

Software Development

*Manual for the Planning, Realization, and
Installation of DP Systems*

Wolfgang End, Horst Gotthardt, and Rolf Winkelmann

Siemens AG, Munich

Translated by

A. J. and B. D. Tebby

Software Development

*Manual for the Planning, Realization, and
Installation of DP Systems*

Wolfgang End, Horst Gotthardt, and Rolf Winkelmann

Siemens AG, Munich

Translated by

A. J. and B. D. Tebby

A Wiley Heyden Publication

JOHN WILEY & SONS

Chichester · New York · Brisbane · Toronto · Singapore

First published under the title Softwareentwicklung:
Leitfaden für Planung, Realisierung und Einführung
von DV-Verfahren. 3 überarbeitete und ergänzte Auflage
Siemens Aktiengesellschaft (1980)

Copyright © 1987 by Wiley Heyden Ltd

All rights reserved

No part of this book may be reproduced by any means, nor
transmitted, nor translated into a machine language without the
written permission of the publisher

Library of Congress Cataloging in Publication Data:

End, Wolfgang.

Software development.

Translation of Softwareentwicklung.

A Wiley Heyden publication

Bibliography: p

Includes index

I. Computer programming management. I. Gotthardt,
Horst. II. Winkelmann, Rolf. III. Title.

QA76.6.E513 .98 001.64'25 82-.7448

ISBN 0 471 26238 2 (U.S.)

British Library Cataloguing in Publication Data:

End, Wolfgang

Software development.

I. Computer programming management

I. Title. II. Gotthardt, Horst. III. Winkelmann,
Rolf

001.64'25 QA76

ISBN 0 471 26238 2

Preface

The aims of all those who are engaged individually or as part of a team in design, development or management, or of teams, who are involved in planning, implementing, releasing or supporting data-processing systems, must be to keep the costs of the development and maintenance of the software as low as possible and to improve the quality of the software products. This manual is an aid for all these groups as they: follow a systematic approach to planning and implementation; monitor the individual work stages from conception to release; select and use suitable aids for planning and implementation.

The book is based on the experiences of several data-processing departments and the training personnel of the company. It is intended to be a working and training document and can also be used as a reference book. The appendix on 'Check Lists for Software Development', which is a summary of all the check lists, is intended to be a further aid for those involved in software development.

Siemens Aktiengesellschaft

Contents

Preface	ix
Introduction	
Software engineering in the development of software	
PART I PHASES OF SOFTWARE DEVELOPMENT	
1 Project proposal phase	1
1.1 Definition of the problem and requirements; formulation of the objectives	1
1.2 Approval for preliminary study	14
1.3 The organization of each phase	20
1.4 Monitoring each phase	25
1.5 The preliminary study	28
1.6 General network diagram and development expenditure	32
1.7 The development submission	32
1.8 Completion of each phase	35
1.9 Milestone	37
2 Planning phase I	4
2.1 The organization of the phase	4
2.2 Monitoring the phase	10
2.3 Dissemination of information	41
2.4 The idealized scheme	41
2.5 The status study	44
2.6 The status analysis	52
2.7 The general functional design	56
2.8 Approval of the general functional design	60
2.9 Completion of the phase	61
2.10 Milestone	62

3	Planning phase II	63
3.1	The organization of the phase	63
3.2	Monitoring the phase	65
3.3	Applications for amendments	66
3.4	The functional specification	66
3.5	Approval of the functional specification	109
3.6	Provisional testing timetable	109
3.7	Determining the responsibility for support	111
3.8	The training plan	111
3.9	Completion of the phase	112
3.10	Milestone	113
4	Realization phase I	114
4.1	The organization of the phase	114
4.2	Monitoring the phase	115
4.3	Conventions for programming and testing	117
4.4	The detailed system design	120
4.5	Converting software	136
4.6	Programming and coding	136
4.7	Testing	145
4.8	Creating reference data	154
4.9	Documentation for a system or program	155
4.10	Operating manual	157
4.11	Operating instructions for users	165
4.12	Completion of the phase	167
4.13	Milestone	167
5	Realization phase II	169
5.1	The organization of the phase	169
5.2	Monitoring the phase	169
5.3	Training applications and data-processing personnel	170
5.4	Adapting the organization	173
5.5	Preparation for the trial period	174
5.6	Determining the support method	175
5.7	The trial period	176
5.8	Completion of the phase	177
5.9	Milestone	177
5.10	System release	179
6	Installation phase	183
6.1	The organization of the phase	184
6.2	Monitoring the phase and assessing the system	185

