

---

---

**Information Technology — Software  
Engineering Environment Services**

*Technologies de l'information — Services d'environnement  
en ingénierie du logiciel*

**iTeh STANDARD PREVIEW  
(standards.iteh.ai)**

[ISO/IEC 15940:2006](https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006)

<https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006>

**PDF disclaimer**

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

**iTeh STANDARD PREVIEW**  
**(standards.iteh.ai)**

ISO/IEC 15940:2006

<https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006>

© ISO/IEC 2006

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office  
Case postale 56 • CH-1211 Geneva 20  
Tel. + 41 22 749 01 11  
Fax + 41 22 749 09 47  
E-mail [copyright@iso.org](mailto:copyright@iso.org)  
Web [www.iso.org](http://www.iso.org)

Published in Switzerland

# Contents

Page

<b>1</b>	<b>Scope .....</b>	<b>1</b>
<b>2</b>	<b>Terms and definitions.....</b>	<b>1</b>
<b>3</b>	<b>Abbreviated terms .....</b>	<b>2</b>
<b>4</b>	<b>Reference Model for SEE services.....</b>	<b>2</b>
<b>4.1</b>	<b>Categories of SEE services .....</b>	<b>2</b>
<b>4.2</b>	<b>Structure of service description.....</b>	<b>3</b>
<b>4.3</b>	<b>Reference model.....</b>	<b>3</b>
<b>5</b>	<b>Software engineering services.....</b>	<b>5</b>
<b>5.1</b>	<b>Software requirements engineering service.....</b>	<b>5</b>
<b>5.2</b>	<b>Software reverse engineering service.....</b>	<b>6</b>
<b>5.3</b>	<b>Software re-engineering service .....</b>	<b>6</b>
<b>5.4</b>	<b>Software prototyping service .....</b>	<b>6</b>
<b>5.5</b>	<b>Software modelling service .....</b>	<b>7</b>
<b>5.6</b>	<b>Software simulation service .....</b>	<b>7</b>
<b>5.7</b>	<b>Software design service.....</b>	<b>7</b>
<b>5.8</b>	<b>Component based software generation service.....</b>	<b>8</b>
<b>5.9</b>	<b>Source code generation service.....</b>	<b>8</b>
<b>5.10</b>	<b>Compilation service.....</b>	<b>8</b>
<b>5.11</b>	<b>Debugging service.....</b>	<b>9</b>
<b>5.12</b>	<b>Software static/dynamic analysis service.....</b>	<b>9</b>
<b>5.13</b>	<b>Software testing service .....</b>	<b>10</b>
<b>5.14</b>	<b>Software verification service.....</b>	<b>10</b>
<b>5.15</b>	<b>Software integration service.....</b>	<b>10</b>
<b>6</b>	<b>Technical management services.....</b>	<b>11</b>
<b>6.1</b>	<b>Configuration management service.....</b>	<b>11</b>
<b>6.2</b>	<b>Change management service .....</b>	<b>12</b>
<b>6.3</b>	<b>SEE repository management service .....</b>	<b>12</b>
<b>6.4</b>	<b>Reuse Management service.....</b>	<b>12</b>
<b>6.5</b>	<b>Measurement and analysis service.....</b>	<b>13</b>
<b>6.6</b>	<b>Quality assurance service .....</b>	<b>13</b>
<b>6.7</b>	<b>Audit service .....</b>	<b>14</b>
<b>6.8</b>	<b>Software traceability service .....</b>	<b>14</b>
<b>6.9</b>	<b>Documentation service .....</b>	<b>14</b>
<b>6.10</b>	<b>Review service support.....</b>	<b>15</b>
<b>7</b>	<b>Project management services .....</b>	<b>15</b>
<b>7.1</b>	<b>Project planning service .....</b>	<b>15</b>
<b>7.2</b>	<b>Project estimation service .....</b>	<b>16</b>
<b>7.3</b>	<b>Project risk management service.....</b>	<b>16</b>
<b>7.4</b>	<b>Project monitoring and scheduling service.....</b>	<b>17</b>
<b>7.5</b>	<b>Project evaluation service .....</b>	<b>17</b>
<b>8</b>	<b>Process management services .....</b>	<b>17</b>
<b>8.1</b>	<b>Process definition service .....</b>	<b>18</b>
<b>8.2</b>	<b>Process library service .....</b>	<b>18</b>
<b>8.3</b>	<b>Process initiation service .....</b>	<b>18</b>
<b>8.4</b>	<b>Process usage service .....</b>	<b>19</b>
<b>8.5</b>	<b>Process monitoring service.....</b>	<b>19</b>
<b>8.6</b>	<b>Process improvement support service .....</b>	<b>19</b>
<b>8.7</b>	<b>Process documentation service.....</b>	<b>20</b>

9	SEE support services .....	20
9.1	SEE common support service .....	20
9.2	SEE publishing service .....	21
9.3	SEE cooperative work support service .....	21
9.4	SEE user communication support service.....	21
9.5	SEE administration service.....	22
9.6	SEE policy enforcement service.....	22
9.7	SEE data/information mining service .....	23
10	SEE Infrastructure services .....	23
10.1	SEE infrastructure management service.....	23
10.2	SEE information sharing service.....	24
10.3	SEE repository service .....	24
10.4	SEE Operating System service.....	25
Annex A (informative) Exemplary automated support for the SEE Services .....		26
Annex B (informative) Services mapped on to ISO/IEC 12207 activities .....		36
Annex C (informative) Application of this International Standard.....		47
C.1	General .....	47
C.2	Users/software engineers .....	47
C.3	Tool and SEE suppliers .....	47
C.4	Acquirers.....	47
C.5	Software engineering educators .....	47
C.6	Software engineering consultants .....	48
Annex D (informative) Illustrated image of SEE concept .....		49
Annex E (informative) WG4 Standards Architecture (from WG4 business plan).....		50
Annex F (informative) Bibliography .....		51

ISO/IEC 15940:2006  
<https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006>

## Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 15940 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and system engineering*.

ITC STANDARD PREVIEW  
(standards.iteh.ai)

[ISO/IEC 15940:2006](https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006)

<https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006>

## Introduction

Software engineering environments, or “SEEs” refer to a collection of software services, partially or fully automated by software tools, that are used to support the execution of human activities in software engineering.

These activities are usually carried out within a software development/maintenance project, and cover such areas as the specification, development, re-engineering or maintenance of software-based systems. ISO/IEC 12207 describes in a comprehensive manner all of the processes, activities and tasks performed during the software life cycle.

The term "Software Engineering Environment" may cover several situations; from the mere juxtaposition of a few tools running on the same operating system, to the fully integrated environment, able to handle, monitor, and even control all the data, processes, and activities in the software life cycle. A SEE provides support to human activities through a series of services that describe the capabilities of the environment. The software process supported by a SEE becomes an assisted or automated software process. This International Standard describes SEE services and relates them to ISO/IEC 12207:1995, ISO/IEC 12207:1995/Amd.1:2002 and ISO/IEC 12207:1995/Amd.2:2004 in a manner applicable to a range of organizations. In defining a life cycle process for an organization, the user needs to find the appropriate level of automation provided by a software engineering environment. This may result in establishing a new SEE or improving an existing one.

**iTeh STANDARD PREVIEW**

Through the automation of activities, either partially or fully, the SEE provides benefits to an organization through reduced cost (higher productivity), improved management and from the higher product quality that can result. For example, the automation of repetitive activities such as the execution of test cases provides not only productivity gains, but can also help to ensure completeness and consistency in the testing activities.

<https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5d129-0386/iso-iec-15940-2006>

This International Standard defines the SEE services conceptually in a reference model that can be adapted to any SEEs to automate one or more software engineering activities.

For a user interested in a specific process, this International Standard describes the relationship between given software engineering processes, the software engineering services, and the corresponding exemplary software engineering tools.

The suite of SEE services described supports the process definitions in ISO/IEC 12207. The purpose is to define a set of SEE Services that are compatible with ISO/IEC 12207:1995, ISO/IEC 12207:1995/Amd.1:2002 and ISO/IEC 12207:1995/Amd.2:2004, and that can be used either as a general reference, or to define an automated software process.

# Information Technology — Software Engineering Environment Services

## 1 Scope

This International Standard provides a description of SEE services that supports all of the software life cycle processes defined in ISO/IEC 12207.

The services are intended as a complete set and can be used in any software engineering development or support organization where there is a need to select one or more SEE services. Such an organization may or may not have software projects that use the ISO/IEC 12207 process framework.

A reference model for SEE Services is provided within this International Standard. This reference model has been produced starting from references [8] and [9]. This document was produced using material originally published by the Software Engineering Institute (Carnegie Mellon University, USA), NIST and ECMA, which finally resulted in a joint effort from ECMA and NIST indicating a broad consensus at the time of publication. In addition to this background process, structure from ISO/IEC 12207:1995, ISO/IEC 12207:1995/Amd.1:2002, and ISO/IEC 12207:1995/Amd.2:2004 has been used as a baseline.

## 2 Terms and definitions

For the purposes of this document, the following terms and definitions apply. From other International Standards.

### 2.1

#### life cycle model

framework containing the processes, activities and tasks involved in the development operation and maintenance of a software product, spanning the life of the system from the definition of its requirements to the termination of its use

[ISO/IEC 12207:1995].

### 2.2

#### CASE tool

software product that can assist software engineers by providing automated support for software life-cycle activities as defined in ISO/IEC 12207:1995.

[ISO/IEC 14102:1995].

### 2.3

#### organization

group of people and facilities with an arrangement of responsibilities, authorities and relationships

[ISO 9000:2005].

**2.4  
Software Engineering Environment  
SEE**

provides automated services for the engineering of software systems and related domains (e.g., project management, process management, etc.)

NOTE It includes the platform, system software, utilities, and CASE tools installed.

**2.5  
SEE Service**

consists of one or more service operations to support life cycle activities for the SEE

NOTE A SEE Service supplier provides a SEE Service for a SEE Service acquirer.

**2.6  
automated or assisted software process**

software process that is performed either fully or partially supported by CASE tools

**2.7  
actor**

organization or CASE tool that supplies and/or acquires SEE Services

**2.8  
operation**

action needed to perform an Activity

NOTE One or more operations are necessary to execute an Activity. An operation may consist of other operations.

**iTeh STANDARD PREVIEW**  
**(standards.iteh.ai)**

**2.9  
SEE Service acquirer**

actor that acquires a SEE Service

[ISO/IEC 15940:2006](https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006)

<https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006>

**2.10  
SEE Service supplier**

actor that supplies a SEE Service

**3 Abbreviated terms**

CASE – Computer Aided Software Engineering

SEE – Software Engineering Environment

**4 Reference Model for SEE services**

**4.1 Categories of SEE services**

This (draft) International Standard provides a reference model for SEE services. As a reference model, this (draft) International Standard uses a set of conceptual descriptions to describe each service used in a software engineering environment. The “conceptual description” indicates that the description is from a reference viewpoint, and does not deal with any specific implementation. The description is therefore general and does not assume any specific application domain, life cycle model, or tool in a project. In this way, this (draft) International Standard can be applied to any defined organizational environment.



An actual environment is one that is built from a reference model containing conceptual descriptions. Therefore, an actual description of a specific environment would reflect a particular activity with its tools and standards. The services described in this (draft) International Standard are grouped into six categories that reflect broad functional activities within a typical software engineering organization. The six categories are:

- Software engineering services (e.g., System Design, Software Modelling, Simulation);
- Technical management services (e.g., Reuse, Configuration management);
- Project management services (e.g., Estimation, Project monitoring);
- Process management services (e.g., Process Monitoring, Process improvement);
- SEE Support services (e.g., Publishing, Policy enforcement);
- SEE infrastructure services (e.g., Repository, Communication, OS services).

#### 4.2 Structure of service description

Each service is defined under two headings:

- Service Concept, to provide a description of the service in terms that are not related to a specific implementation.
- Service Operations, to list those operations that may be included in a service. These lists of operational capabilities represent, in most cases, primary services only and are not intended to be complete.

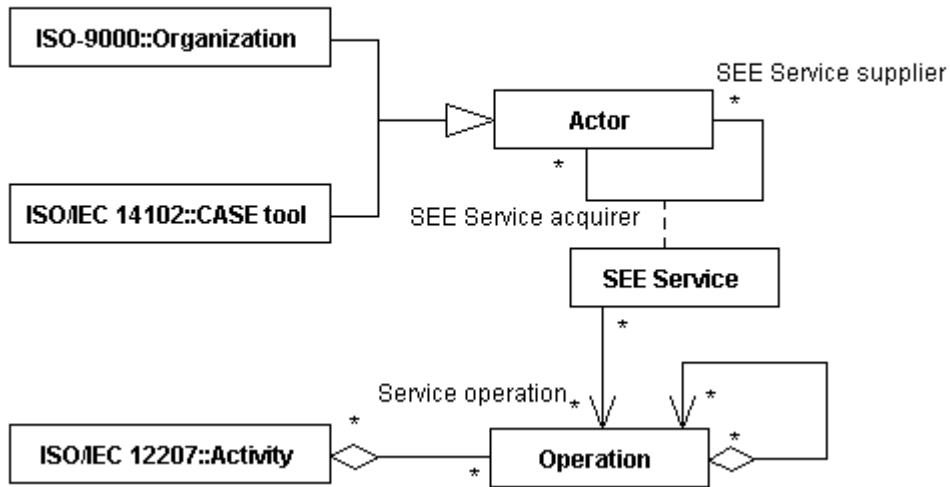
Exemplary automated supports for each SEE Services are listed in Annex A, it includes lists of corresponding service operations to help readers understand SEE

#### 4.3 Reference model

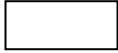
SEE services can be identified within a Reference model. This section presents those concepts that are part of this reference model (See Fig.1 SEE Reference Model described in UML). The reference model is made of the following concepts:

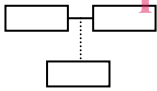
- Software Engineering Environment (model itself)
- SEE Service
- SEE Service Operation
- CASE Tool
- Actor
- