduce the cost of mistakes and misuses the user should be allowed to change the false operations with undoing and redoing. The reversibility may be possible for a single action, a data-entry task or a group of actions such as entry an address block.

8. Error prevention and simple error handling
As much as possible, the user cannot make a serious error, for example, alphabetic characters are not allowed in numeric entry field. In case the user makes an error, the system should have the ability to detect the error and to offer simple instructions for recovery.

There are some tips for designing a good user interface described in [Ambler 2005] and [Shneiderman 1998]:

1. Design according to standards and guidelines
2. Easy to be used for non-expert
3. Quick and easy navigation within a screen
4. Easily switch between screens

5. Effective and sufficient messages
6. Suitable widget for right understanding
7. Good visual effects
8. Effectively group the elements

2.2 Models for Human-Machine Interface

2.2.1 Use Case Model

2.2.1.1 Definition

For long time software developers have captured requirements of a system and understood how a system works without formal methods. The requirements capture has not been documented in an effective manner. Use Cases are technique that formalizes the capturing and analyzing of system requirements. Use Cases help developer design and model systems as efficiently as possible.

“A Use Case delivers a measurable result of value to a particular Actor”
Pete McBreen

In [Jacobson 1992] Use Cases was defined as object-oriented. Actually the mechanism of Use Cases is independent of object orientation. Capturing the scenarios of systems enables the users and the developers to review the system [McBreen 1998]. The functional requirements are used to evaluate the initial design and final implementation of the system.

2.2.1.2 Use Case Diagram

Use cases play a role as interaction between user and the system to be designed. “Actor” is used to describe a person or system that will achieve a goal, such as “worker”. The goal is an action such as “turn off the lamp”.

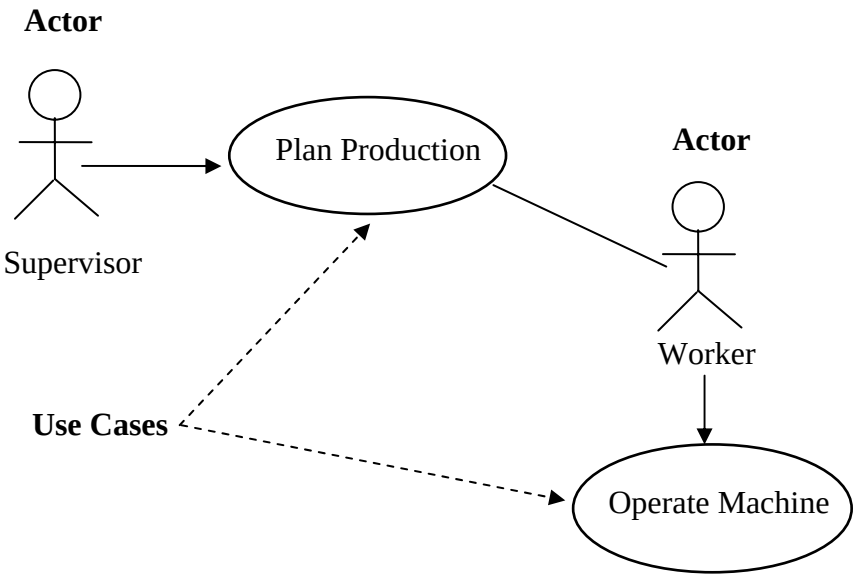


Figure 2.1: Use Cases Diagram

The Unified Modeling Language (UML) specified a graphical notation for depicting Use Cases [UML 2005], in use cases diagram shown in **Figure 2.1** a stick-man symbol denotes actor and ellipses denote the Use Cases. From the diagram the relationship of Use Cases and context of system are clearly to be observed.
The use cases can be detailed by one or more scenarios and each scenario belongs to one case [Oesterreich 1997]. The scenarios describe concretely the interactions between users and applications that are necessary for implementing the use cases in special ways. Because the use cases models support the development of software each project with user interface must include the analyses of use cases.

2.2.2 Task Models

2.2.2.1 Task Model

Task model is an important part of modeling user interface system that is employed to analyze and describe the user operations in branch of software ergonomics. The task model attempts to describe how the users resolve tasks and problems. Each step of work can come from other steps and be associated with other operations. At all levels the user can freely choose operations and steps from a set.

In [Norman 1988] seven stages of actions for modelling tasks of human-computer interaction were described:

1. Forming the goal
2. Forming the intention
3. Specifying the action
4. Executing the action
5. Perceiving the system state
6. Interpreting the system state
7. Evaluating the outcome

2.2.2.2 GOMS Model

Between 1980 and 1983 Card, Moran and Newell proposed the GOMS model for modelling tasks. The GOMS model, **G**oals, **O**perators, **M**ethods and **S**election rules is the most frequently used model that attains the goal of task model. Goals are what the user can achieve by using the software. Operators are user actions that lead to reach to the goals. Methods are description of the structure of models that can consist of more atom steps so that the goals can be reached step by step. With selection rules the developer can define the alternative methods for a goal. It is possible to choose the alternative methods under given conditions. **Figure 2.2** shows a simple example of GOMS for setting the time and alarm of a clock.

Goal: Set time / alarm
Press and hold button "Time" / "Alarm"
Goal: Set hour
... if not already right
Goal: Set minute
... if not already right
Release button
Goal: Set hour / minute
Press button "Hour" / "Minute" briefly
... until correct number appears

Figure 2.2: A simple Example of GOMS Analysis

For a process control system the GOMS model is not only developed for analyzing generated user interface but also adopted for a long time in the phase of process planning. Comparing to the mechanism of Use Cases both models has great equivalence.

2.2.3 Entity Relationship Model

2.2.3.1 Definition

Entity-Relationship (ER) Model was developed in 1976 by Chen, it is a high-level data model which describes conceptual data models and helps the designer understand and specify the components of the database and the relationship among the components.

Entity: An entity is a real-world item such as a particular student, a department or experiment. The possible values for an entity are defined as entity types. Each entity has also attributes or properties which describe the entity. For example, the name, identification number and grade of a student are the attributes of a student entity. The set of all possible values of an attribute is the attribute domain.

Relationship: Relationship is the associations among entity types. For example a student entity type is related to the department entity type because each student is a number of a department. A relationship is an ordered pair consisting of related entities. A relationship can also contain attributes that describes the details of relationship, but there are not connections between relationship attributes and entities.

2.2.3.2 Entity-Relationship Diagram

ER model provides a graphical notation for representing the data models in entity-relationship diagram.

Entity and attributes: In an ER model an entity is presented by a box with name of the entity. An attribute of the entity appears in an oval that has a line to the corresponding entity box. In **Figure 2.3** the student is modelled as an entity with an attribute “Name”.

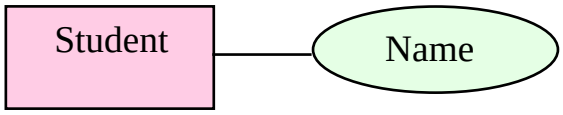


Figure 2.3: An Entity Model with Attribute

Relationship and relationship attribute: A diamond shape is used to illustrate the relationship between entities in an ER model. The relationship is read conventionally from left to right or from top to bottom. The arrangement of the diagram depends on the affiliation of the entities. **Figure 2.4** (a) illustrates the examples of entities and the relationship between them, **Figure 2.4** (b) shows the model of relationship attributes.

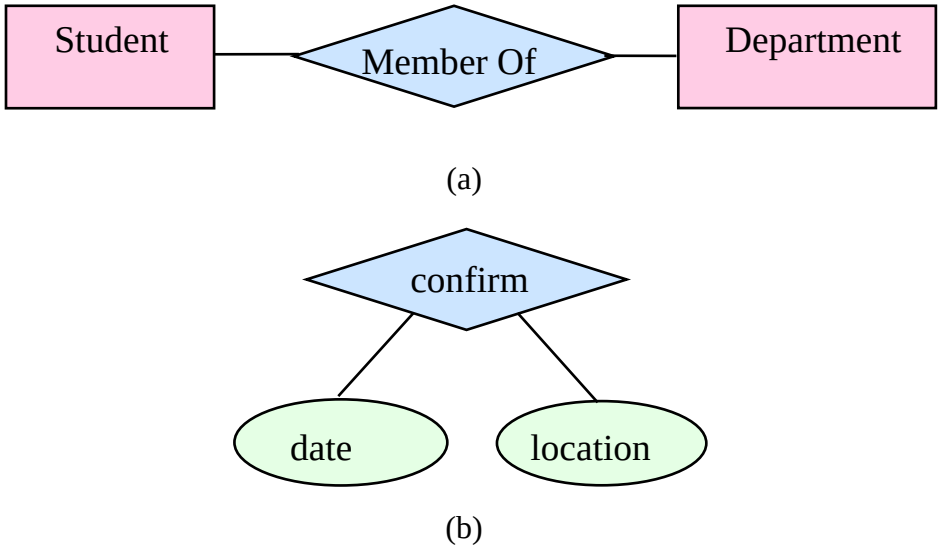


Figure 2.4: (a) Entities and Relationship between them; (b) Relationship Attributes

2.2.4 Software Architecture Models for Human-Machine Interface

Software architecture model which is used for modelling human-machine interface describes how the software is structured and which approaches support the interactions between human and machines. In following sections three familiar software models for building human-machine interface are introduced.

2.2.4.1 Seeheim Model

Seeheim-Model is structured for user interface management systems. Seeheim-Model is divided into three parts:

1. Application interface model which represents application program from aspect of users. In this layer the manipulable objects and communication functions are built with corresponding restrictions.

2. Dialog control model which defines the dialog structure between the user and application. In this dialog control model the sequence of input messages of presentation components are mapped to the sequence of commands for application.

3. Model of presentation components for extern display of user interface at end devices such as monitor and input devices.

The communication between individual components occurs from tokens which are dependent on the information structures. This model is used as reference for

programming user interfaces. The Components of Seeheim Model and communications between them are described in **Figure 2.5**.

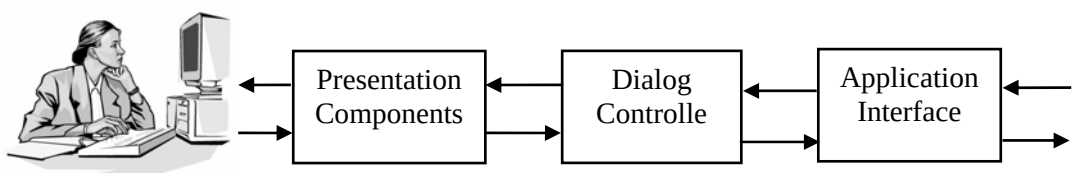


Figure 2.5: Seeheim-Model

2.2.4.2 MVC

The MVC (**M**odel-**V**iew-**C**ontroller) model is a very useful architectural software design pattern because it separates the codes into distinct components according to different applications so that modifications to one component can lead to minimal impact to the others. MVC-Model is successfully and widely used in development of user interface systems. The basic idea of MVC is to separate an application into three separate components: data model, view (presentation) and control logic as shown in **Figure 2.6**. The View represents visually the information and Model is use cases of application. The Controller plays the role as communication between View and Model.

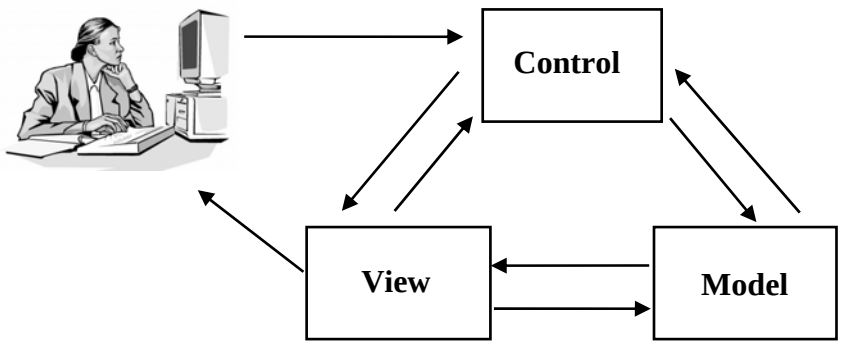


Figure 2.6: MVC-Model

The MVC-Model benefits implementing user interface applications because it enables programmer to give multiple views for the same model. The other advantage of using MVC-Model is code separation. For example, the developer can flexibly change the view of a model by just modifying the codes of presentation objects without to change

the codes of functions. With MVC-Model the efficiency of development is greatly economized because it is possible to avoid duplicable works.

2.3 XML Data Structure

2.3.1 XML

XML is a W3C recommendation that stands for Extensible Markup Language. The most important benefit of XML is to describe data and focus on the structure and types of data. Using XML Schema or Document Type Definition (DTD) the users can design self-descriptive XML documents. Because of the advantages XML is widely used in branch of data management.

With XML, data can be exchanged between incompatible systems. Because many computer systems and databases contain data in incompatible formats, the developer should be troubled by exchanging data between such incompatible systems. If the database is converted to XML the complexity of development can be reduced.

```
<Static_Info_Process>
    <NumberWorkers>50</NumberWorkers>
    <NameSupervisor>Meister Li</NameSupervisor>
    <ModelEndProduct>PASSAT 123</ModelEndProduct>
    <InfoCustomer>VW</InfoCustomer>
    <Task_This>Door</Task_This>
</Static_Info_Process>
```

Figure 2.7: XML Syntax

Another Advantage of XML is that the XML tags are not predefined such as HTML, the user is allowed to freely define own tags and own document structure.

2.3.2 XML Schema

XML Schema was originally proposed by Microsoft, in May 2001 W3C admitted XML Schema as an official W3C recommendation. XML Schema describes the structure of an XML document and restricts the types of data. The code in **Figure 2.8** is a simple example of XML Schema document.

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com"
elementFormDefault="qualified">
<xs:element name="note">
   <xs:complexType>
     <xs:sequence>
           <xs:element name="to" type="xs:string"/>
           <xs:element name="from" type="xs:string"/>
           <xs:element name="heading" type="xs:string"/>
           <xs:element name="body" type="xs:string"/>
     </xs:sequence>
   </xs:complexType>
</xs:element>
</xs:schema>
```

Figure 2.8: A Simple Example of XML Schema Document

An XML Schema defines the elements in a document, the number of child elements of an element, the data types for elements and attributes and the default or fixed values for elements and attributes. With XML Schema tools the developer is enabled to design a project easily and visually in a flow diagram.

XML Schema is the successor of DTD and will be used in most Web applications as a replacement for DTD because of many advantages. XML Schema is richer and more useful than DTD and extensible to future additions. XML Schemas supports namespaces and rich data types and also allows user create own data types derived from standard types.

2.4 SVG Technologies

SVG (Scalable Vector Graphics) [SVG 1.1] is an XML-based language that describes two-dimensional graphics as a W3C recommendation. The robust functionalities of SVG allow graphic objects, animations and also interactions. With SVG Document Object Model (DOM) it is possible to completely manipulate all of the elements and the attributes. The Event Handlers in SVG such as “onclick” and “onmouseover” provide access to any defined graphical objects. With rapid development of web applications and mobile devices, the advantages of SVG are presented more and more.

The SVG 1.1 recommendation [SVG1.1] defines three main profiles: SVG Full which includes all modules, SVG Basic, and SVG Tiny that are targeted at mobile devices, cellular phone devices and PDA.

2.4.1 Fundamental Elements in SVG

In this section the basic shapes in SVG are described with simple examples.

Rectangle:
The <rect> tag creates a rectangle with variations.

```
<rect width="300" height="100"
      style="fill:rgb(0,0,255); stroke-width:1; stroke:rgb(0,0,0)"/>
```

Circle:
The <circle> tag creates a circle by defining coordinate of the center and radius of the circle.

```
<circle cx="100" cy="50" r="40" stroke="black" stroke-width="2" fill="red"/>
```

Ellipse:
The <ellipse> tag creates an ellipse closely to a circle. The difference is that an ellipse has two different radius in x and y directions.

```
<ellipse cx="300" cy="150" rx="200" ry="80"
         style="fill:rgb(200,100,50); stroke:rgb(0,0,100); stroke-width:2"/>
```

Line:
The <line> tag creates a line by determining start and end point of a line.

```
<line x1="0" y1="0" x2="300" y2="300"
      style="stroke:rgb(99,99,99); stroke-width:2"/>
```

Polyline:
The <polyline> tag defines a set of connected straight line segments that form an open shape.

```
<polyline points="0,0 0,20 20,20 20,40 40,40 40,60"
          style="fill:white; stroke:red; stroke-width:2"/>
```

Polygon:
The <polygon> element defines a set of connected straight line segments that form a closed shape.

```
<polygon points="220,100 300,210 170,250"
           style="fill:#cccccc; stroke:#000000; stroke-width:1"/>
```

Path:
The <path> tag creates a path following a set of commands for movement.

```
<path d="M250 150 L150 350 L350 350 Z" />
```

Text:
The <text> tag creates any texts.

```
<text x="30" y="120" style="font-size: 18px; fill: #090">
   Here are the SVG Elements!
</text>
```

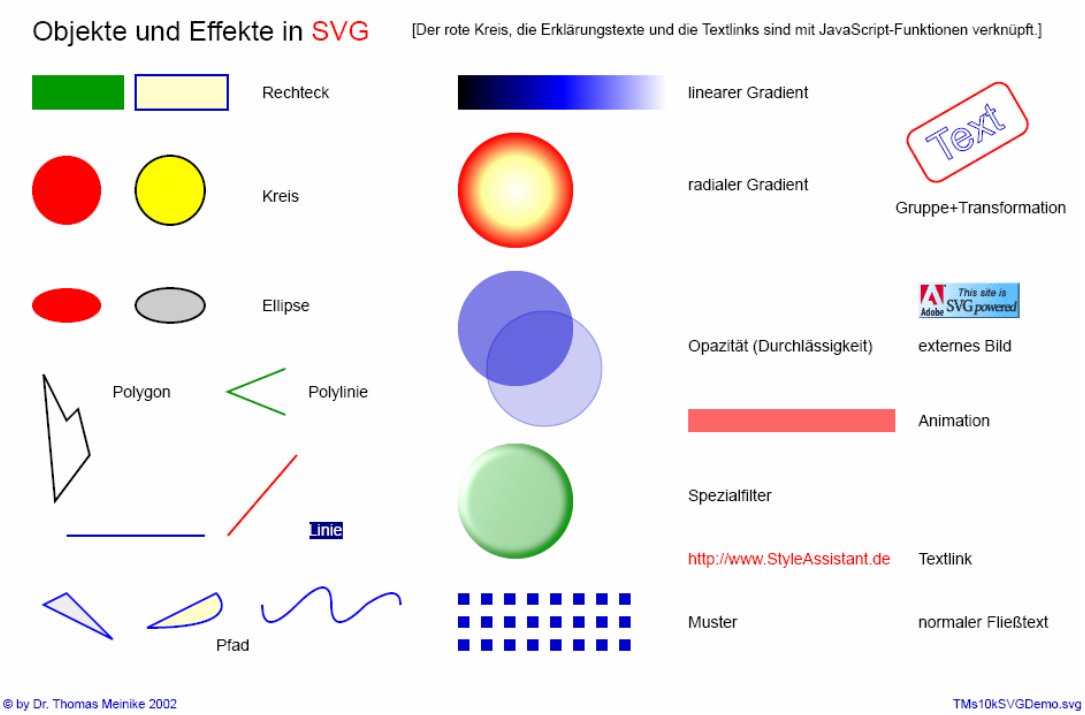


Figure 2.9: SVG Elements and Filter Effects (Author: Thomas Meinike; Source: http://svglbc.datenverdrahten.de/)

Filter effects:
With SVG filter special effects to shapes and text can be added. A filter effect consists of a series of graphics operations that bring a modified graphical result to original graphic shape. SVG allows also multiple filters on the elements. A lot of SVG shapes with filter effects are shown in **Figure 2.9**.

Transformation:
By using transform-attributes the elements or objects in SVG can be transformed. The transformation in SVG includes moving, scaling, rotating and bending. All transformations can be created by the transform matrix. By using design pattern transform-attributes gives the instance of design pattern appropriate position and size.

Animation:
SVG offers elements moving effects, that is, animation of SVG elements is possible by integrating time-controlled actions with graphical elements. The animations can be started by various events, for example, by loading a graphic, by clicking mouse, etc. With animation-attributes the developer can move an object along a path, change the colours, and make an object invisible and so on. The animation elements derive from W3C recommendation multimedia language SMIL.

Grouping in SVG:
Grouping in SVG is important and useful feature of SVG, with the tag <g> it is possible to bind more elements for a certain purpose in a group. The elements in a group can hold shared attributes, styles and be treated as a unit. The frequently used combination of elements can be defined in a group and be reused as needed, in other words, a group plays a role as a module or container. Transformation and animation are performed for all elements in a group. The efficient grouping of elements can shorten the SVG Document and advance programming efficiency.

2.4.2 JavaScript for DOM Manipulation

The script languages such as JavaScript, ECMAScript, Jscript or VBScript [SVG 1.1] [Pohlmann 2003] can be used in SVG document in order to enable interactions and dynamic manipulations. It also allows some precise modification and control of existing SVG objects.

The Document Object Model (DOM) [Phillips 2000] is an API for XML documents. Because SVG is pure XML syntax it is compatible with the DOM. The DOM of a SVG document consists in a tree structure and the nodes in such tree structure can be handled as objects which are accessible for script language. The objects of DOM represent the node of SVG documents that can be manipulated by script language.

The Script is written in the definition area <defs> of SVG document. The JavaScript can be also written in a separate file which is referenced in <defs>, thus the functions in script may be reused by more than one SVG documents.

With methods of JavaScript ***getElementById(), getElementsByTagName()*** the nodes of DOM tree is searched and SVG elements with corresponding Ids and Names can be accessed.
With the methods such as ***getChildNodes(), getFirstChild()*** the nodes of the found elements are returned.
With the method ***getAttributes()*** the attributes of a node are read out. With ***setAttributes()*** the attributes of a node is set or changed.
With methods such as ***createElement(), appendChild(), insertBefore()*** the new nodes can be created and inserted in the DOM tree. It can be determined, on which node the new node should be appended so that the definitive position of the new node in DOM tree can be given.

In the references [Mintert 1996] and [Flanagan 2002] the methods and purposes in JavaScript are described in details.

Chapter 3

Human-Machine Interface for Manufacturing System

3.1 Introduction

Human-Machine system is established for the cooperation of natural human and technical systems. The information exchange between human and machines is realized in the layer of human-machine interface that enables interaction between human and machine. In the manufacturing system the machine denotes the tool machines or electronic equipments and the human is the control person or supervisor who controls the machines by using human-machine interface. The modern manufacturing system exerts the best power under best cooperation of human and machine.

In this chapter as basis of this work the fundamental of human-machine interface are described, the roles of human and machine played in manufacturing system help us understand the human-machine interface. The requirements of modern human-machine interfaces are principles for all development works in different problem domains, whatever mechanical engineering or software engineering. Addressing to the today's problems in manufacturing system, an EU project INMAS is running in order to resolve the problems and to extend the EU's leading role in manufacturing. Developing efficient and effective human-machine interface is a part of the project and will be introduced in this chapter. Moreover, the tasks of developing the human-machine interface focused on production process control system will be specialized. The approach of how to accomplish the tasks will be compactly described in this chapter.

3.2 Human-Machine Interface

For a long time the human-machine system is developed on the basic of empirical gained experiences without duly considering the special abilities of human. The incompatible adaptation of system to the human leads to defective operations or accidents. For example, the false adaptation to human should on the one hand cause the information flood that overloads the human capability but on the other hand not exploit the full capability of automat machine.

Figure 3.1: Cooperation of Human and Machine
(Source: Project' Proposal INMAS)

The human-machine interface depicts general and technique-dependent cooperation of one or more natural human with technical system [Wahl 1999]. The effect of information exchanging, in the other word the interaction between human and machine provides the approach of resolving those by costumer specified problems and interpreting by operator self-defined commands. The transition at the boundary between human and machine is defined as human-computer interfacing after considering the standard DIN 44300. Upon the control system, from machine-side the technical components is summarized in the form of hardware and software adopted for controlling the machine, from the human-side, the human can join in the technical processes and access the information data.

Control system is a form of interface between human and computer, and the dialogs upon the control system play important roles in communication between human and machine which visually describe the information exchange for accomplishing a task. Normally, the human machine interface consists of combination of dialogs. The basic

component of dialog is menu dialog because the options are classified normally through the menus. The modern window-oriented user interface is built with more side by side arranged windows that each of the windows present different information. The principles of presentation for process control using display screens are described in guideline VDI/VDE 3699 which specifies regulations regarding the design of displays (such as mimics, curves and messages) for cases of using full-graphical display system for process control.

To transmit the information in manufacturing branch, many media are employed such as optic, audible and tactile methods [Johannsen 1993]. By the reason of great noise in manufacturing system the information is transmitted mostly through the visual channel. In some special situations such as danger alarming the audible mean is employed. In order to accelerate the perception and interpretation the information should be classified according to the grade of importance.

3.3 Human and Process Communication

The Human-Process communication is an integrated cooperation that works together with information acquisition via sensors, information processing by control system and information feedback which supports control personnel to manipulate the processes with the captured information in real time. In process control system a process works for a certain part of whole production that can consist of more than one sub processes. The information of process such as process value and machine state is gathered by the sensors. This process information is handled by process control system and displayed via user interface on output devices to control personnel. Then, the control person can manipulate the process according to requirements by inputting commands or other actions.

In process control system the configuration of individual equipment is components of the software of control system. The planning of physical devices can be simulated and configured in the software that is designed in analogy to devices layout and similar connection. The important components to be considered for modeling the human-computer communication are devices layers, process operation functions, process-oriented layers and situation-dependent presentation [Boeckler 1996] [Peters 2002].

3.4 Requirements of the Future Human-Machine Interface

With rapid increase of the functionalities and complexes in modern manufacturing, the modern information technologies of human-machine system must be employed. In order

to improve productivity more and more developers and institutions apply themselves to develop modern human-machine interfaces. There are some requirements of future human-machine interface system.

- The interactions of human and machine must satisfy the increasing requirements of safety and dependability.
- The models of interaction and presentation should base on the today's technologies. The future human-machine interface should overcome impossibly the shortcoming of today's monitoring and control system.
- The user interface for process control system must be stable and work continuous because the instruments for production work often full continuously.
- The approach of implementing user interface should follow current standards and adapt to the increasing requirements of web services.
- The system should allow updates and extensions without destroying the stability. The configuration must be executed efficiently, quickly and with least errors.
- To design human-machine interface of monitor and control system two challenges must be considered: how to build graphic interface and how to use the generated interface to control process.

3.5 Task Description

Today's manufacturing systems consist of machines which accomplish tasks according to previously defined deterministic programs downloaded from the central server. Production parts are transported from machine to machine by robots. All machine tools are supervised by human personnel who adapt the individual parts to the whole production process. Because the manufacturing of a single part consumes a considerable amount of time the quality control is difficult to be realized during the production process. Error diagnosis is difficult due to the complexity of tasks and lack of support for the supervising personnel.

The INMAS project addresses the current problems of the today's manufacturing system by developing the Smart-Connected-Control Platform that will allow controlling the production plant in a revolutionary way. For diagnostic purposes the valuable sensor data can be sent to the SCC platform through innovative mechatronics. Intelligent robots deliver goods in a manufacturing plant on the routes that are determined in real

time by the SCC platform. The human operators can be integrated in the maintenance and diagnose process. The supervisors and the maintenance crews will receive the information of product parts and machines through PDAs during the production process. For advanced manufacturing system the advanced control of robots and machines is important. The supervisors must gather the information in time for the production process planning and error diagnosis.

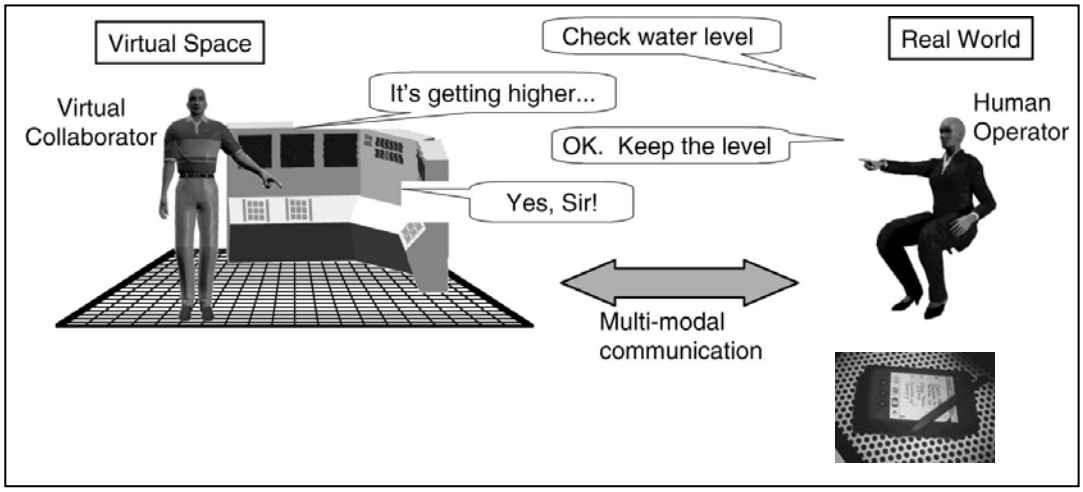


Figure 3.2: Communication between Real Human Operator and Virtual Space (Source: Hidekazu Yoshikawa, [Jacko, Sears 2003])

The user-friendly interface is a part of the SCC platform that visualizes complex data. The user interface will be embedded on PDA handing by the control personnel and on the display wall in a production plant [Ballagas 2005].

With help of ubiquitous augmented and virtual reality the control and customization will happen in a virtual space, the resulting of control and specifications are returned to real human controller and evaluated and implemented by SCC platform in real time. Upon SCC platform the human operator sends commands the virtual space. The communication between real human operator and virtual space is illustrated in **Figure 3.2**.

The Human computer interface is developed according to the different situations in which human computer interactions will occur. The best suited user interface form is a two-dimensional graphical presentation and therefore a web interface. Due to the complexity of the system the presentation will also be complex. Hence, a careful modelling of the use cases and the application is necessary in order to guarantee a good usability. This design will be implemented according to W3C standards and a hardware abstract form.

3.6 Approaches

To develop a human-machine interface, the problems are analyzed from abstract to concrete. In this section the approaches of developing user interface for process control system of manufacturing are explained. **Figure 3.3** illustrates the phases of developing a human-machine interface that depict the designing approaches from abstract to contract.

1. At first for understanding the whole system the developer must comprehend the goals of developing the user interface, i.e. forwhy and for whom is the user interface developed. For process control system, which goals will be achieved by the control system should be defined as prerequisite of development.

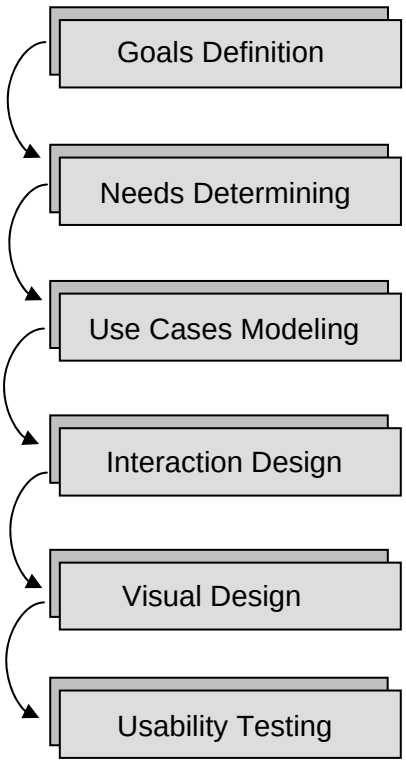


Figure 3.3: Phases of developing the Human-Machine Interface

2. After determining the goals of the user interface for a process control system, the next step is to determine needs of the production process in order to decide what should be contained in the user interface for process control. For the production processes it should be considered, which production processes or machines employed in the processes should be monitored, which process values are necessary, and which machine states are controllable.

3. Addressing different problem domains analyzed by last step the task models

according to different use cases can be built. The model of the whole production process is decomposed into individual components, and the individual component is modelled in an entity model. Finally, the model of whole system is built step by step by associating all entity models according to their relationship.

4. The next step is to transform the task models to presentation models which realized the interaction between human and machine. The exchange between information data and presentation objects is important for interaction design. Besides, the possible user commands and the effects are considered for interaction design.

5. With visual design the looks and layout of user interface are designed. Usability test is the last step which tests whether the user interface is useful or not, with usability test the shortcoming of developed user interface can be detected and corrected. The phases of visual design and usability test are not emphases in this work and will be not described in details.

Chapter 4

Determining Needs of Process Control in Different Problem Domains

4.1 Introduction

The process control engineering encompasses all technical means that assist human to control a process according to requirements. In automation system the processes are controlled by monitoring the process values and allow full access to measurement and configuration of data. The users, such as supervisors, want to monitor the production process machine operators in real time and the machine operators should operate the process if it is necessary.

The analysis of requirements for process control system is foundation of building models of human-machine interface. The restriction of requirements is the basis to choose GUI technologies for implementing the user interface. To capture the requirements for a process control system there are many problem domains to be considered. The use cases of the problem domains are based on two main goals, one is which information is necessary to build the human-machine interface and the second is the possible situations by using the user interface.

To achieve the goal of process controlling there are three important problem domains to be analyzed.

1. The display of process values upon the human-machine interface that assists the control personnel to monitor the production process and product quality in real time. The control personnel acquire the information of process values per output devices such as monitor, display wall or PDA, the approach is shown in **Figure 4.1**.

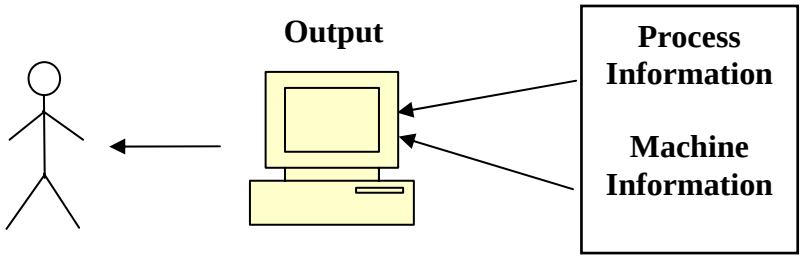


Figure 4.1: Process Information is displayed on Output Device for Human

In this domain it should be considered which general production information is important for the control personnel to master the whole production process, which measurable values of process or running machine are necessary for the process control and which parameters and attributes of the values are required for the process values.

2. In modern process control system the control personnel must have the ability to control and operate the processes. During the processing of a production process the control person can change the process values in order to adapt the process into a satisfied state for the whole production, as illustrated in **Figure 4.2**. In this case the configuration of changeable values and their types is in the foreground.

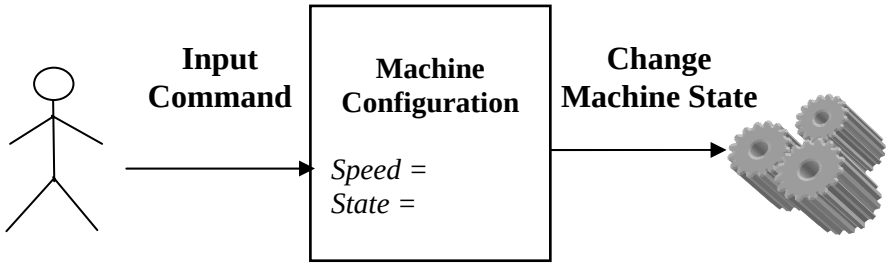


Figure 4.2: Operating Processes and Machines

3. In the lifecycle of a manufacturing system, from planning to assembling, there are many production phases to be executed step by step. The information is maintained in different database and documents according to different tasks and usages. The information of different functions is integrated in a phase model (in [Reuth, Künzer 2001] named as “task sub-network”). In other words, each component of phase model presents a process element of the production.

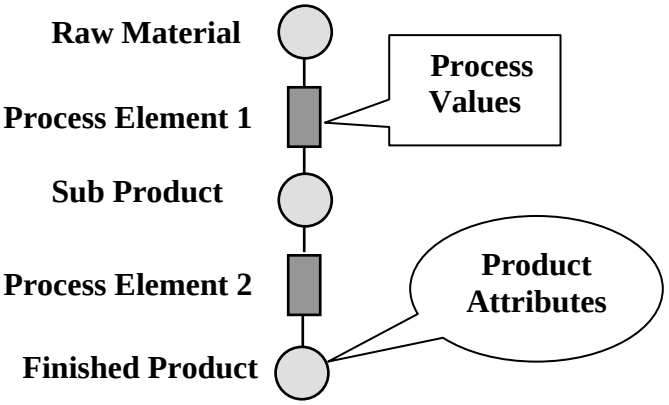


Figure 4.3: A Simple Phase Model

To integrate the models of process elements the phase model of production process should be built. The connections between the models of process elements are constructed in the process phase model according to the correlations of process functions and the tasks of production. **Figure 4.3** shows a simplest example of phase model, in this model a product is produced through two process elements, from first process element the sub product is produced from raw materials, afterwards the sub product is handled to a finished product by the second process element. In the praxis, the phase model can be very complex.

In this capture the typical use cases for the modern human-machine interface are analyzed in the above mentioned three problem domains. These use cases are the foundation for modelling the process control system.

4.2 Monitoring and Control System

4.2.1 Introduction

Monitoring system is greatest part of process control system. In human-machine system the users are machine operators and supervisors who monitor and manage the production process. Visual display of process values and their parameters are the most direct means for the control personnel to acquire the real time information of production processes and the history information for monitoring the trend of production processes. Today in process controlling system there are various arts of representations to be adopted, therefore, the information and the data types are important for choosing suitable graphic objects for user interface of a process control system.

4.2.2 Process Value Display

The necessary function of human-machine interface for process control system is to display current process value because the control personnel must acquire the process value for process management in time. In order to choose adapted display objects the requirements capture must be analyzed.

For this purpose the following cases should be considered:

- **General Information**

In the early design phase, it must be considered which general information should be displayed for the whole process. The general information is the important information that must be known not only by the control person but also by all of the workers deployed for the production process. For example, the information of the whole process, such as descript of product, model of machines, number of workers, date and time and so on, is required by all process control systems. **Figure 4.4** shows some typical general information of a process.

Title: General Information of Process ER001

Terms	**Information**
Date	10.10.2005
Time	10:30
Product Model	PASSAT
Task	Assembling
Customer	Firma xxx
Supervisor	Meister Li
Workers	38
Target	85 / day

Figure 4.4: Example of General Information for a Process

- **Inner information of an atom element**

For the atom element such as a running machine the possible useful information is the inner information such as the running speed of a machine, the incoming product and outgoing product, the state of machine, current value and target value, etc. An example of inner information of a machine in a production process is shown in **Figure 4.5**. The inner information is important for modeling an entity with the information as attributes.

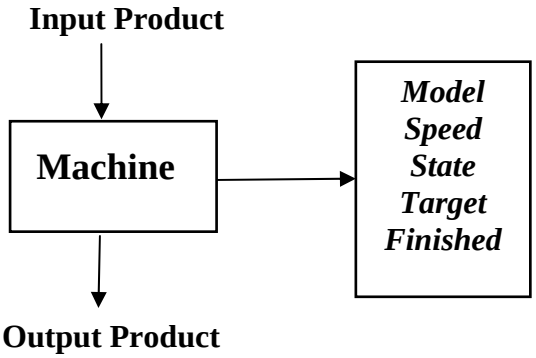


Figure 4.5: Inner Information of a Machine in Production Process

- **The exact address from where a value is coming.**

Because there is a great of information in a process control system, it is necessary to display a process value with exact notation in order to avoid confusion. For a value the importance is the address from where the value is coming, that is, for which machine or process the value is measured.

- **Attributes of a value**

For a single value, at the first it must be considered if the value is measurable or not. Furthermore, for a measurable value it should be determined whether the value has a unit or a measurement range. The unit and measurement range will directly influence the representation of a value.

There are two examples of different attributes of values illustrated in **Figure 4.6.** The values in (a), supervisor and product are the simple types which are not measurable and have not units and measurement range. The values in (b), temperature and speed are complex types, they are measurable by instruments and for them the units and measurement ranges is indispensable information for presenting the values.

Value without unit and measurement range:

Name of value	Measurable?	Value
Supervisor	False	Meister Li
Model of Product	False	PM001

(a)

Value with unit and measurement range:

Name of Value	Measurable?	Value	Unit	Min	Max
Temperature	True	25	Grad (°C)	18	32
Speed	True	2500	U/Min	2000	3000

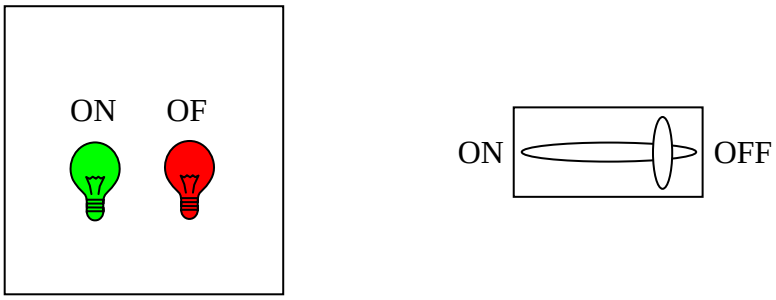
(b)

Figure 4.6: Examples of Possible Attributes of Values
(a) Values without unit and measurement range;
(b) Values with units and measurement ranges

Because the unit of a value denotes the attribute of the value and range of a value is important parameter for displaying a value, all of these can influence the choice of the suitable representation objects.

- **Controllability**

To display a value, weather the value is controllable is an important factor to be attended. If the value is controllable the suitable controllable representation element should be chosen. For example, if the machine state is only observed by the user a lamp symbol can be employed by changing the colours according to machine state, it is illustrated in **Figure 4.7(a)**. If the user can manually change the state of machine a controllable graphic element such as a slider shown in **Figure 4.7(b)** must be chosen.



(a) **(b)**

Figure 4.7: Representation of values according to their controllability.
(a) Alarm Symbols for Representation of the uncontrollable value;
(b) A Slider for Representation of the controllable value

If a value is dynamic the change of the value could influence other relational values or functions. Therefore, it is necessary to identify the value by giving each value a unique name as its Id. If a value is changed the information is sent with the Id of the value and then all of the values and functions related to this value are awaked and changed in real

time. The display refresh frequency should be also considered for the dynamic values such as temperature and speed of machine. The choice of representation object for a value is dependent on the type of value.

- **Position and size**

For generating presentation models of user interface the possible size and position of the value must be considered. Alternative representation objects for the value are dependent on the GUI technique that is chosen for the human-machine system.

4.2.3 Process Value Alert and Alarm

For a reliable process control the ability of response to exceptions and errors in the case if a value goes out of range is important and necessary. If the control person is just alarmed when the value goes already out of the threshold, it is helpless to avoid the dangerous errors. Therefore, the alert range of a value should be defined in order to alert the control person while the value goes nearly to the threshold value. The **Figure 4.7** illustrates that the temperature keeping in a range from 18°C to 27°C denotes regular status, lower than 18°C and higher than 27°C the alert is started.

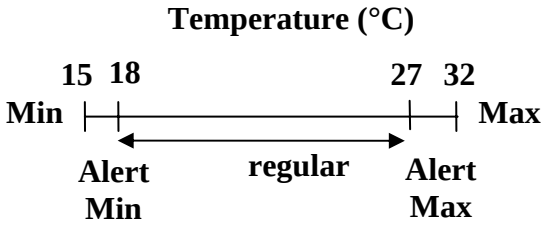


Figure 4.8: Range of Alert and Alarm for Temperature

The presentation object for the value with alert and alarm range must respond to the action of alarm. For example, if a value goes out of range a graphical alarm object can appear or the colour of presentation object for the value changes dynamic according to the current value.

4.3 Statistical Process Control

For the quality control of a manufacturing system not only the current process values but also the trend and history or statistics of the values interest the control personnel. In

the other word, it should be compared what is happing currently and what happened previously. In database of production process the process values during a certain time should be stored in archive.

Normally the history and trend of a process are displayed in a diagram or a chart with an axis for value and an axis for time. The following questions should be considered for the statistical process control:

- How long the history should be reported
- How many values should be compared in a diagram
- How many curves should be displayed at the same time
- The scale of coordinates
- How the history of a binary value is presented such as the running state of a machine
- The refresh frequency of the statistical diagram, such as pro minute or pro hour or pro day.
- Has the Diagram more than one abscissa (or ordinate) for different spaces of time
- Has the Diagram more than one ordinate (or abscissa) for different values

Graphic representations:
There are many ways to visualize the process values. In [Engineering Statistics Handbook] the following main investigating tools are described:

- Histograms
- Check Sheets
- Pareto Charts
- Cause and Effect Diagrams
- Defect Concentration Diagrams
- Scatter Diagrams
- Control Charts

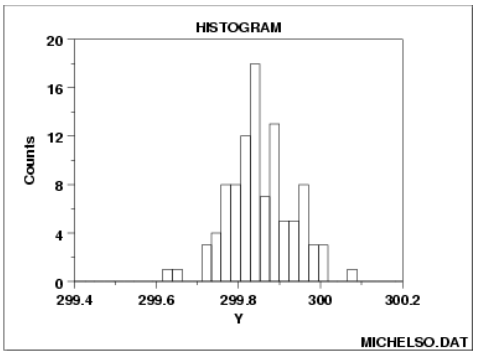


Figure 4.9: Histogram (Source: [Engineering Statistics Handbook])

According to the type of production process only limited types of diagrams are used in a user interface to represent the statistical values. **Figure 4.9** shows a typical histogram for the statistical analysis.

4.4 Phase Model of Product Processes

4.4.1 Phase model

A process of manufacturing system can be described in different models. Te model of workflow of process is a phase model. A process can be decomposed into more process elements and each process element can also consist of other process elements. A process can exist individually or proceed as a successor or predecessor of another process. The simple process contains process elements, input product, semi-finished products from process elements and output elements.

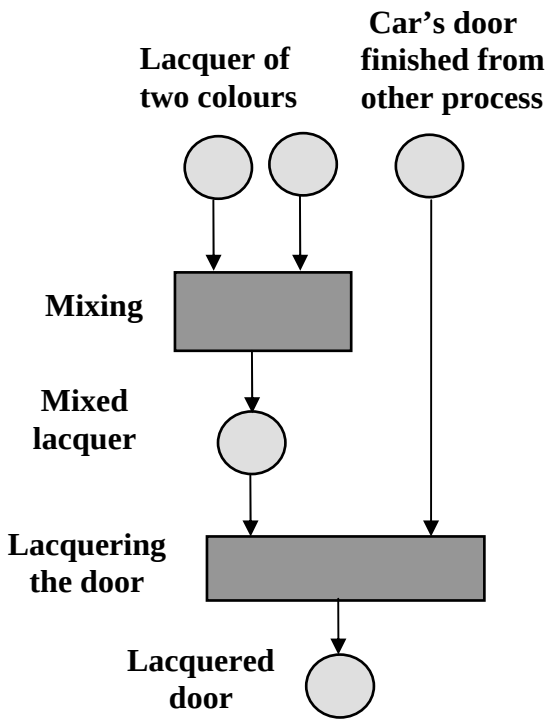


Figure 4.10: A Phase Model of the Process for Lacquering the Door of Car

In the phase model, each process element must have an input product and an output product. The output product of a process element can be used as input product for another process. For the process control system the developer can design the phase

model after the real process structure, the process elements are positioned after the real layout of the plant.

Figure 4.10 shows a phase model of the process for lacquering the door of car. The door of car is an output product produced from other processes and acts in this process as input product. The first process element (Mixing) accomplishes mixing of the raw materials and the output product from this process element works together with another input product (door) through the second process element (Lacquering).

4.4.2 Inserting and Deleting Process Elements

According to the requirement of production process it is possible that the new process elements must be inserted in a phase model or a finished process must be deleted from the phase model. After inserting or deleting a process element the phase model should keep its consistency.

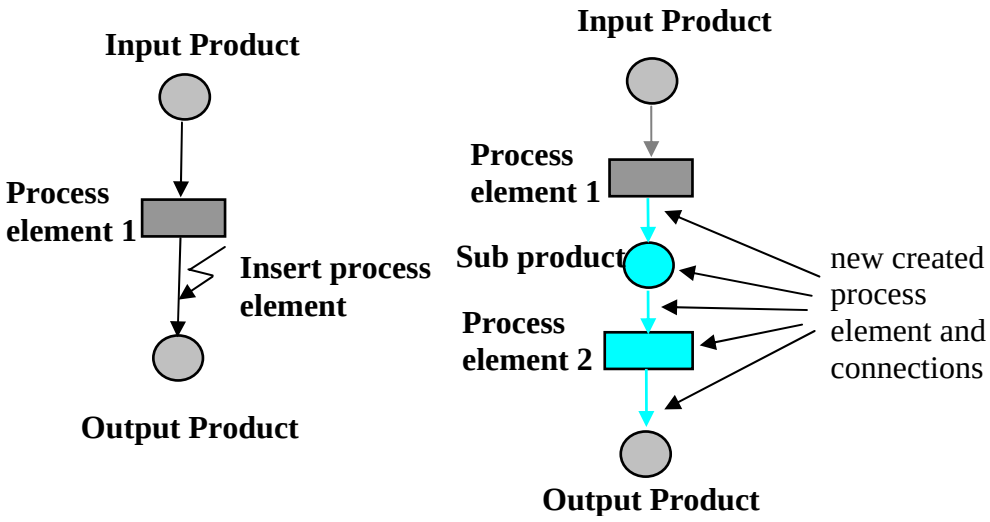


Figure 4.11: Insert a Process Element into a Phase Model (modified [Peters 2002])

In **Figure 4.11** a simple phase model with one process element is expanded with a process element. To insert a process element into a phase model not only a new process element is created but also the connections at the insert point must be modified according to the new conditions. For example, the connection between the process element 1 and output product in the old phase model is broken and then the new connections are built in the new phase model between them.

In this chapter use cases of process control systems are described in different scenarios. The complete graphical formal Use-Cases diagrams in UML-notation are not accomplished in this work that can refer to [Oesterreich 1997], [Bleek 1999] and [Ambler 2004].

Chapter 5

Model-based Dynamic User Interface for Process Control System

5.1 Introduction

After analyzing the requirements of production processes it is possible to build task models according to these requirements. Because manufacturing system consists of production processes and each production process is succeeded by cooperative works of machines and robots, then the human-machine interface of manufacturing system can be modelled step by step from simplicity to complexity. At first the models for simplest entities such as products and machines are built with their attributes, each entity must have a unique identifier in the whole production process in order that it can be easily referenced by the other models without confusion. Based on the models of all entities it should be considered how the entities are combined for a complex function. In this chapter it is described how the modelling of production process control system is accomplished by associating the simple entities models step by step.

5.2 Identifier

In order to let a process control system run trouble-free it must be avoided that a component of system has different meanings or a notation corresponds to more than one component. The ambiguous relationship of system components will lead to confused operations and false control. Therefore, the uniqueness of all components in the control system is necessary.

The identifier of a component is a solution for realizing uniqueness of components. To identify the components of process control system the functions of components stand in the foreground. Each component with different functions should be identified differently. In following sections the methods and influencing factors to identify the components are studied in different cases.

5.2.1 Product

Product is the essential element in a manufacturing system because the final goal of manufacturing is to produce some products. The product is indispensable component throughout all manufacturing processes.

From raw materials to finished products a product is manufactured through many process elements, the process elements produce semi-finished product and parts of product which are assembled by the assembly processes. Therefore, in the whole production process there are various products for the different usages. For an effective and clear process control system, it is important to distinguish the different products. In order to distinguish different product in different process a product must hold a unique tagging.

Raw material:
Raw materials are especial products in production process. The raw materials are used only once and not reusable. It is simple to identify the raw material. It needs the name of material and for which process the material is supplied. In the model of raw material one Id denotes uniquely a certain material, its model, unit, and for which processes it will supply. **Figure 5.1** shows some influencing factors for identifying the raw materials.

ID	Name of Material	Model	Unit	To Process
M001	Metal Part 1	MP01	Piece	P001
M002	Metal Part 2	MP02	Piece	P002+P003
M003	Red Lacquer	LA01	L	P004

Figure 5.1: Some Influencing Factors for Identifying the Raw Materials

Finished Product:
Because the finished product is the last step of whole process, only the last process employed for the product should be considered to identify the product.

Semi-finished product:
To identify semi-finished product there are many factors to be considered. A semi-finished product is an outgoing product and at the same time incoming product for another process. A product can be produced from one process and then provided for more processes. It is also possible that more processes run for manufacturing one product which is incoming product of a process. To make the product unique in the control system, the product model should contain the information of the model of product, from which process the product comes out and to which process the product will go. The Id of the incoming and outgoing processes should be given for identifying the product. An example is illustrated in Figure **5.2**.

ID	Name	Model	Unit	From Process	To Process
SP001	Door	D001	Piece	P005	P006

Figure 5.2: Some Influencing Factors for Identifying Semi-finished Product

5.2.2 Machine

A tool machine produces or handles a part of product. Product is the kern of a tool machine. The model of a tool machine denotes the function and configuration of a machine. The configuration is internal technical information of the machine and is changed to adapt the product. To identify a machine, the model of the machine and the product handled by the machine is useful information. If a machine can produce different products in a production process, for each usage the machine must hold a single identifier. The position of a machine in a process should also be considered for unique reference in the whole system. The example in **Figure 5.3** illustrates that the Id "M1" corresponds to a machine "CUT01" in the production process "P001", the input product of the machine is "D001" and output product "D002".

ID	Model of Machine	In Process	Product In	Product Out
M1	CUT01	P001	D001	D002

Figure 5.3: Identifier of a Machine

In current automatization system the robots play the indispensable role in the whole manufacturing process. A robot is used not for producing something but for packing, assembly and transformation. A robot is identified similarly to a machine but with the function of the robot instead of products.

5.2.3 Process

Process controlling is most important task in a process control system. A process in the whole manufacturing system works for a certain processing phase. The attributes for identifying a process consist of the name of the process and function for which the process works. If the process is not the initial process or end process in the whole system it should has predecessor and subsequence, as shown in **Figure 5.4**.

ID	**Name of Process**	**Function**	**Last Process**	**Next Process**	**Product In**	**Product Out**
PC002	Lacquer1	Lacquering	PC001	PC005	P01	P02

Figure 5.4: Important factors to identify a process

5.3 Product Model

A product model should consist of not only the tagging of product but also the other important information such as the current and target output of products. A concept model of product depicts the required information for control engineering. **Figure 5.5** illustrates a product model for a raw material. The two main attributes groups are attributes for identifier and the attributes of state information of the product. Attributes of identifier are mentioned in the above section, the state information is how many raw materials are used and how many still remain. Two instances of product model are shown in the **Figure 5.6**.

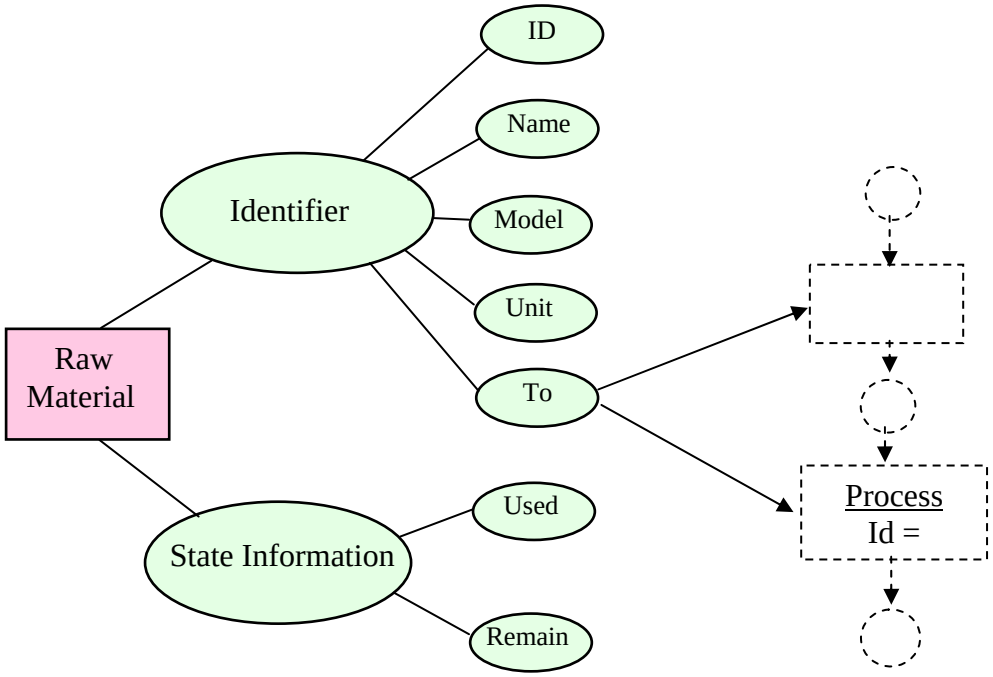


Figure 5.5: A Product Model for a Raw Material

Name	**ID**	**Model**	**Unit**	**Used**	**Remain**	**Supply for Processes**
Metal Part 1	M01	MP001	Piece	200	350	PRC01, PRC02
Lacquer 1	L01	Red	L	500	800	P003

Figure 5.6: Instances of the Product Model in Figure 5.5

A product model can be described in an object. For each special product an instance of the object is created. One product model can correspond to more than one product with variable parameters which are distinguished by the unique Ids. Because each Id corresponds to one product for a certain usage the control person can monitor the states of all products by choosing the product Id.

After determining the required attributes of a product model, the data type of each attribute must be given for continuing works such as implementing the class of product model or representation of the value of an attribute. **Figure 5.7** shows examples of data types of attributes in a product model.

Data Types of Attributes of Product Model

	Types	Examples
Attributes		
ID	String	P001
Name	String	Metal Part1
Model	String	MP001
Unit	String	Piece
From Process	String	P001
To Process 1..n	String	P002
Variables		
Used	Integer + String	200 St.
Remain	Integer + Sting	350 St.

Figure 5.7: Data Types of Product Model

5.4 Machine Model

To model a tool machine the configuration and state information of the machine should be considered. The configuration of a machine is determined by the usage and functions of the machine.

For different functions the information is divided into two blocks, the internal information and the external information. The internal information of a machine exists in form of context-free, that is, the information is independent of other machines and the position in a process. The internal information is mostly defined by the machine manufacturer.

The external information of product is context sensitive that depends on cooperative machines and the position of the machine in a process. The external information determines the logical relationship between the machine and other cooperative compositions.

5.4.1 Model of a Single Machine with Single Function

In a manufacturing system some tool machines produce always fixed product. The function of the machine is never changed and independent of the production process. Although a machine products only one product it can be also employed in various processes. A machine model consists of the information of identifier and the state information of the machine. The information of identifiers was discussed in above

section. The state information of a machine contains the information which relies on the internal function of the machine, such as the running speed, the machine state, and the information associated with production such as the target of production, the finished products from the machine and the percent of defect products. All of the information composes a machine model as in **Figure 5.8** illustrated.

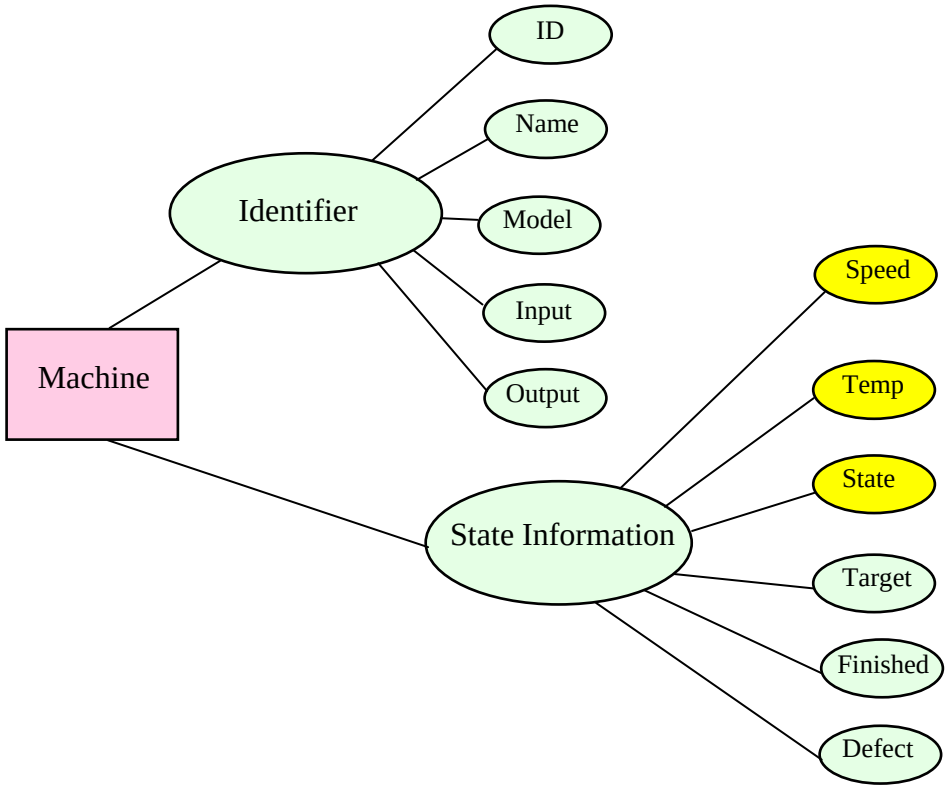


Figure 5.8: A Machine Model

The real time state information of machine is obtained by the sensor and controlled by the control personnel. Each state type is defined with required parameters and attributes. The attributes of state decide also the art of presentation object. Some attributes of state information such as speed, temperature and machine state are complex types which consist of many simple attributes. The attributes of an attribute are shown in **Figure 5.9**, “temperature” is attribute of “state information” but contains attributes for itself such as “MAX”, “MIN”, “Current” and so on.

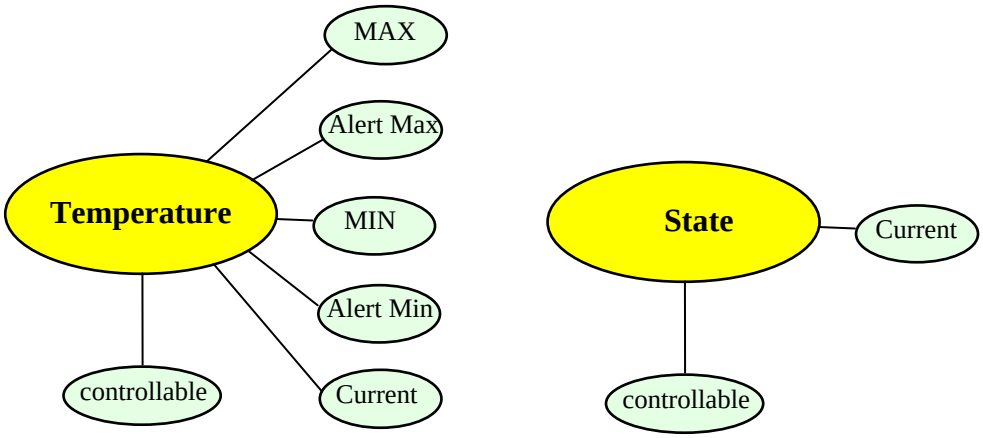


Figure 5.9: Complex Attribute Types

For instantiation of the abstract machine model the types of all components in the model must be specified. The data types for the contents of the model are divided into two groups, simple types and complex types. The simple data types are the common data types such as string, integer, float and so on. The complex data types consist of various simple types and the other complex data types. In **Figure 5.10** the complex data type "Temperature" is illustrated which is built with several simple types, for example the float data type "Max" and the Boolean type "Controllable".

Complex Data Type: Temperature

	Types
Attributes:	
Max	float
Warning Max	float
Min	float
Warning Min	float
Controllable	boolean
Variable:	
Current	float (> Min & < Max)

Figure 5.10: A Complex Data Type

By combining all entity models and the data types of the attributes a complete machine model is build in XML Schema in **Figure 5.11**. In XML Schema the data types can be easily self-defined and be referenced anytime as required. XML Schema diagram enables the developer to have an entire concept of the model.

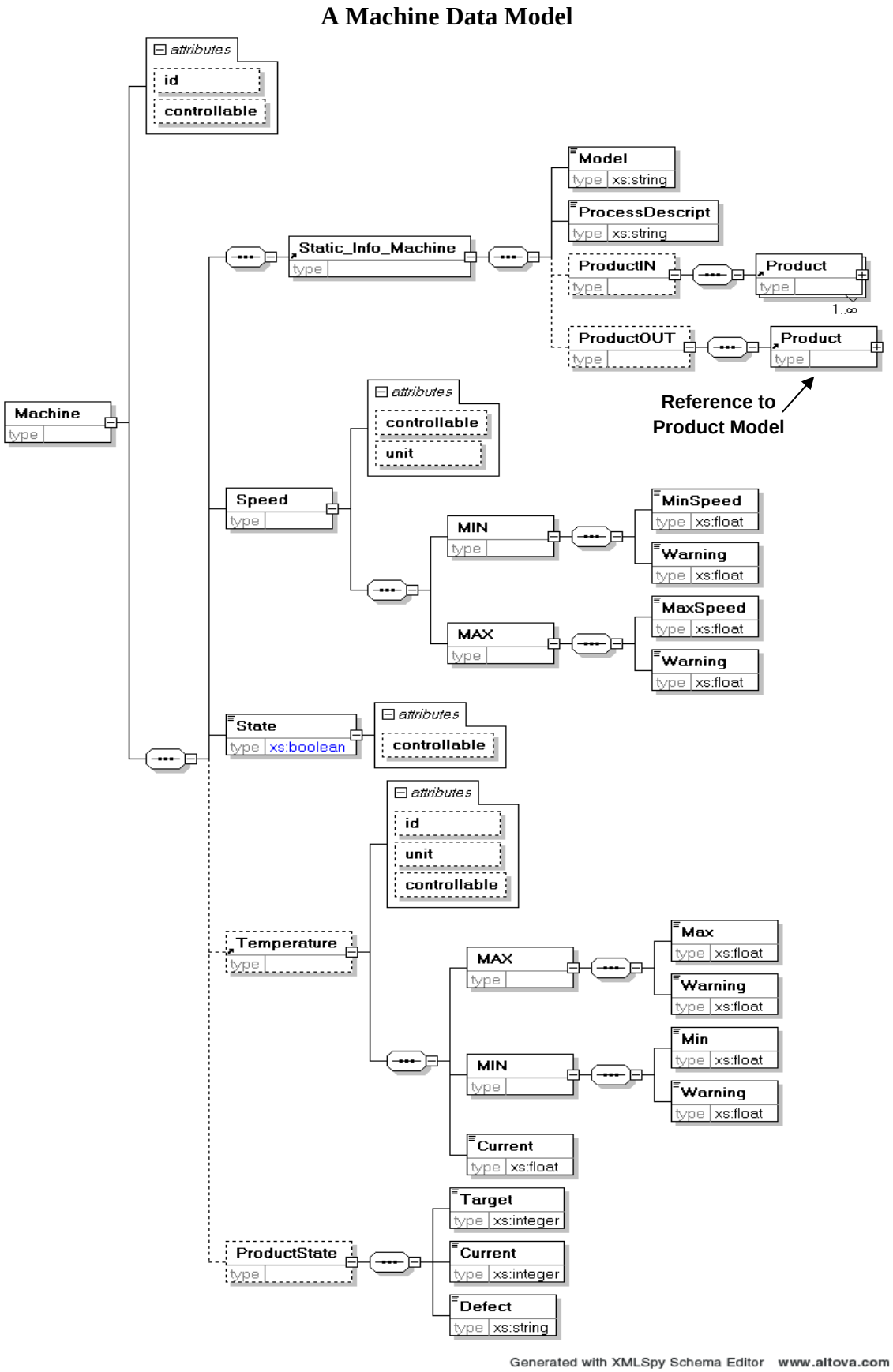


Figure 5.11: Complete Machine Model built with XML Schema

5.4.2 Model of a Single Machine with Multiple Functions

With the development of technologies many machines are designed for multiple functions. With different configurations a machine can be adopted for various usages. According to the requirements of orders and the position in a manufacturing process, the machine plays diverse roles in the system. Therefore, a machine model for such multi-function machines is required for a definite process control system.

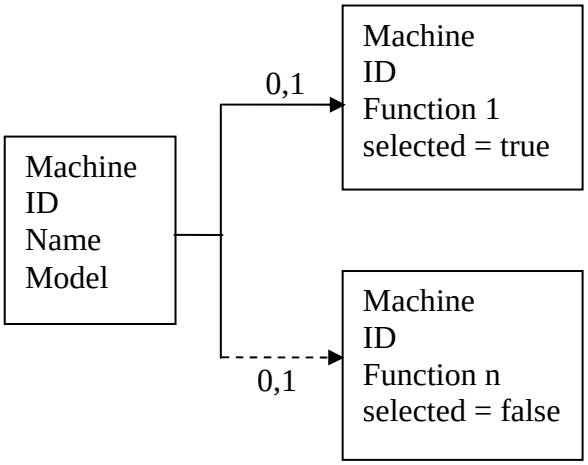


Figure 5.12: Machine with Multiple Functions

The model of a multiple function machine is built in XML Schema in **Figure 5.13**. This model is based on the model of single-function machine. In fact, such multiple-function machine model is a combination of all possible functions and each function is modelled using a single-function machine model and the attribute which decides whether this function is selected. At one time, only one function is allowed to be selected.

Model of a multiple function Machine

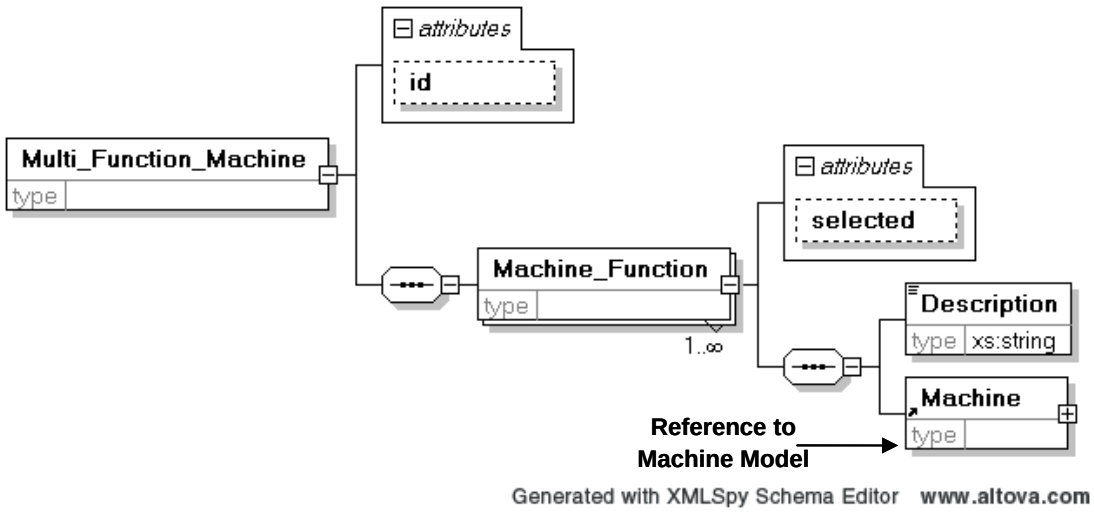


Figure 5.13: Model of Multi-Function Machine built in XML Schema

5.4.3 Model of Machine Group for One Usage

To produce a product or accomplish a process it is possible that more than one machine is employed. Although each machine has its own function and state information during a production process, the machines are indecomposable for one usage. In this case, the machines can be modelled as a group which has unique identifier in the whole system, workflow of a machine group is shown in **Figure 5.14**. The atom elements of the model of machine group are the individual machines. The state information of each machine can be acquired and controlled individually but for the organizational application the machines are handled as a unit.

In the machine group the function of the entire group is interested instead of the function of each individual machine. If the machine group is applied for manufacturing a product it should be focused on the raw materials or input products and output products. The unfinished products that are produced by the machine elements in the machine group during production process are not be considered.

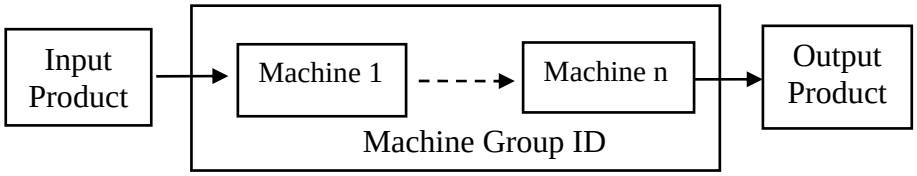


Figure 5.14: Workflow of a machine group

In the model of a machine group the incoming products and outgoing product are interested only for the whole group but not for each individual machine element. Because the machine group works together for producing a certain part of product the production state of individual machine should not be monitored.

In the model of a machine group the machine models are referenced to the machine elements. Comparing to the model of individually used machine, the machine elements in a machine group don't require the information of production state, that is, the target, current produced products and the number of defect products.

Model of a Machine Group

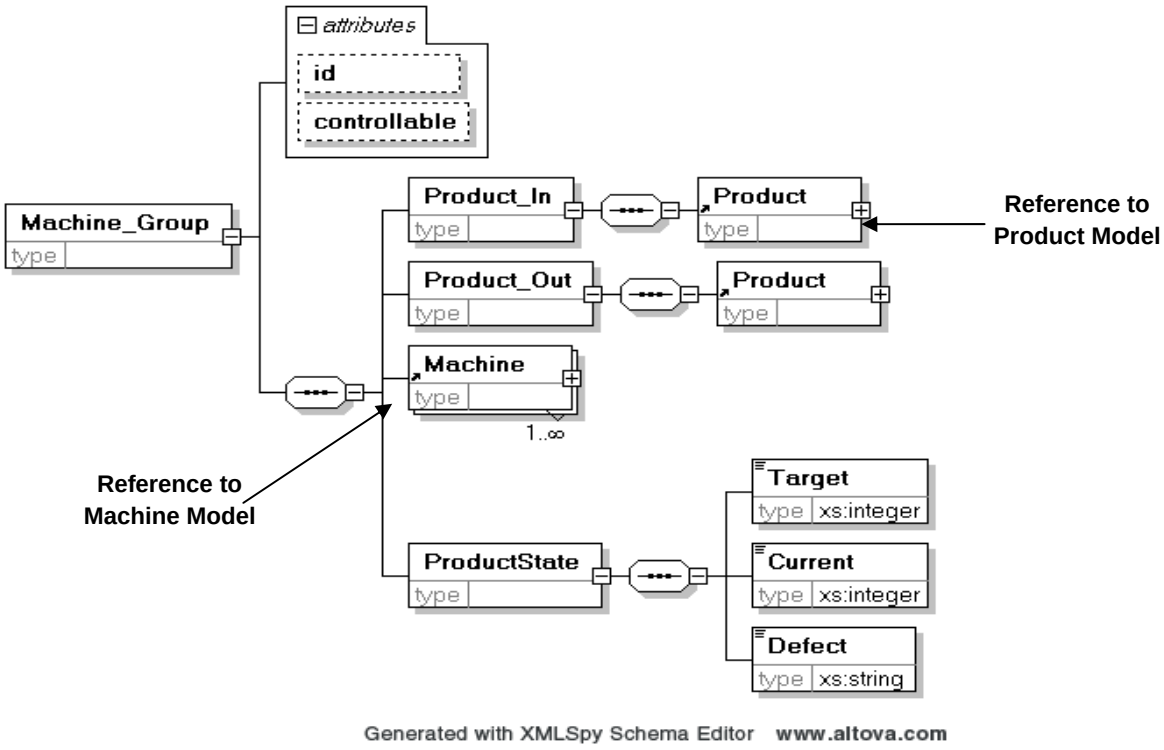


Figure 5.15: Model of a Machine Group

5.5 Process Model

A process describes the function that is completed by serial actions and various means. A manufacturing system is a combination of processes with different functions. A process is defined for a phase of production or processing. A process consists of tool machines, robots and other assistant tools. To control a process it should be focused on monitoring process values in real time.

5.5.1 General Process Information

Normally, the general information of production process consists of a set of data with simple data types. The necessary information is name of each term of the general process information and data type of each term. Due to the simple data types it is easily to represent general process information.

Information	**Types**
Date	date, string
Time	time, string
Name of Process	string
Number of workers	integer
Name of Order	string
Model	string
Customer Information	string

Figure 5.16: Data Types of General Process Information

5.5.2 Model of Process Element

In the process control system the most important task is monitoring and processing the process values, therefore, the information of process values is the most important influencing factor for modelling a process. The composition of a process is machines and products. A process element model can be built by combining the models of product and machines with additional process information. A process model can consist of more process element model. **Figure 5.17** illustrates a process element model, in this model the product model and machine model built in last sections are referenced as components of the process element model.

Model of a Process Element

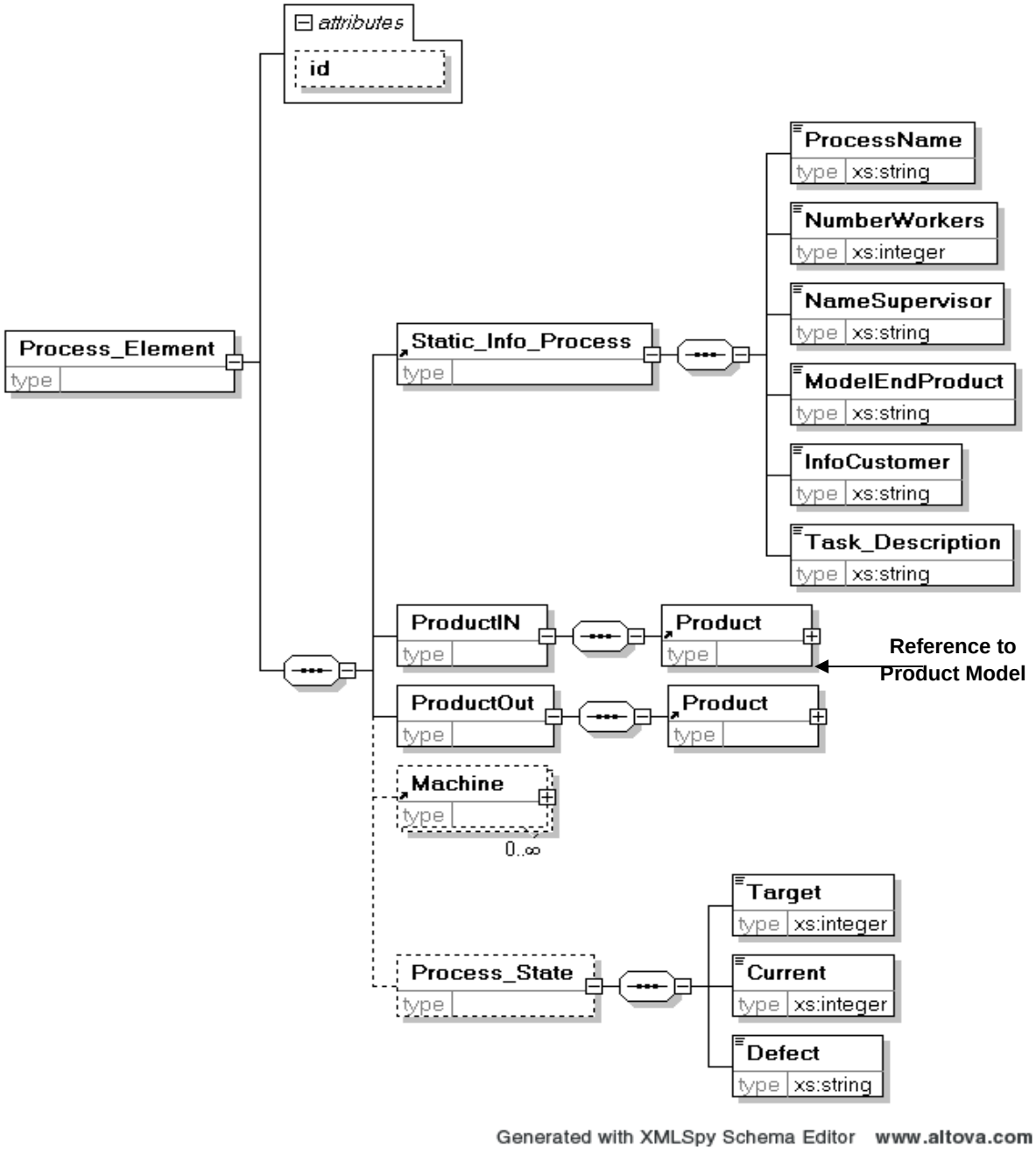


Figure 5.17: Model of Process Element

5.5.3 Connections of Process Elements

A process consists of process elements and products. The process elements and products are connected according to their functions and their positions in a process [Lauber 1996]. The process elements are connected by reference of their Ids, the importance of identifier mentioned in last sections is also indicated here.

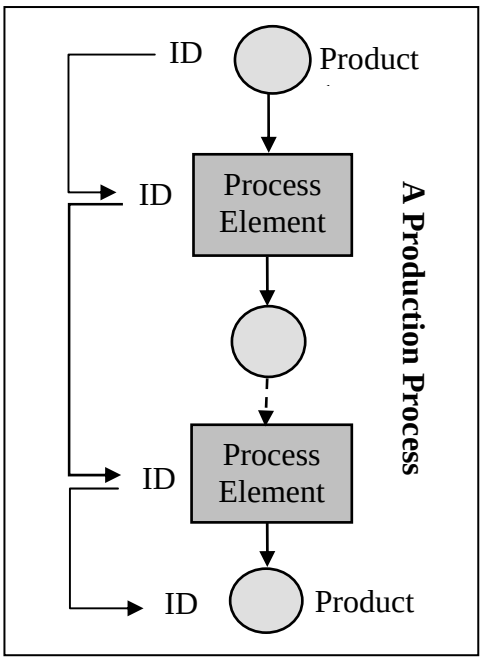


Figure 5.18: Connected Process Elements and Products Models build Process Model

Figure 5.18 illustrates the connections between process elements and products. Each process element references to the product model and other process elements using the Ids. Because each Id denotes a special product or process element, the connection between two components of process can be handled as a relation between a pair of Ids. One Id can correspond to more than one Ids and at the same time be referenced by more than one other IDs.

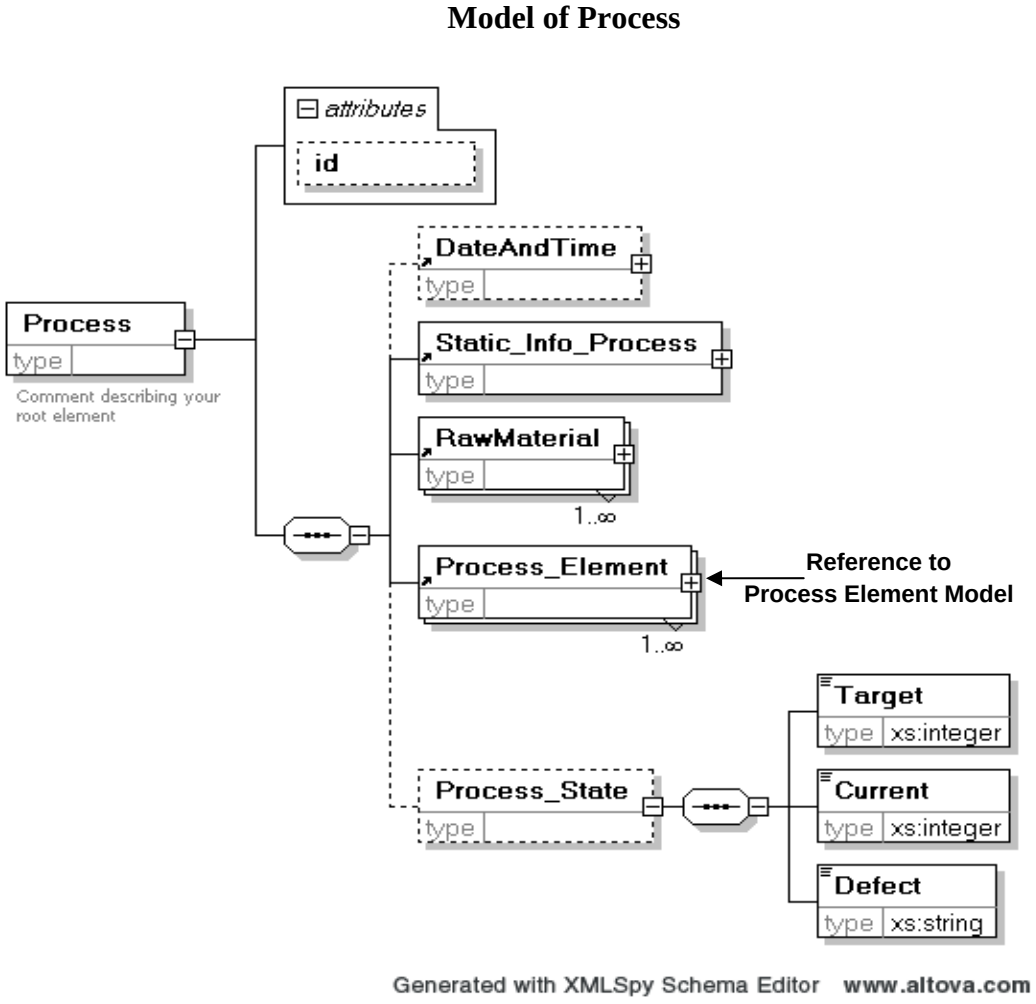


Figure 5.19: A Process Model

5.6 Association of Different Models

For the process control system all function components are modelled with their parameters and attributes. To associate the models for a whole production process control system the functions of the models, the relationship of the functions and the sequence of the models should be considered. For example, the product model is the essential composition for a machine model. A machine requires at least one input product and at most one output product, as shown in **Figure 5.20**. For a process the Id of the product must be unique. The machine model references to the Ids of Input products and output product that ensures the uniqueness of machine function for certain process.

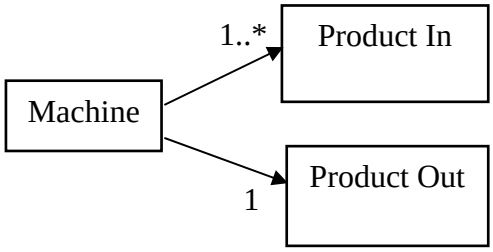
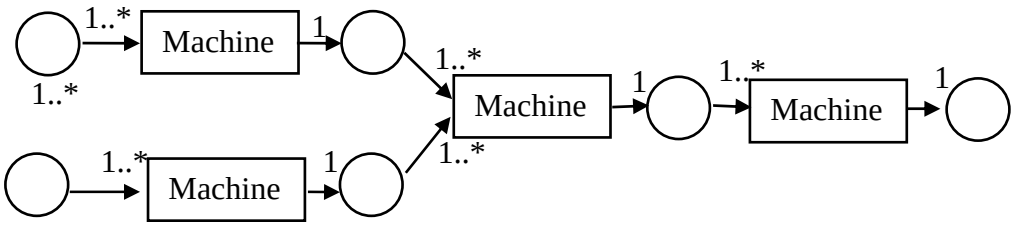


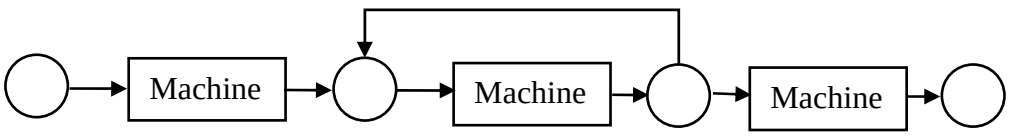
Figure 5.20: Connection of Machine and Product Models

A process element is an association of machine models and product models. Similar to a machine model, a process element model should also contain at least one input product and at most one output product. Because the goal of a process is producing products or processing something, the machines are indispensable part of a process. The sequence of the components of a process is important, in a process the output product of a machine can be handled as the input product for the next machine.

There are two types of workflow for production process system illustrated in **Figure 5.21**, the one is non-circular diagram, and in this form the production process runs always in a direction without return to the previous processes. The process can be determined by a sequence of Ids. Another workflow is circular diagram in which it is possible that the components in such workflow can return to their predecessor during the processing. To describe the components in a circular process, a set of predecessors and successors should be given.



Non-circular Workflow



Circular Workflow

Figure 5.21: Two types of workflow for production process system

In this chapter the models of components in human-machine interface for production process control are built step by step from simplicity to complexity according to the use cases analysis in chapter 5. In actual manufacturing system there are many complex cases of integrating the individual entity models. The relationship between the entities and the sequences of the entities in a relation are the most important factors, besides the number of instances of a model used in a process must be determined. The models built in this chapter include all alternatives of use cases, to create an instance of a model it should be considered, which parameters and attributes are necessary for the process and which can be ignored. After the whole process model is built the values of the attributes and parameter are read from the central server and be refreshed with a certain frequency according to the requirements. If a process elements is completely finished it will be deleted from the process and connections between components are built anew in order to adapt to the new status.

Chapter 6

Transformation from Data to Interactive Elements

6.1 Two-Block Function Model

After the analyses of use cases in process control system and modelling based on the use cases another important function model will be introduced for generating the user interface from the models.

To present the data of useful information, it must be considered how the data can be transformed into presentation object. The concept of two-block function model is adopted for transformation from data to graphical elements [Peters 2002]. With two-block function model the task of human-machine interface is divided into two blocks, the one is data model and the other is presentation model. In the data model, the data of process values is processed in classes according to the data structure and data types. The presentation model describes the graphical objects that present the data in various forms. The bridge to connect the two function model is interactive elements.

In the data model the data is described in various generic classes that build the framework. The framework is the runtime environment for the objects or instances that are created from concrete classes. A task can be divided into different objects. The data types of the concrete classes influence directly presentation objects. The interactive elements play a role as management of data model and presentation objects. The interactive elements translate the data and the commands from users in order to choose the suitable presentation objects in presentation model.

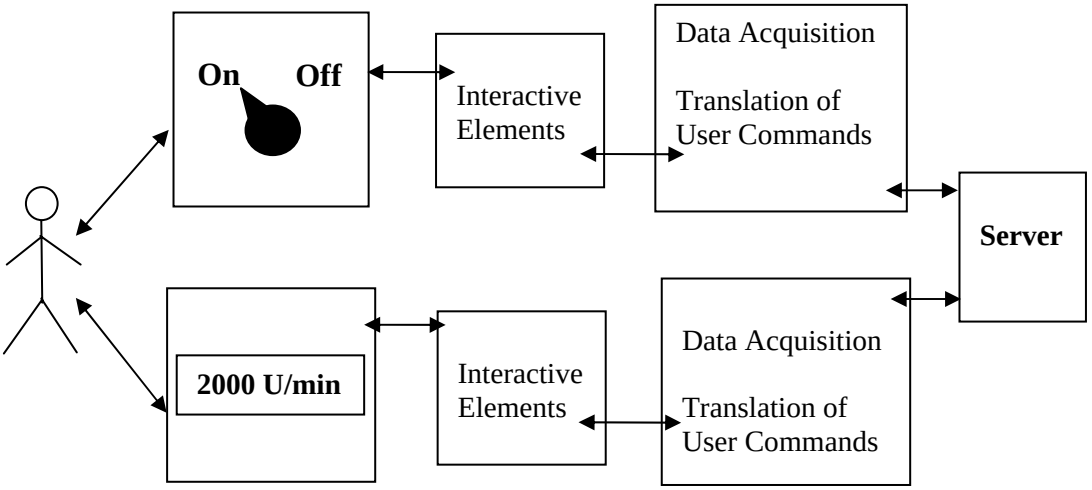


Figure 6.1: Two-Block Function Model for Transformation of Data

The **Figure 6.1** illustrates the function and data structure as the approach of implementation in procedural program language. From global and opened data memory the function can access the objects that can read and write the data. The interactive element encapsulates all of the data in object and permits the access by the method. If the two-block model is adopted, for each presentation object an object in interactive element and the data processing should be created. Consequently, the unique allocation of a presentation object and corresponding data processing function can be determined.

To implement the transformation from data to representation object, the first is to define classes for data processing objects that contain the data structure, data type and user commands. And then the classes for presentation objects are defined to describe graphical presentation objects depending on the data types. Information classes determine the relationship of data classes and presentation classes. In the information class it must be defined which data classes and which presentation classes are relevant and which relationship is between them.

6.2 Interactive Elements

The interactive elements build the interface between data objects and presentation objects. The interactive elements can acquire the data change of process information and then inform the corresponding presentation objects. Inversely, through the presentation elements the user commands can be transmitted to interactive elements so that the data object can actively respond to the user requests. Because each presentation object corresponds to a data object, the corresponding interactive element should be built for the pair of presentation object and data object.

In process control system the interactive elements transform the process data to presentation element, the important information for building interactive elements is described following [Peters 2002]:

- The name of process position that is used to determine the allocation of machine and process control
- The measure unit and measure branch for a process value that influence the choice of representation objects for the value.
- The state of a value or data
- The information whether a process is controllable by the human or not
- The index of the possible user commands

6.3 Presentation Models

The presentation model describes which graphical elements will be presented to the user. The presentation elements should have the ability to display the data and to receive user commands. For a human-machine interface, on the one hand the users receive the data information through visual presentation devices such as display wall or pen-sensitive screen and on the other hand the users can send message through track point or touch-sensitive screen.

The graphical user interface can be divided into more areas and each window display a block of information. In these windows the presentation objects are arranged with appropriate sizes in order that all useful information can be rightly displayed in a screen.

The first step of building presentation models is to decide which presentation elements are needed to present the data information for the process control system. The second step is to specify the frames or windows that contain presentation elements and divide the whole window into several parts according to the functions.

6.3.1 Minor User Interface Elements

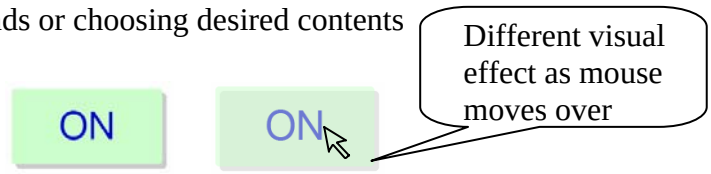
The minor elements of presentation model are the graphical elements which represent the data information such as buttons, input fields, log in field, lists and so on. The minor elements can be further divided into two parts, atom elements and containers.

6.3.1.1 Atom Elements

The atom elements are the smallest presentation elements which are indivisible. The atom elements are predefined in GUI toolkits such as a text label or a symbol. Several

frequently used atom elements for user interface and the important attributes for defining them are described following:

Button:
For confirming user commands or choosing desired contents
Example:



Attributes:

- X-coordinate of the button object, right upper corner
- Y-coordinate of the button object, right upper corner
- Width and Height of button object
- Caption of the button

To enhance the attention of users some special visual effects should be added to the events. For example, the colour of button changes if the mouse moves over a button, when the button is pushed down the button should have the lively sunken effect.

Text Label:
For the displaying name of process or product and so on and the description of process value
Example:

The Process Control System for a Manufacturing Plant

Attributes:

- X-Y-coordinate for determining the position
- Text size
- Text format
- Length of the Label

Text input field:
For inputting of user commands with text, in process control system for the configuration of machine state
Example:

Attributes:

- Text label for describing the text input field
- Position of the Text input field
- Maximum number of characters

List:
For enumerating information of materials, product list, general information
Example:

Date	12.12.2005
Time	16:00
Workers	45
Temperature	23 °C

Attributes:

- Number of terms in the list
- Name of each term
- Data type of each term

The choice of a list must depend on the layout of presentation field. The list fits for the information that is difficultly presented by graphical objects. The content in a list is the overview of certain information for the whole process. For example, if the supervisor will observe the consumption of all raw materials a list is a suitable presentation object better as other graphical objects.

Drop down list:
Used for listing information but with limited place Position. Only one term is displayed if no interactive event occurs upon the list.
Example:

The content of drop down list should contain the information of relevant things. The most frequently used value is set as default term in the drop down list.

Symbol:
Display an object or a command visually such as zoom in and zoom out symbols.
Example:

Symbol can also visualize certain commands or process state. In process control system each process element can be symbolized according to characteristic of the process element. If the supervisor will monitor a certain process he can simply choose the symbol of the process.

Process bar:
For displaying the rate of progress
Process bar can only present the process value but not be controllable.
Example:

Attributes:

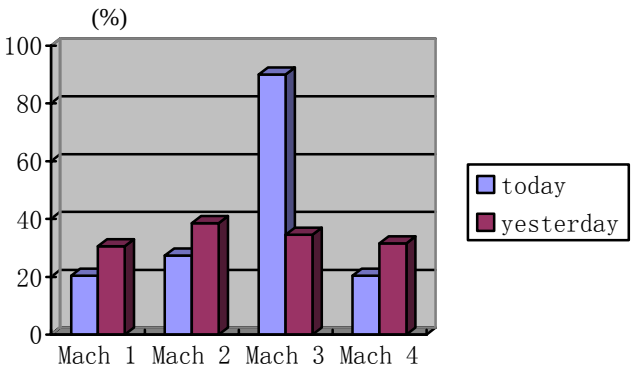
- Position of process bar
- Name of the value for the process bar
- Dynamic text for displaying the value
- Branch of a value
- Unit of the value
- Frequency of refreshing the process value

Slider:
Used for presenting the controllable value
Example:

Attributes:

- Scale
- Value branch
- Unit
- Changeable value text

Diagram:
For the statistical process control, diagram can be used to compare more process values and display the trend of a value.
Example:



Attributes:

- Axes of the diagram
- Scales of the axes
- Values for the two axes
- Distinguish of display for different values by different colours, different lines or etc.

6.3.1.2 Container

A container consists of atom elements and the other containers. The composition in a container can be treated as a set. The components in a container have their own attribute and parameter, at the same time they have also the same attributes depending on the general feature of container. If a container is chosen for a presentation object, all of the components in the container are chosen along with the container. In praxis the elements that present for the relevant functions and exist always at the same time should be encapsulated in a container.

Controller Faceplate:

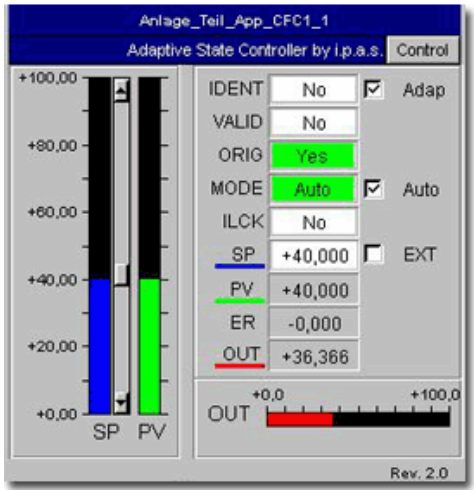


Figure 6.2: ADCO Controller for PCS7 Control Software (Siemens)

In **Figure 6.2** a controller faceplate for configuration is illustrated. This faceplate consists of text labels, input fields, process bars, slider, radio buttons and buttons.

Configuration group:

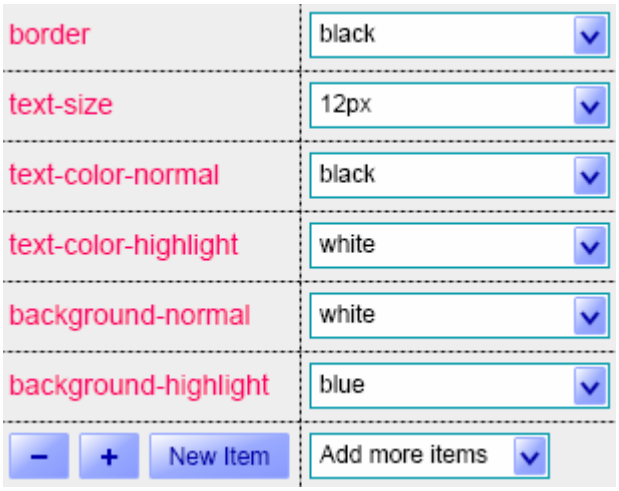


Figure 6.3: A Container for Configuration Group

Figure 6.3 shows an example of a container for configuration group, this configuration group consists of several drop-down lists and push buttons.

Combination of graphical elements to display a value in different forms:
To satisfy different requirements of ergonomics a value or data can be presented by different graphical elements. For example, the time can be displayed by digits or a graphic of clock shown in **Figure 6.4**. The digits present the time accurately and the clock symbol displays the time more directly due to the visual effect.

Figure 6.4: Two Representations for One Value

6.3.2 Major User Interface Elements

The major user interface elements are windows and dialogs which split the information in several blocks according to the functions. Windows provide the fundamental way in which the user views and interacts with data. The consistent window design is important because it assists users to easily complete their tasks and to gather the desired information.

Dialogs are supplemental secondary windows that can be used to allow users to specify parameters or to give commands or to offer details of objects and actions included in the primary windows.

Because windows provide access to different types of information, for designing such major user interface elements the developer should focus on the primary usages and functions of the user interface. It must be considered, which information should be grouped in a window or a dialog and how the windows and dialogs should be arranged in a screen. A screen should not be broken into too many windows otherwise the attention of users will be troubled by the unwanted information.

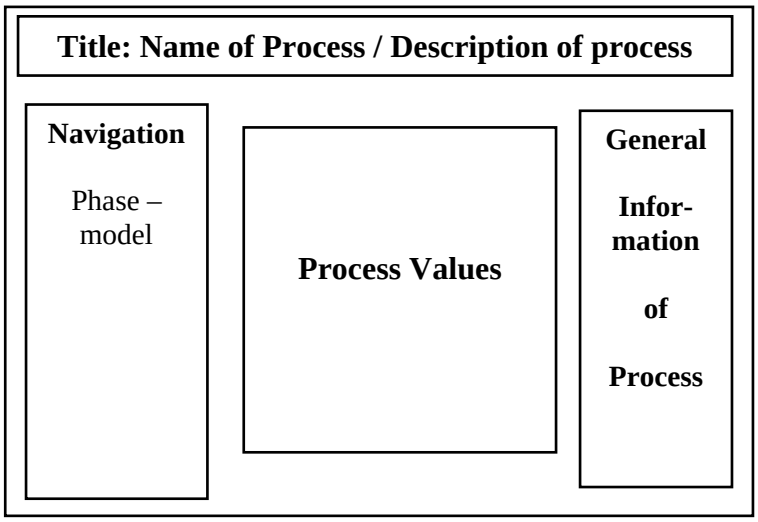


Figure 6.5: Major User Interface Elements for Process Control

6.3.2.1 Main Window

A particularly descriptive or memorable title can awake the big response of the reader. Therefore, choosing a best title for the windows is a complex matter that requires serious thought.
For the user interface of process control system the main window should contain the important information of the production process.

Title of the main window:

- The name of the manufacturing factory
- The Name of the production project
- Simple description of the production process
- Model or serial number of process

The title of the main window is an overview of the whole production process. The length of a title should not be more than two rows.

Color of background and text:

The colour of the background of main window and the text in it must correspond to the ergonomic rules. The rightly selected colour can improve task performance by attracting the attention of users, but contrarily the misuse of colour is dangerous.

6.3.2.2 Navigation window

For a production control system there are abundant process values to be controlled, it is impossible to display all information of a process control system on one screen. At one time the supervisor can only monitor limited number of processes. How the control personnel can access the desired processes to control depends on the navigation system. The navigation window is a part of field on the screen where the main content of the system is presented in a navigation diagram. The navigation system can exist in the form of indexes, menus, navigation tree, or navigation flow chart.

For a simple and concise navigation in process control system the graphical navigation flow chart is a good selection because the navigation flow chart can visualize the production phase for quick switch between processes.

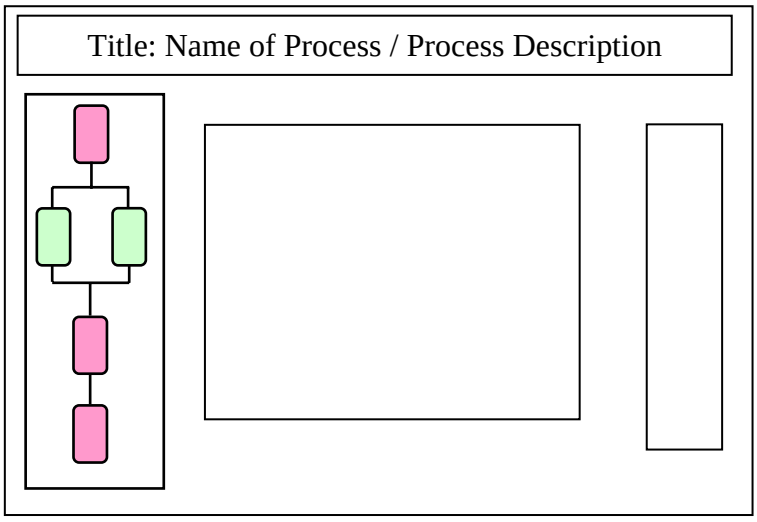


Figure 6.6: Navigation Window for Process Control System

In the navigation flow chart the sequence and relationship of components should correspond to the real process phase model. The flow chart can be organized by several main process phases.

Each process phase is represented by an icon and each icon must be annotated with name of process phases in order to assist control personnel to distinguish the icons immediately.

There are some tips of visual effects for navigation window that assist the control personnel easily and quickly to find the desired information.

- The symbol of currently progressing production phase should be highlighted.
- The currently displayed process phase should be highlighted but different from the progressing phase.
- The means of highlight colours are explained nearby the navigation flow chart.

6.3.2.3 Information Window

The Information window is the main window where the process values and common information of production process are displayed. Such information windows interface provides a means of viewing and editing the information and viewing the content and properties of the process values. The information windows can be also used to display information such as parameters of process and user input for configuration, palettes of options and setting and messages to inform the user of a particular situation [Microsoft 1999].

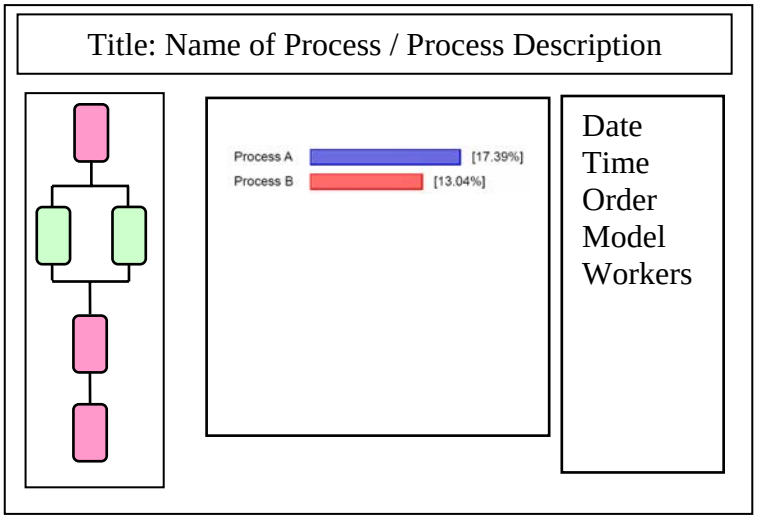


Figure 6.7: The information Windows to display Process Values

The lifecycle of the information window is an important influencing factor to be attended. For example, the display area for common information may be indispensable

for the control personnel throughout the whole production process so that such area must be always presented or interrupted only for a short time. Contrarily, the Information of process values may be presented only temporarily according to the needs of control personnel or disappear when the process is finished.

6.3.2.4 Dialogs and Message Boxes

Dialog Boxes:
A dialog offers the exchange of information between the user and the application. The user can carry out commands and tasks by using a dialog box to obtain additional information. The essential components in a dialog are described in standard ISO 9241 and [Microsoft 1999].

The essential components in a dialog:

- **Title bar text:** Because the dialog box appears after the user clicks a command button or shortcut, the title text for the dialog should be defined with name of associated command.

- **Close and help buttons:** A Dialog appears when the user needs and must be closed as the task ends. Normally, the user closes a dialog by pushing the close button at the top right corner. A dialog should provide the user help and suggestion if button for help information is pushed.

- **Dialog box command:** The popular commands in a dialog box include OK and Cancel command buttons. The command buttons should be labelled to clearly define the purpose of buttons.

The layout of a dialog box is dependent of the output devices. There are guidelines in ISO 9241 and [Microsoft 1999] for desktop-sized computer screens. In a manufacturing plant the large display can be employed for displaying process values. The current wall-size displays typically offer seamless display surfaces that are created by multiple tiled projectors [Wallace]. The design for large-size screen is in principle similar as the desktop screen but with some variations because of the technical reasons [Collomb 2005].

Message Boxes:
Message box sends only messages to the users but does not receive commands from users. When a message box appears is not decided by users but by the system. Generally, message box comes out in order to warn the users by abnormal work status or to confirm the user commands before they are performed.

Similar to the dialog box a message box has also the essential components like title bar, close button and command buttons. Additionally, there are symbols at distinct positions in a message box which denote the primary functions of the message, for example:

Information symbol: provides information about the results of a command.

Warning symbol: Warning message alerts user to a situation condition that requires user's confirmation before an important command is performed.

Error symbol: Informs the user of serious problems that must be corrected in order that work can continue.

6.4 Navigation Models

Navigation design is critical step in the design of user interface that is constructed based on the conceptual models. A good navigation system assists the user to easily access the desired information. Building a navigation model is helpful not only for the application structure but also for a more increased structured navigability.

In [Ambler 2004] the navigation model can be structured in a user interface-flow diagram, also called windows navigation diagrams or context-navigation maps. The flow diagram offers a high-level overview of the user interface for the application and enables the developer to model the high-level relationships between the user interface elements.

The navigation model is effective combination of all the behaviours derived from the use cases, in [Constantine, Lockwood 1999] the result is called the architectural view of the user interface. The high-level overview approach assists the developer to easily understand the complete user interface for a system.

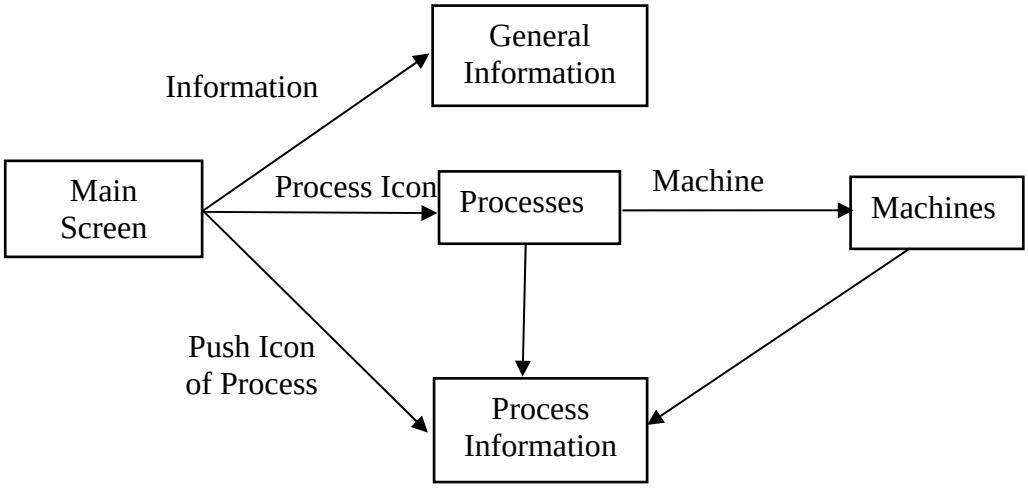


Figure 6.8: Workflow of User Interface Model

Figure 6.8 illustrates an example of navigation flow diagram. The boxes present the major user interface elements and the arrows represent the possible flow between them. For example, the Processes icons in the navigation field take the users to the process information screen. In the position of process information screen it is possible either go back to the main screen or go to the details of machines working for the process.

Because the navigation flow diagram presents a high-level view of the interface of a system, on the one hand, it enables the user to understand how the system works as expected and on the other hand the overall flow of user interface can be validated in this position.

Chapter 7

Developing Graphical User Interface with SVG

7.1 Introduction

In the modern process control system of manufacturing system, the control person has more and more requirements for quick and precise acquisition of information. In a control system it is possible that many devices are employed to monitor and control the process. For example, a display wall in a manufacturing plant or control centre can display the process value and all of the useful information. With PDA the control personnel can control the production process without location limit. Therefore, the platform-independent generated user interface system is required for the human-machine interface of modern process control system.

7.2 MVC Model in SVG

The frequently used MVC model for software architecture is employed for Web applications based on SVG implementation.

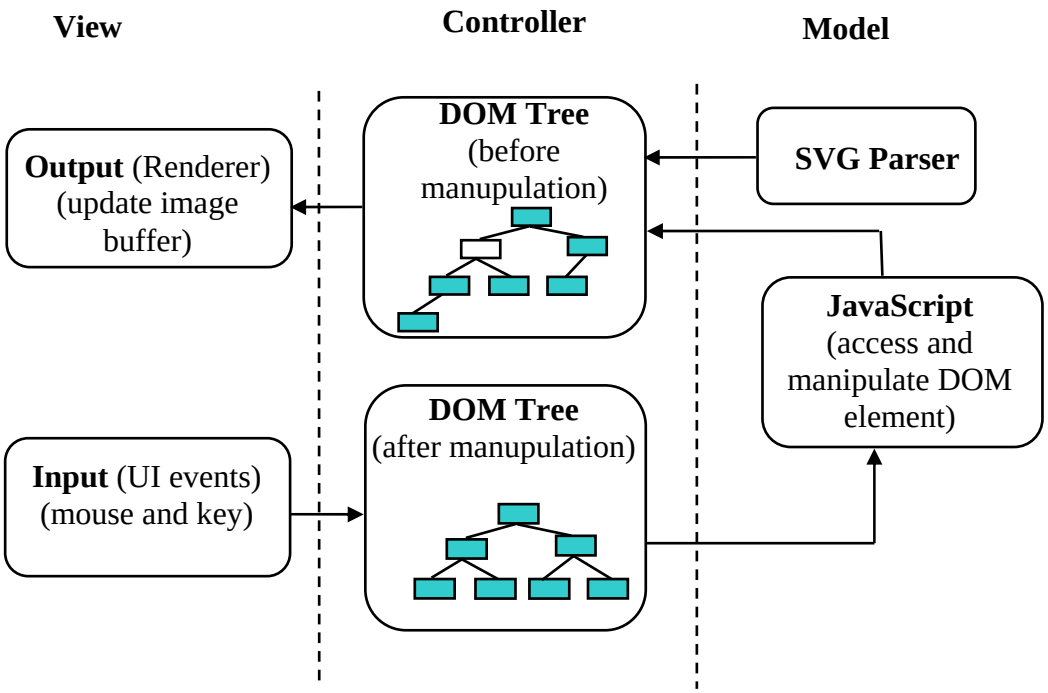


Figure 7.1: MVC model used for SVG Application

Figure7.1 illustrates the SVG application based on MVC software architecture model. The SVG application is decomposed into separate Model, View and Controller components.

The Model consisting of DOM and JavaScript (or ECMAScript) defines the state information about the interface application and the logic of software components. In particular the model can be used to store information about UI widgets such as the state of a button, the current value of a process bar.

The View offers visual representation of the model. The separate View enables developer to change the looks without influencing the application's functionality. The View includes interface components that can be dynamically rendered to the client side. There are different means that can give the interface a different look such as the SVG skinning and CSS.

The Controller works as the connection between the Model and the View. Generally, the controller handles events by manipulating the DOM tree of SVG application. In [Qiu 2004] the controller is described as communication through event-based messages in a message-based MVC architecture of SVG application for web service.

7.3 Implementing UI Patterns in SVG

The mechanism of patterns design is one of the most useful benefits of SVG. With pattern the repeatedly used UI elements are defined in a pattern that will shorten the length of code and ease the work of developers. The UI patterns can be called at any time with suitable attributes and parameters.

7.3.1 Atom

The pattern of an atom UI element with SVG consists of fundament graphic elements and the relative distances between the elements inside the pattern. The absolute coordinate is defined with (0, 0). The structure of an atom pattern will be explained with an example of a simple push button.

```
<g id="button" style="filter:url(#filter)">
   <rect x="0" y="0" width="80" height="30" rx="5" ry="5"/>
   <text x="18" y="20"> onclick </text>
</g>
```

Figure 7.2: SVG Content for a Button

The SVG code section defines a simple push button with text caption. The effect of the push button is shown in Figure 7.3. In the group with Id “button” a button pattern is defined. “style” defines effects of the button such as shade, spotlight. “rect” defines form of the button and “text” defines the caption of button.

Figure 7.3: A SVG Button

In order to enable the ability of repeatedly using the button, the button can be defined in the definition area comprised by the tag <defs>. The different filter and CSS decide the different looks of the button.

By referencing to the Id of the button pattern the developer can create and insert a push button at optional position by giving the coordinate. The code in **Figure 7.4** denotes that a button identified with “b1” is inserted in the position (15, 15).

```
<use id="b1" xlink:href="#button" x="15" y="15" />
```

Figure 7.4: Create Instance of a Button

7.3.2 Container

Similar to the atom UI widgets, a container is defined by combination many atom widgets or other containers in a group. The idea for defining containers is to place the atoms and to add extra functionality depending upon more than one widget.

An example of a container for displaying temperature is shown in **Figure 7.5**. In this example the temperature is represented in different forms. All information of temperature is packed in a container. As soon as the container is referenced, it will be handled as a node and then inserted into the DOM tree. Therefore, the container can be manipulated for desired events such as delete, move, zoom in, zoom out, etc.

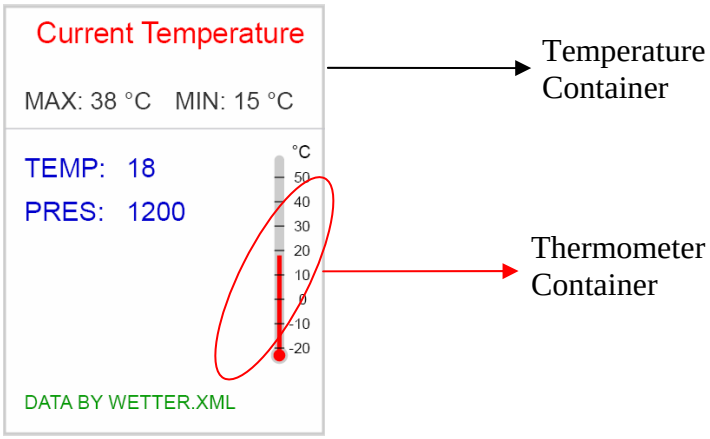


Figure 7.5: A Container for Displaying information of Temperature

In this example the container for presenting information of temperature consists of atom elements such as texts and rectangle and another container for the symbol of thermometer. In SVG the container of thermometer is defined in a group <g> consisting of atom elements and included in the group defined for the container of temperature information.

Because a container can contain other containers, the mechanism of “group” in SVG is advantaged. By defining containers in group it is possible to enable the developer to separately modify the compositions of container without having to change the code of whole container.

7.3.3 DOM Manipulation for UI Elements

The user interface models that are described in the frontal section can be implemented efficiently by using DOM tree of SVG. Because the main window of user interface consists of many sub windows and each sub window contains of their sub windows, consequently the tree structure can be adopted for the user interface. The main window is handled as the root of tree and the sub windows can be implemented as its children nodes derived from the root. Similarly, each node can have its own children nodes

The structure is illustrated in **Figure 7.6**. For process control system, the process values are read dynamically from the server and displayed to the control personnel. In order to enable the control personnel to actively access the data the process values should keep always in active refreshing status.

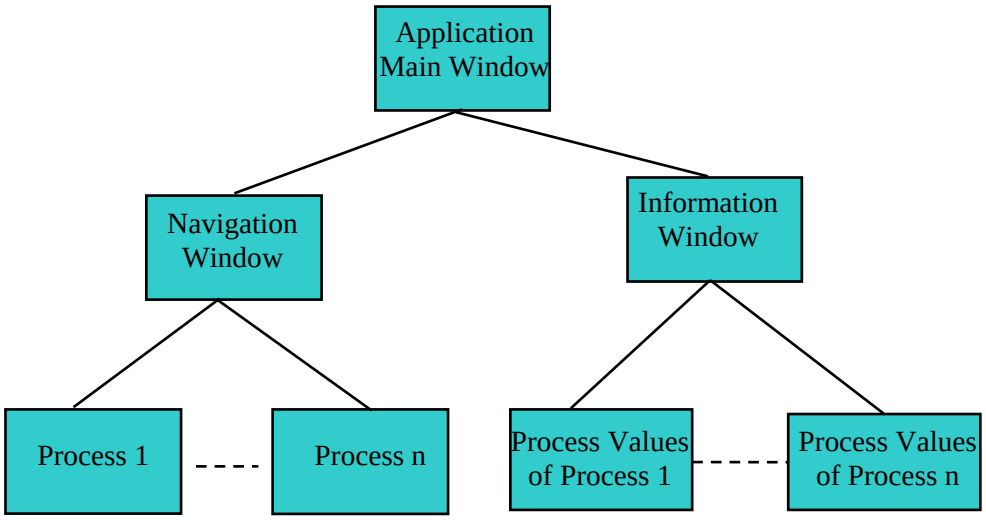


Figure 7.6: The structure of SVG DOM tree for presenting user interface elements.

Steps:

- Create container for the main window in a group which contains the containers of next hierarchical layer. That is, the container for navigation window and process information windows. The group of main window is stored in SVG DOM as root node.

- Define the container for navigation information in a group. In this group the phase model for the whole process control system is visualized. Each process element can be represented by an atom widget such as a button or a process symbol. For clear monitoring the currently running process and process selected to be monitored should be highlighted with different effects such as changed

colour and flickering animation. The container of navigation window is inserted into DOM tree as child node of root node.

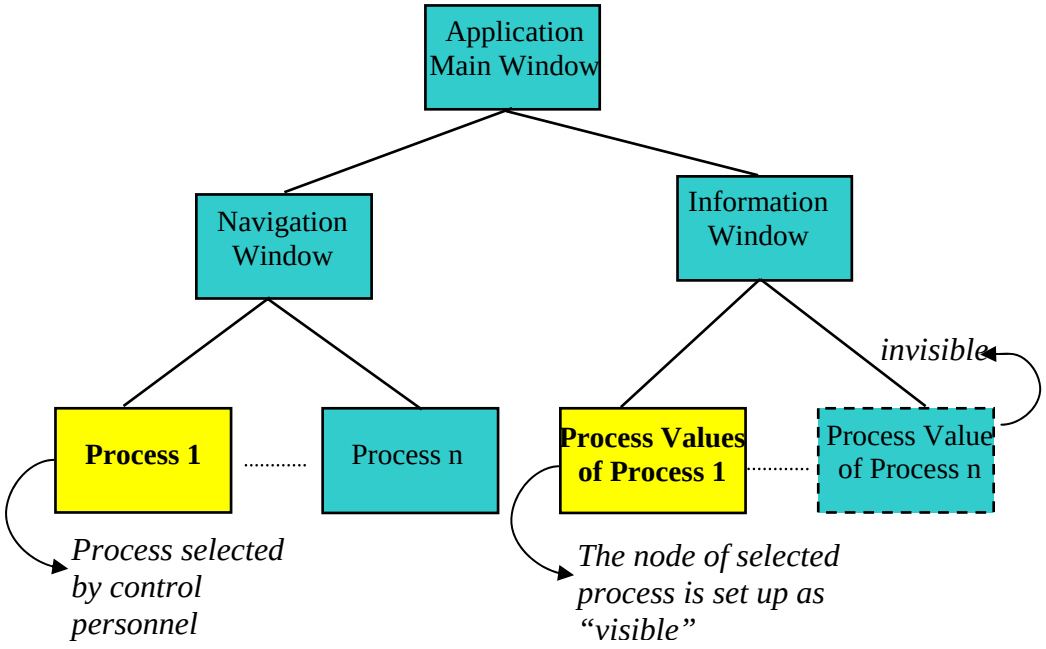


Figure 7.7: Displaying Selected Process Values by Manipulating Attributes of Node

- The information windows are the most important windows to display the process values for controlling. In this window the corresponding process information is presented to the control personnel when the control personnel select a desired process element from the navigation window. In SVG the information container can be defined in a group consisting of all process elements and the information of each process element is stored in a separate group. The group of the whole information window is stored in SVG DOM as the child node of root node for main window, and the group for information of each process element is inserted into the node of information window as its children nodes. With the benefits of SVG group function the group can have shared attributes; it is possible to display only the node of process element selected by the control personnel and to hide all of the other nodes with attribute "invisible". In fact, all information of process elements is stored in DOM tree and actively refreshed. Only by the requirement of control personnel the desired information is represented to the user.

7.4 Interaction in SVG with JavaScript

7.4.1 Dynamic Data Access from XML Database

In process control system the data exchanging is the most important function because the control personnel must acquire the data of process values in time. Currently, the popular database is often maintained in XML data format. The SVG is W3C standard so that SVG can support rightly data exchanging between XML database and SVG presentation objects. The connection between XML database and SVG graphical objects is Script Languages, there are many Script Languages and in this work JavaScript is used for the examples. **Figure 7.8** illustrate the approach of data exchanging between XML database and SVG representation objects.

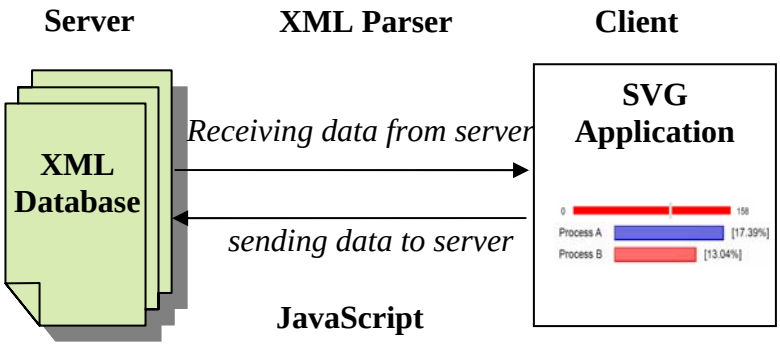


Figure 7.8: Data Exchange between XML Database and SVG Presentation Objects.

- **getURL()**

The method *getURL(filename, fileLoaded)* with JavaScript is defined on the window object to allow to retrieve data from data to a URL. The code in Figure 7.9 realizes loading remote XML Document with *getURL*. *getURL* supports Adobe Viewer 3.0 and Batik 1.5.1

```
function loadFile () {
            getURL ("wetter.xml", fileLoaded);
}
function fileLoaded (data) {
   if (data.success) {
       alert (data.contentType);
              alert (data.content);
   }
}
```

Figure 7.9: Loading XML Database with JavaScript

- **parseXML()**

With the method *getURL()* above only strings can be sent or received. Using *parseXML()* it is possible not only to receive SVG content from the server and append it to the existing SVG tree but also to sending SVG code generated on the client to the server, in which case the document fragment should be convert into strings. Function for parsing XML document is shown in **Figure 7.10**.

```
function parseAndAddData (string) {
        var node = parseXML (string, document);
        document.documentElement.appendChild (node);
}
```

Figure 7.10: XML Parser in JavaScript

- **setTimeout ()**

setTimeout() method is used for time controlling. This method determines how long after a function begins to run should the function be executed repeatedly. The method is useful if the regularly refresh of data required. The code in **Figure 7.11** denotes that a function *timeAndDate()* is executed with a frequency of 1000 ms. If the function runs for displaying current time and date, the time is refreshed every 1s.

```
setTimeout ("timeAndDate( )",1000);
```

Figure 7.11: Method in JavaScript for Refreshing functions

7.4.2 Events Handler

With JavaScript SVG can realize many interactions for events handler and animations for ergonomic requirements by manipulating SVG DOM tree. The events Handlers build interaction between the users and the SVG application. Each event handler can awake an animation or a function.

SVG supports following events handler:

- ***onclick*** – click on a element
- ***onactivate*** – activate a element through certain pointer such as clicking mouse and keys.
- ***onmousedown*** – push down the left mouse key
- ***onmouseup*** – release the left mouse key
- ***onmouseover*** – the mouse pointer enters in area of a element

- ***onmousemove*** – move mouse pointer upon the area of a element
- ***onmouseout*** - mouse pointer quits from the area of a element
- ***onkeypress***/***onkeydown*** – push key
- ***onkeyup*** – release key
- ***onfocusin*** – the element becomes focus
- ***onfocusout*** – the element loses focus

With the rich functionalities of events handler SVG application allows the most events for interactions of user interface.

7.4.3 DOM Manipulation

With JavaScript the content of a SVG document can be accessed by DOM manipulation. Because all nodes in DOM tree can be manipulated it is possible to change the user interface elements and their attributes which are stored in a DOM tree.

With JavaScript the attributes of an element can be changed for animation effect. For example, if a machine is in working order, there is an alarm lamp showing green and when the machine is running abnormally the colour of the alarm lamp will be changed to red. In this case, the colour attribute is manipulated by JavaScript under predefined conditions.

The pseudo code in **Figure 7.12** illustrates two alarm lamps which colour are manipulated by JavaScript.

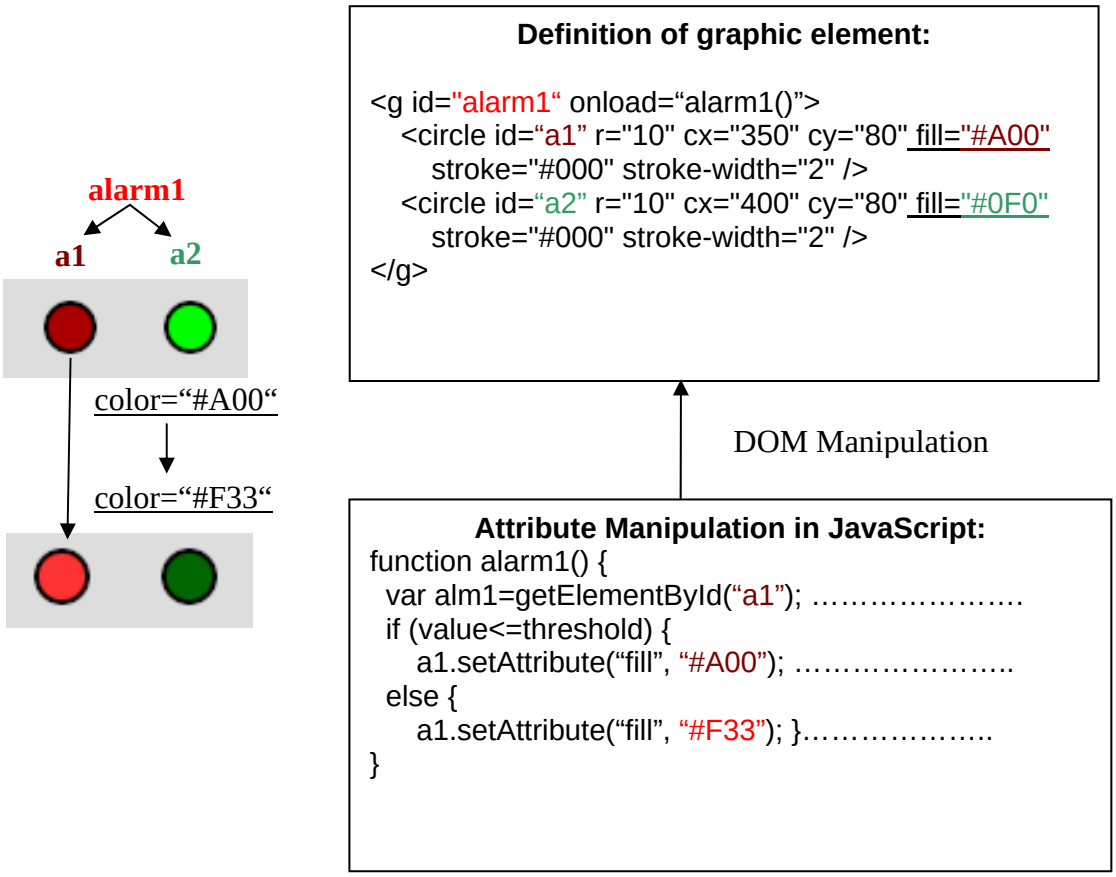


Figure 7.12: Example of DOM Manipulation with JavaScript to Change Colour attributes

In this example, the colours of alarm lamp for a machine are determined by the value of rotate speed read from server. In JavaScript, the if-condition in function *alarm1()* checks whether the value keeps under predefined threshold. If the value is not over threshold the left alarm represents the colour of deep red and the right light green. Contrarily, in the case that the value is over threshold, the colour of left alarm will change to light red and the right alarm turns to deep green.

Refreshing the function *alarm1()* frequently, the value of machine state can be read from server in time so that the colours of alarm lamp keep always in active status and duly change according to the current machine state.

7.5 XSLT and SVG

7.5.1 XSLT Overview

XSLT, XSL Transformation is template language expressed in XML syntax. XSL was developed to add style to an XML document and XSLT was designed to be more general and to allow the transformation of documents into any documents of XML type. XSLT allows users to transform XML documents into widely supported vocabularies, such as SVG.
An XSLT transform is formed in a list of templates which define the mapping between an object from source document such as element or attribute to an object in the output document.

There is a simple example which illustrates how the data in XML document is transformed into diagram in SVG format. Figure 7.13 is the simplest data in XML document. The stylesheet in Figure 7.14 below transforms the data into a histogram in SVG format. Because XSLT can produce any type of XML, with requirements of graphical representation of data documents the SVG format is a good choice for the users.

```
<data>
        <machine>35</machine>
        <machine>85</machine>
        <machine>55</machine>
</data>
```

Figure 7.13: Source Data in XML document

```
<xsl:stylesheet xmlns:xsl=http://www.w3.org/1999/XSL/Transform version="1.0">
  <xsl:template match="/">
   <svg xmlns:svg="http://www.w3.org/2000/svg" viewBox="0 0
     {10*(count(data/machine)+1)} 100">
    <xsl:for-each select="data/machine">
     <rect x="{10*position()}" y="{100- .}" width="10" height="{.}" fill="red"
     stroke="black"/>
    </xsl:for-each>
   </svg>
  </xsl:template>
</xsl:stylesheet>
```

Figure 7.14: XSL Document transforms XML data to SVG document

The result of the transformation shows in Figure. The diagram is generated in SVG format from XML data transformed by XSL.

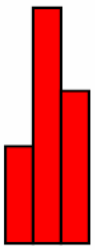


Figure 7.15: Result of Transformation from the data in **Figure 7.13** with XSL in **Figure7.14**

7.5.2 Generating SVG Graphs dynamically with XSLT

The two main benefits of SVG are that it is XML grammar and it is an open format. Because SVG is an XML grammar, it is easy to exchange data between documents with XML syntax and to be generated and manipulated in software components which use XML markup.

The technologies of SVG and XSLT make it possible to realize the data exchanging between database and graphical representation of data. Using XSLT is also an option for customizing the user interface. **Figure 7.16** illustrates the process for creating SVG through the use of XSLT and adding custom SVG content to different applications which are targeted for specific users or platforms. The benefit of the process is that the skin and feel could be easily internationalized for different languages or adapted for mobile devices.

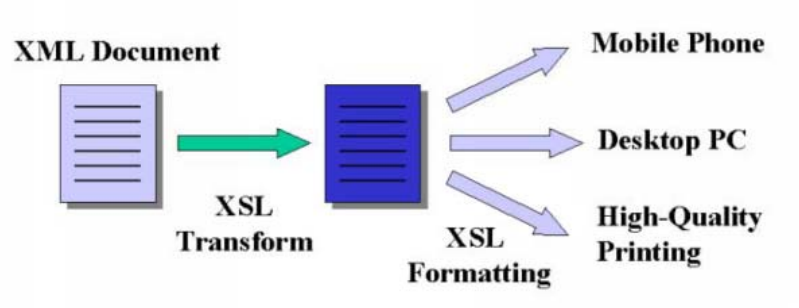


Figure 7.16: Generating SVG application from XSLT [Clark 1999]

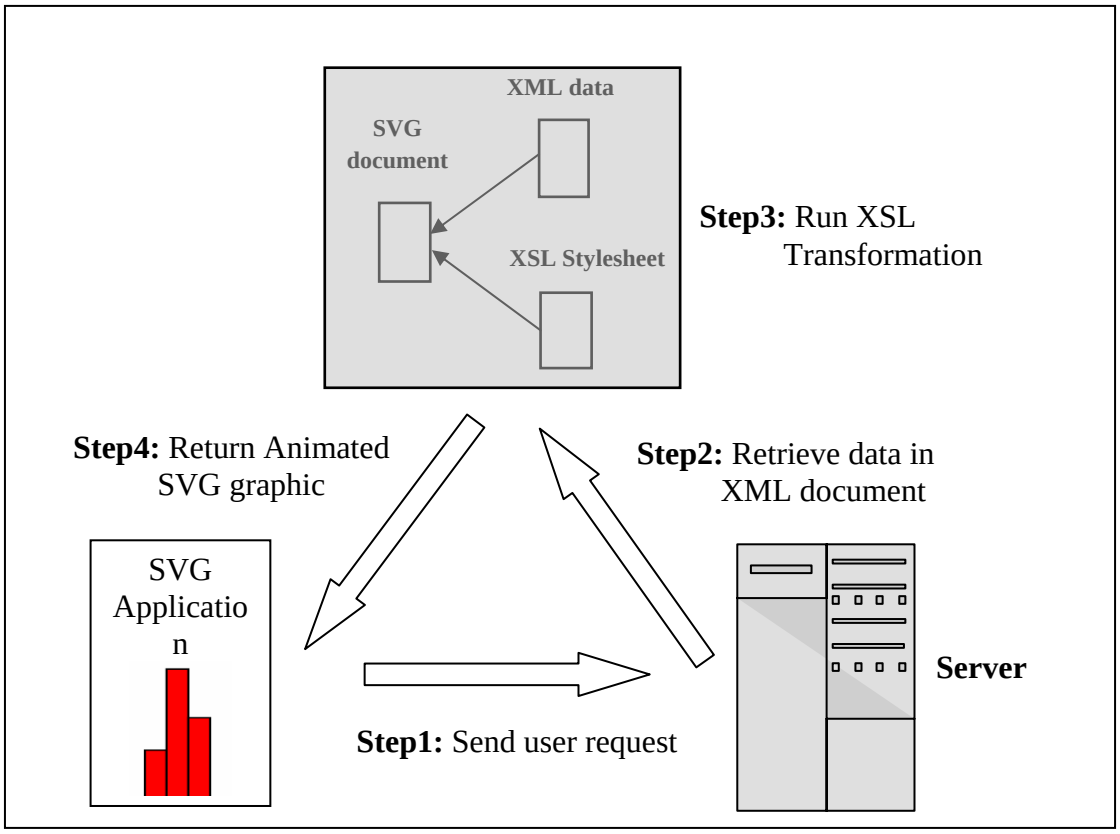


Figure 7.17: Generating SVG Application Step by Step

The Figure 7.17 shows how to dynamically generate a graphical representation of process values for process control system. Step by step, the data of process values is read from active server and converted to graphical representation by using XSL Transformation.

Step1:
The user gives request through user interface and the user request is sent to the central server.

Step2:
Following the user request the corresponding data information is retrieved from the server as an XML document.

Step3:
Using XSL transformation the XML document of data information is converted to SVG content which represents graphically the data information.

Step4:
The result of the finished transformation is an animated SVG document that is sent to the requestor. The requestor might be a human operator or another web service.

7.6 Transformation

The models built in previous chapters can be transformed into graphical objects by two methods using SVG technologies. The one is using XML parser and SVG DOM manipulations with JavaScript, and the other is using XSLT. In this section the approaches of transformation are explained with an example of temperature element. **Figure 7.18** shows an instance of temperature model in XML and the representation of the temperature element in SVG.

An Instance of Temperature

```
<Temperature unit="true"
   id="temp" controllable="false">
   <MAX>
        <Max>38</Max>
        <Warning>35</Warning>
   </MAX>
   <MIN>
        <Min>15</Min>
        <Warning>18</Warning>
   </MIN>
        <Actual>18</Actual>
</Temperature>
```

Temperature

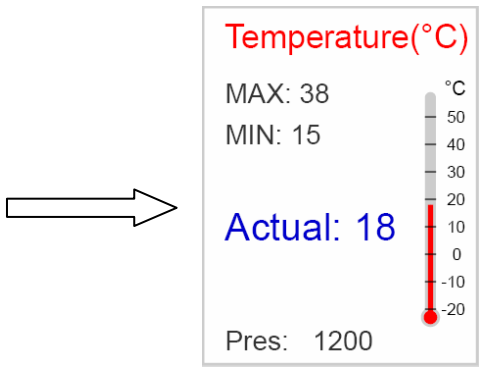


Figure 7.18: An Instance of Temperature Model and the Representation in SVG

Transformation using XML parser and SVG DOM manipulations

1. XML document for describing the instance of temperature model is parsed by using the method “parseXML()” in JavaScript.
2. After the element node “Temperature” is found by the XML parser the value of the node attribute “controllable” is read for determining whether temperature is controllable or not.
3. SVG element “use” is created by DOM manipulation with JavaScript. If “controllable” is false, the predefined design pattern for uncontrollable temperature element is called from resource and refers to “use” element by setting attribute “xlink”.
4. The new created SVG node “use” is appended to DOM tree so that a SVG presentation object is built.
5. To display the data such as “MAX”, “MIN” or “Actual” in temperature model on

the SVG temperature element, XML parser searches continuing the children nodes of element "temperature". There are two possibilities for the next step of transformation, if the values are not defined in template of design pattern, for each value a new SVG element must be created and appended to the DOM tree. If the values are already predefined in design pattern, no new SVG elements are required instead that the values parsed from XML can be simply assigned to according SVG elements in design pattern by DOM manipulation using JavaScript.

Workflow of this approach of transformation is illustrated in **Figure 7.19.**

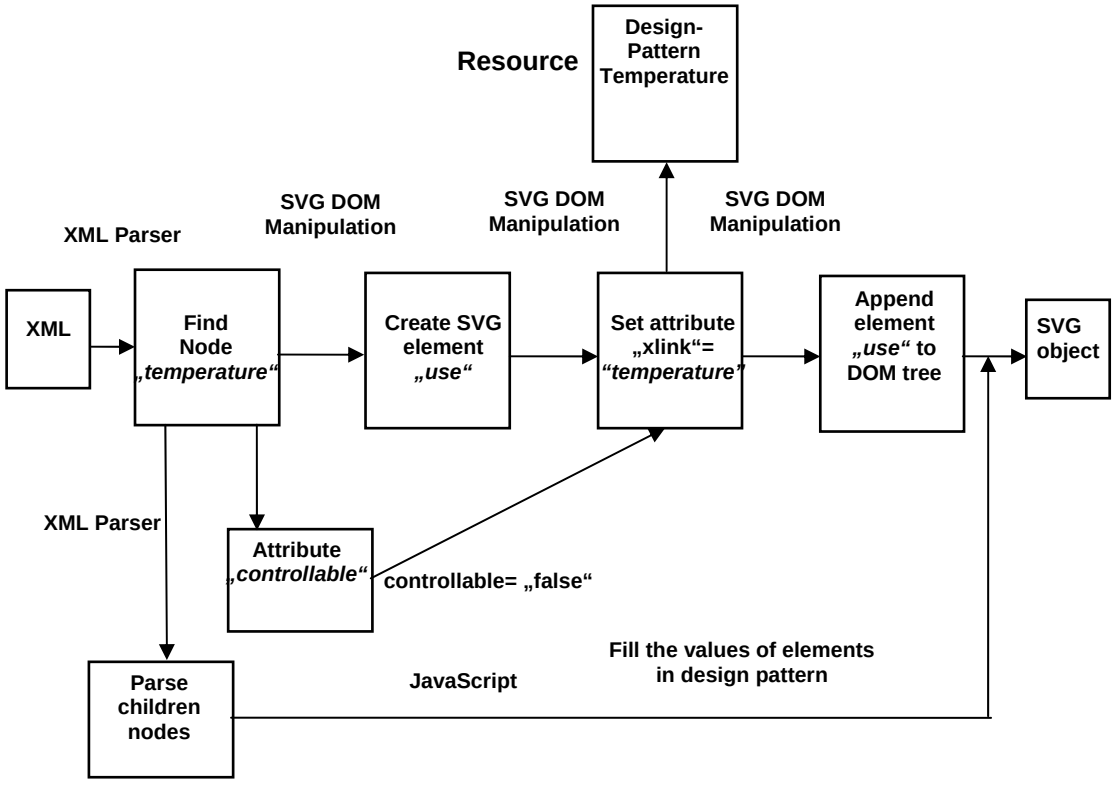


Figure 7.19: Transformation of Model to SVG presentation object using XML parser and DOM Manipulations

Transformation using XSLT

1. With the method of XPath the attribute value templates can be defined with SVG markup. For the values in the source document (here refers to the instance of temperature model) the value of the attribute in SVG is limited by XPath attribute value template.
2. XSL defines stylesheet and layout rules for output document.
3. Using specialized software the source document in XML format can be generated

into desired output SVG document. For example, Java programming language defines a standard API which allows transforms to XML content in the package "java.xml.transform". It is possible to apply stylesheet in all kinds of contexts by integrate transformation in applications. Therefore, a template of SVG stylesheet can be populated with actual information read from server side.

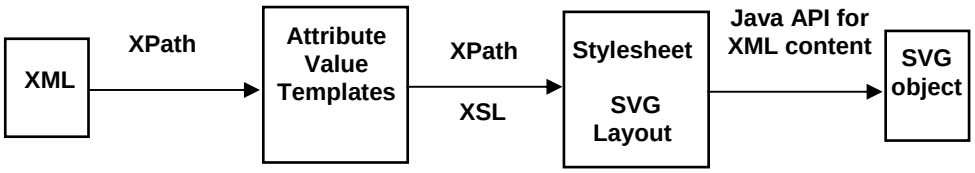


Figure 7.20: Transformation of XML Content to SVG Object using XSLT

7.7 Other SVG Technologies used for User Interface

With the rapid developing of technologies of web services, SVG is a popular mechanism used for graphical representation of data information. There are many SVG technologies in active development that can be adopted for user interface.

RCC: Rendering Custom Content

In [SVG 1.2] RCC is introduced as a XML-centric extension mechanism for defining the presentation and interactive behavior of particular elements described in a namespace. In essence, RCC is a binding mechanism that matches non-SVG XML elements to an SVG tree. Among the main focuses, the user interface work is handled as a primary use case for the usage of custom XML markup in an SVG document. Anyone can create an RCC library and make it available to others on the web.

The essential concept of RCC is to create templates for UI widgets in self defined namespaces. As long as the namespace for UI widgets is defined, the templates for graphical widgets can be created in the form of prefixed namespace. By using template of widget different instances of the UI widget classes can be created.

Multimedia User Interface

Audio:

Audio files can be embedded in SVG document. In SVG the audio element specifies an audio file which is to be rendered to provide synchronized audio. With the animation features of multimedia language SMIL the audio can be started and stopped at the appropriate times. The developer can also create graphical representation of control panels in SVG to control the audio.

Video:

Video file can be embedded in a SVG document similar to audio file. The difference is that the video element produces a rendered result, therefore the attributes for width, height and coordinate should be given.

Chapter 8

Evaluation

8.1 Evaluation of Prototype

In this work the prototype of graphical human-machine interface for process control system of manufacturing is built based on production model, machine models and process element model. For realizing the prototype for practical application there are still a lot of works to be made.

The product model built in this work is adapted to most products and semi-products in a production process. For practical application not all products in a production process must be modelled and sometimes the information of “For Process”, “To Process” and “Target” modelled in the product model is redundant. For some processes the product can be directly modelled as attributes in a machine model or process model.

The machine model built in this work is suitable for a simple tool machine employed in manufacturing, and the information modelled in this model can appropriately represent a machine. For the machines with complex functions the machine model offers not enough information. In the prototype the robots are defined as specialization of machine using machine model, but with increasing technologies the robots play a very important role in manufacturing and can complete very complex works, therefore a separate and detailed model for robots would be required.
The model of process elements built in this work consists of products and machines such that it can be rightly adopted for a process which produces some products or processes some products. In practical production processes there are also process elements which produce nothing or are carried out manually and therefore do not require a machine, for example the process of moving products to somewhere by human

workers. This will make necessary the introduction of a manual process step type to the model.

From the prototype a 2D graphical human-machine interface can be generated, but according to the property of complexity of manufacturing systems the graphical interface can not represent all information described in the models. The prototype contains most state information of machines and processes but little configuration information of machines. Some state information described in the prototype may not be measurable by instruments and can therefore not be treated as real-time values. For the realisation of productive applications the experts of control engineering and user interface developers should cooperatively develop the model and adapt the generated interface to their needs.

8.2 Comparison of XML Modelling Methods

Both of XML Schema and DTD are document type definition languages. In this work XML Schema is employed to model the information of production process in different problem domains. XML Schema works for the same purpose like DTD but owns many advantages comparing to DTD.

- Using specialized XML editor, for example XMLSpy [Altova 2005] adopted in this work, the developer can easily model the data in a graphical edit environment that enables the developer to better master entire hierarchical document structure and relationship between components.
- The mechanism of references makes the modelling work more easily. The frequently used data types or models can be at first separately modelled in an element and be referenced as the other components require. With this mechanism a lot of works can be economized.
- The developer can self define complex data types consisting of simple data types and other complex data types.
- The functionality of XML Schema is so robust that it can model fast all information which is required for a process control system.

8.3 Using SVG for Generating the User Interface

In this work SVG technique is adopted for implementing a user interface.

8.3.1 Advantages of an SVG User Interface

From the work the advantages of SVG for user interface are summarized:

- Because SVG is XML syntax following W3C recommendation, SVG can

cooperate with other XML documents using today's XML DOM technologies.

- SVG document can be manipulated by JavaScript which is an interpreter language so that the codes are executed without having to be complied.
- SVG elements can be flexibly manipulated using DOM
- SVG can be used in web application that adapts to the increasing requirement of web services.
- Using transformation the XML content can be transformed into SVG graphical objects.
- SVG offers many animation and filter effects which increase the visual effects of a user interface.
- At client-side the SVG graphics can receive the data from server-side and effectively display various diagrams.
- For large display the great advantage of SVG is that SVG is scalable graphic so that the graphical objects can be arbitrarily scaled without distortions.

8.3.2 Limitations of an SVG User Interface

Because SVG is relative new technique for implementing a user interface, there are still many limitations and problems to be resolved.

- There is no SVG editor that can at the same time not only realize WYSIWYG (What You See Is What You Get) but also support animations and interactions. Therefore, it brings inconveniences in the work of development.
- There are few resources of SVG widgets, at the most time the developer must manually write the codes for each widget.
- Not all browsers support SVG.
- Currently only Adobe SVG Plug-in supports full animations and interactions of SVG. There are many applications of SVG viewer but all of them have more or less limitations for displays or animations.

SVG supports user interface of web application rather than common software applications.

Chapter 9

Conclusions

Applications from the problem domain of process control engineering for manufacturing are characterized by a broad diversity in the form of information exchange which integrate the requirements of controller and processed into integrated information infrastructure. The emphases in this work are building human-machine interface of production process control system based on concept models, information-oriented function models and presentation models.

To build the human-machine interface the concept of analyzing problems from abstract to concrete is well adopted in this work. Because the essential of process control engineering is information exchange, the communication between human controller and machines is the basis of concept models. Information communication exists in form of data. The well-defined concept models in this work rely on the types of process values and state of machines. A good method used to build infrastructure of information is to define the data of process values with data types in XML Schema. Based on the in XML Schema document defined models the data can be specialized in XML format corresponding to different tasks. The task models built in this work prove that XML Schema can effectively define meta-model of information data even with complex data types. In our approach the generic model plays the role as a meta-model consisting of reasonable and manageable set of first-class modelling artefacts so that it can keep small and invariant. Comparing with meta-model the application-specific models are the lower layer of abstraction, they do not introduce new artefacts but are depicted using the meta-model artefacts.

In order to better model and manage the user interface for information exchange, the functions of data exchange through user interface are separated into data acquisition and data representation. Both of the parts can be modelled individually and connected by

interactive elements. The separation of functions benefits code separation and code reuse. Presentation models rely on the previous built task models, according to specific task only limited presentation objects are required. The data types in meta-model give us an early perception of which presentation objects may be possibly used.

In this work SVG technology is employed for implementing the model-based user interface. Cooperating with script language such as JavaScript the SVG application can achieve diverse interactions. With JavaScript the SVG application can access data from central server and display the data change dynamically on output devices in real time. For different output devices, such as screen or PDA different transformations are built for transforming data to presentation objects. JavaScript can also control data refreshing, in the other word, in what frequency the SVG application should read data from the server so that the data displayed on output devices can be automatic refreshed.

Chapter 10

Future Works

The approach of build model-based human machine interface described in this work is still in early phase of development. To put the task models and presentation models built in this work into application, there are some further works to be accomplished.

At first, for each specific application the XML document based on meta-model is specified that describes use cases of the process control system. For example, the meta-model of a process element built in this work contains optional number of machines which are applied in a process element. In specified XML document the definitive number of machines should be given. For each machine, the actual state information is chosen from the alternatives in meta-model.

With JavaScript the data can be accessed from central server, in the future work it can be considered, whether the server can send messages to the application as soon as the data are changing. The messages can awake the function in JavaScript which reads changed data in time.

Dialog design relying on ergonomics influences directly the usability of the user interface application. Dialog design for different platforms is a great part of the further works. There are entirely different dialog design guidelines depending on the size of display screen of output devices.

Today, the SVG technologies are actively developed by many organisations and at one time more and more hardware will support SVG graphics. The new SVG technologies will better support the SVG-based user interface. In the future, to shorten the time of development the commonly used UI widgets can be defined in pattern in open source, consequently, the developer can call a widget only by giving position and size without defining all basic shapes for a widget.

References:

[Albrecht 2002] Albrecht, H.: On Meta-Modeling for Communication in Operational Process Control Engineering, VDI Verlag, 2002.

[Altova 2005] XMLSpy: http://www.altova.com/, September 2005

[Ambler, Jeffries 2002] Ambler, S. W.; Jeffries, R.: Agile Modeling: Effective Practices for Extreme Programming and the Unified Process, Wiley, 2002.

[Ambler 2004] Ambler, S.W.: The Object Primer: Agile Model-Driven Development with UML 2.0, Cambridge University Press, 3 edition, 2004.

[Ambler 2005] Ambler, S.W.: User Interface Design Tips, Techniques and Principles, online-resource, http://www.ambysoft.com/essays/userInterfaceDesign.html, November 2005.

[Ballagas 2005] Ballagas, R.; Robs, M.; Scheridan, J.G.: Sweep and Point & Shoot: Phonecam-Based Interactions for Large Public Displays, CHI '05: Extended abstracts of the 2005 conference on Human factors and computing systems, ACM Press, Portland, Oregon, USA, April 2005.

[Bleek 1999] Bleek, W.-G.; Lippert, M.: Framework-basierte Anwendungsentwicklung, OBJEKT spectrum, 3/1999.

[Boeckler 1996] Boeckler, R.: Kraftwerksautomatisierung mit einem Standard-PLS-System, 7. VIK-Fachtagung, Technik, Darmstadt, November 1996.

[Clark 1999] J. Clark: XSL Transformations (XSLT) Version 1.0 Specification, W3C Recommendation, November 1999.

[Collomb 2005] Collomb, M.; Hascoet, M.; Baudisch, P.; and Lee, B.: Improving drag-and-drop on wall-size displays, In Proceedings of GI 2005, Victoria, BC, pp 25-32, May 2005.

[Constantine, Lockwood 1999] Constantine, L.L.; Lockwood, L.A.D.: Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design, Addison-Wesley Professional, 1999.

[Elliot 1991] Elliot, S.: Hierarchical State Machines in the Automation Process, article in website Automation.com, Resources for Industrial Automation, Process Control & Instrumentation, http://www.automation.com/sitepages/pid1522.php, 1991.

[Engineering Statistics Handbook] http://www.itl.nist.gov/div898/handbook/, last update, September 2005

[Fettes, Mansfield 2004] Fettes, A.; Mansfield, P.: SVG-Based User Interface Framework, SVG Open Conference 2004, September 2004.

[Flanagan 2002] Flanagan, D.: JavaScript – Das umfassend Referenzwerk, deutsche Übersetzung von Dorothea Reder & Harald Selke, O'Reilly Verlag, 2. Auflage, 2002.

[Froumentin, Hardy 2002] Froumentin, M.; Hardy, V.: Using XSLT and SVG together: a survey of case studies, SVG Open Conference 2002, July 2002.

[Jacko, Sears 2003] Jacko, J.A.; Sears, A.: The Human-Computer Interaction Handbook, Lawrence Erlbaum Associates, 2003.

[Jacobson 1992] Jacobson, I.: Object Oriented Software Engineering, Addison Wesley Professional, June 1992.

[Johannsen 1993] Johannsen, G. : Mensch-Maschine-Systeme, Springer Verlag, 1993.

[Jolif 2003] Jolif, C.: Bringing SVG Power to Java Applications, Sun Developer Network (SDN), http://java.sun.com/developer/technicalArticles/GUI/svg/, 2003.

[Kruschinski 1999] Kruschinski, V.: Layoutgestaltung grafischer Benutzungs-oberflächen, Spektrum Akademischer Verlag, 1999.

[Lauber 1996] Lauber, J.: Methode zur funktionalen Beschreibung und Analyse von Produktionsprozessen als Basis zur Realisierung leittechnischer Lösungen, Verlag Mainz, 1996.

[McBreen 1998] McBreen, P.: Using Use Cases for requirements capture, www.mcbreen.ab.ca/papers/UseCaseNotes.pdf, 1998.

[Microsoft 1999] Microsoft Windows User Experience, Microsoft Press, 1999.

[Mintert 1996] Mintert, S.: JavaScript Grundlagen und Einführung, Addison-Wesley, 1996.

[Norman 1988] Norman, Don: The Psychology of Everyday Things, Basic Books, New York, 1988.

[Oesterreich 1997] Oesterreich, B.: Objektorientierte Softwareentwicklung: Analyse und Design, Oldenbourg München 1997

[Peters 2002] Peters, B.: Ein Benutzungsschnittstellenmodell für die Prozessleittechnik, VDI Verlag, 2002.

[Phillips 2000] Phillips, L.A.: XML Modernes Daten- und Dokumentenmanagement, Autorisierte Übersetzung der amerikanischen Originalausgabe Special Edition Using XML, Markt+Technik Verlag, 2002.

[Pohlmann 2003] Pohlmann, R.: SVG online Tutorial, http://svg.tutorial.aptico.de/, last update, 2003.

[Qiu 2004] Qiu, X.: Building Desktop Applications with Web Service in a Message-based MVC Paradigm, IEEE International Conference on Web Services, San Diego, California, July 2004

[Raymond] Raymond, M.: Interaction Design System Use of XML, in proceeding XML Conference & Exposition 2001, 2001.

[Reuth, Künzer 2001] Reuth, R.; Künzer, A.; Boldt, T; Schmidt, L.; Luczak, H.; Murrenhoff, H.: Modellbasierte Gestaltung einer multimodalen Benutzungsschnittstelle zur Unterstützung von Greif- und Spannprozessen beim 3D-Laserschweißen, 4. Berliner Werkstatt Mensche-Maschine-Systeme, ZMMS Spektrum, Band 13, p.55-70, Oktober, 2001.

[Robertson 2005] Robertson, G.; Czerwinski, M.; Baudisch, P.; Meyers, B.; Robbins, D.; Smith, G.; Tan, D.: Large Display User Experience, In proceeding IEEE Computer Graphics & Application, Special Issue on Large Displays, pp. 44-51, August 2005.

[Rotard 2004] Rotard, M.: XML-basierte Graphikstandards für interaktive Systeme: Erleichterter Zugang zu graphischer Information, Virtuelle und Erweiterte Realität , p.373-384. Shaker Verlag, 2004.

[Schlegel 2003] Schlegel, T.: Entwurf und Erprobung eines software-gestützten Verfahrens zur Anwendung software-ergonomischer Methoden in den frühen Phasen der Anwendungsentwicklung, Diplomarbeit Nr. 1985, Fakultät Informatik, Uni. Stuttgart, 2003.

[Schlick 1999] Schlick, C.: Modellbasierte Gestaltung der Benutzungsschnittstelle autonomer Produktionszellen, Shaker Verlag, Aachen, 1999.
[Shneiderman 1998] Shneiderman, B.: Designing the User Interface, Strategies for Effective Human-Computer Interaction, 1998.

[SVG1.1] Scalable Vector Graphics (SVG) 1.1 Specification, W3C Recommendation, January 2003.

[UML 2005] UML: The Unified Modeling Language: Superstructure, http://www.omg.org/uml, August 2001.

[Wahl 1999] Wahl, M.: Systematische Entwicklung nutzergerechter Maschinen-bedienoberflächen, Verlag Universität Kaiserslautern, 1999.

[Wallace] Wallace, G.; Tools and Applications for Large-Scale Display Walls, IEEE Computer Graphics and Applications, vol. 25, no. 4, pp. 24-33, August 2005.

[Zühlke 2001] Zühlke, D.: Bedienung komplexer Maschinen – heute, morgen und übermorgen, 4. Berliner Werkstatt Mensche-Maschine-Systeme, ZMMS Spektrum, Band 13, p.42-54, Oktober, 2001.