6.3	Running the system	186
6.4	Supporting the system	188
6.5	Milestone	192

PART II PLANNING AND IMPLEMENTATION AIDS

7	Planning, managing, and monitoring data-processing projects	195
7.1	Data-processing framework plan	196
7.2	Project plan	198
7.3	Project partitioning and development summaries	200
7.4	Assessment of time and costs	203
7.5	Records and project assessment	205
7.6	Network diagram technique	208
7.7	MPM network diagram	208
7.8	Creating a network diagram	211
7.9	Scheduling	212
7.10	Costing and personnel planning	216
8	Estimating the duration of a project	217
8.1	Estimating methods	217
8.2	Estimating the duration of the functional solution	221
8.3	Estimating the duration of the system solution	222
8.4	Procedure for estimating the duration of a project	226
9	Cost-effectiveness of data-processing systems	228
9.1	Successful rationalization	228
9.2	Calculating cost-effectiveness	230
9.3	Direct cost-effectiveness	230
9.4	Indirect cost-effectiveness	231
9.5	Methods for evaluating cost-effectiveness	234
9.6	Determining the marginal return on capital employed	236
10	Data preparation	238
10.1	Processing stages for data preparation	238
10.2	Methods of data preparation	239
10.3	Data media	244
10.4	Data preparation equipment	245
10.5	The procedure for selecting the data preparation method and equipment	247
11	Programming standards	253
11.1	Programming languages	253

11.2	Structuring	253
11.3	Formation of structure blocks	254
11.4	COBOL conventions	256
11.5	Structure of a COBOL program	258
11.6	Recommendations	258
11.7	Data dictionary	260
12	Estimating run times and optimizing block sizes	263
12.1	Processing time	264
12.2	Input and output time	265
12.3	Time for external operations	282
12.4	Waiting time in the system	282
12.5	Total run time	282
13	Minimizing test cases	284
13.1	The principle	284
13.2	Procedure using structograms	285
14	Software engineering	289
14.1	Design	290
14.2	Realization	297
14.3	Measurements	297
14.4	Testing	298
14.5	Documentation	301
14.6	Control and administration	302
	Check lists for software development	303
	Bibliography	331
	Glossary	333
	Index	339

Introduction

Background

Data processing makes a significant contribution to the execution of tasks in all fields, from research and development through production to business control and management. In addition it enables people to master new types of problems, for example in research when establishing optimum values by simulation, or in management when considering the merits of alternative decisions by calculating their effects on the financial results of the company. As a result of the introduction of data-processing into flight safety systems and medicine, human lives may depend on the quality of the software.

The increasing penetration of data-processing into all fields as well as the trend towards developing integrated solutions, means that the development, use, and maintenance of computer programs require a large number of personnel, considerable machine time and extensive hardware investment. In comparison with other business expenditure the resulting costs have reached a considerably high level. In 1977 personnel costs in the data-processing field in the Federal Republic of Germany reached £3000 million and the annual expenditure in rental, depreciation, and maintenance of hardware cost about £2000 million. Analyses of completed software projects show that 'planned' schedules and personnel and computing costs have been exceeded especially in the projects which lacked adequate and satisfactory project planning, monitoring, and control. A 10% saving by rationalizing software development and by improving the quality of the software products would result in annual savings far in excess of £250 million (only testing time and reruns are considered in the case of the hardware).

Aim of the Manual

Economic development of software requires a clear conception of the objective, procedure, methods, and techniques to be employed in the planning and development. In particular:

- (a) software developments should proceed from conception to release in

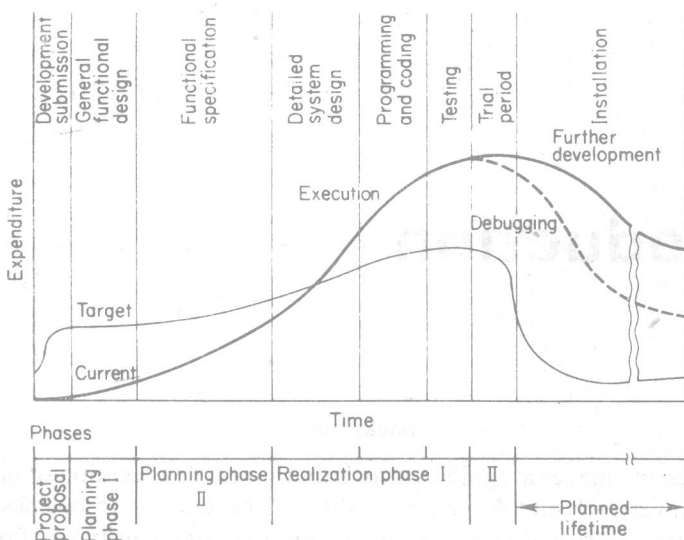


Diagram of expenditure during the development and support of programs

- predefined phases with breaks for reviews. In this way one phase must be completed before the next phase is started; and
- (b) interdisciplinary groups appropriate to the complexity of the tasks should be set up to determine how the project should be approached. A steering committee or manager examines the results, approves the start of the next phase and coordinates this to fit in with other development projects.

The method of developing software illustrated in this manual and the planning and development aids described can be used in industry, commerce, and finance as well as by government departments and public administration. There is no difference in dealing with developments in the fields of business systems, manufacturing or science and technology. It is also possible to use this approach when undertaking problems which will not be dealt with by computer.

The aim is to run a project on the lines represented by the curve labelled 'Target' (see diagram). Typical faults are inadequate planning at the start of a project, insufficiently systematic progress during the project (for example, lack of reviews) and unsatisfactory realization (techniques, methods, and standards). These usually lead to overexpenditure during installation (testing and trial periods) and to further development and debugging during the planned lifetime (support phase).

The proposals put forward in this manual can and should be used as established conventions by all those developing software systems, thereby improving the quality of their software products.

Limitation

The remarks in this manual apply chiefly to the development of single user or general purpose applications software.

Structure of the Manual

The manual is divided into two parts.

Part I *Phases of software development*

- Project proposal phase

- Planning phase I

- Planning phase II

- Realization phase I

- Realization phase II

- Installation phase

The features of this structure are: (a) strict division into individual development stages with clearly defined breaks for reviews; (b) the main planning effort occurs in the early phases of the software development.

Part II *Planning and implementation aids*

- Planning, managing, and monitoring data-processing projects

- Estimating the duration of a project

- Cost-effectiveness of data processing systems

- Data preparation

- Programming standards

- Estimating run times and optimizing block sizes

- Minimizing test cases

- Software engineering

The planning and implementation aids and working methods will include those planning, developing or using software how to solve or approach the solution of problems which may arise in the individual phases of a software development.

Using the Manual

Standardized network diagram

At the beginning of each phase, a standardized network diagram (MPM network diagram in vertical format) illustrates how each task fits into the whole. The standardized network diagram can be simplified or expanded for particular phases depending on the size and complexity of the project.

Tasks

The various tasks within each phase are dealt with under the same heading in the individual chapters of the manual. The tasks are described and methods and techniques are explained.

Check lists

Check lists are either incorporated in the task description or summarized at the

end of each section. They are a means of checking whether the basic requirements of a task have been taken into consideration.

All the check lists are summarized in the section 'Check Lists for Software Development'.

Personnel-task matrix

The personnel-task matrix indicates the assignment of personnel to tasks and is included under the heading 'The Organization of the Phase'.

Glossary

The Siemens software products mentioned in the manual and abbreviations used are listed and explained in the Glossary.

Examples

Through all stages from the formulation of the problem and objectives, to the programming itself, an example is given of each step in software development. The summary on the following pages shows how the chapters are interrelated.

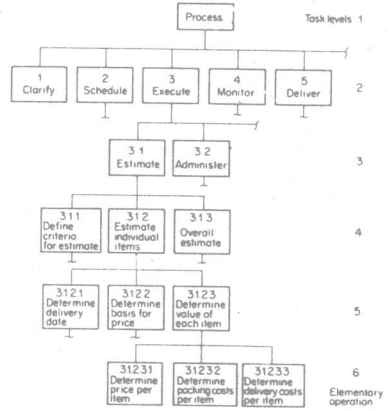
1

The objectives should be formulated on the basis of the requirement (Figure 1.8, page 30)



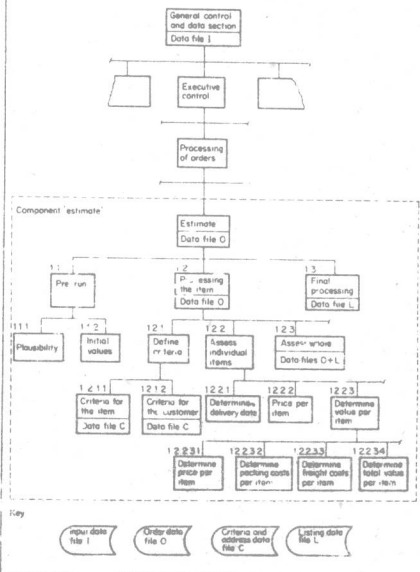
2

After the objectives have been defined, the status study and analysis are carried out and a structure diagram of the operations is drawn up (Figure 2.4, page 48)



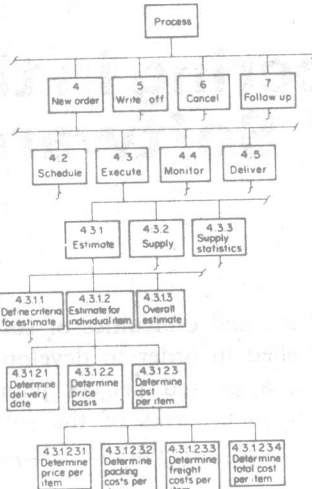
5

The individual components including the data flow are set out in detail in the functional specification (Figure 3.17, page 98)



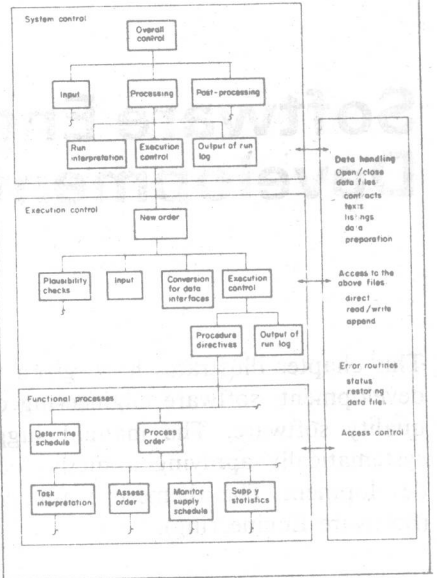
3

The structure diagram of the procedures is drawn up when the functions are defined, based on the structure diagram of the operations (Figure 3.3, page 69)



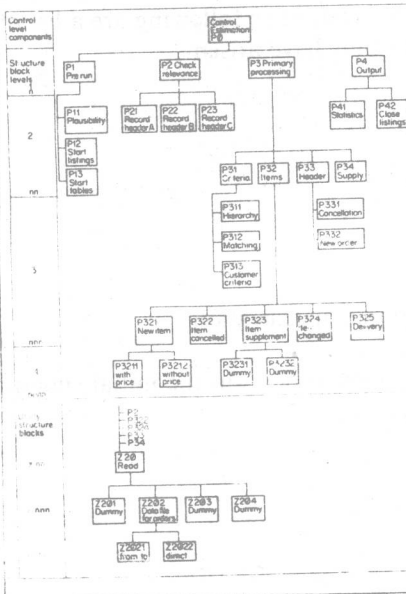
4

Design of the system structure (Figure 3.10, page 92)



6

Plan of the program organization in the specification for the 'estimation' component (Figure 4.3, page 122)



Software Engineering in the Development of Software

This chapter illustrates how, given the problems and difficulties of software development, software engineering can be applied in order to develop high quality software. The manual suggests using a phased project plan and systematically applying a methodology to the main tasks of the software development. This theme is examined in greater depth in Part II, Chapter 14 (Software Engineering).

Problems and Difficulties in Software Development

The economic importance of software stands in direct contrast to the problems and difficulties which until now have arisen in the development, implementation, maintenance, and support of software products. These problems can be grouped under three headings: quality, costs, and time-scale. The following are a list of criticisms and a list of reasons why the problems have arisen.

Criticisms

Quality:

- Unreliability
- Difficulty in use
- Incompatibility
- Non-portability
- Frequent failure to carry out the functional specification

Costs:

- Considerable overspending of the development costs in the initial stages
- Excessive running and maintenance costs
- Poor machine utilization

Time-scale:

- Overrunning the schedule
- Excessive development time
- Delayed documentation

Causes

The problems may have been caused by some of the following factors:

- (a) inadequate formulation of the requirements by the user;
- (b) inaccurate estimation of cost and time (e.g. failure to learn from past experience);
- (c) unsatisfactory development methods;
- (d) insufficient application of standard solutions;
- (e) inappropriate application of software techniques, methods and tools;
- (f) non-existent or inadequate testing methodology;
- (g) inadequate management control;
- (h) insufficient attention to documentation;
- (i) incompetence of the personnel involved in the development.

Objectives of Software Engineering

High quality software should be developed by using the principles of software engineering.

The quality of the software is determined by several criteria, the most important of which are described below. In practice these criteria are taken as the hallmarks of quality software.

Functional performance

The critical measure of a software product is the extent to which the functional specification is fulfilled. The temporary omission of some functions is a means frequently used so that a 'partial solution' can be produced to meet an impending deadline.

Ease of use (User friendliness)

This includes all features which enable the operations service, data preparation service, and the user of the system to utilize the software product easily and efficiently.

Examples include: a language suited to the user; ease of learning; and robustness when mis-used.

Efficiency

Efficiency is mainly determined by the size and speed of a program or system. For a long time these were thought to be the only important considerations in the development of software.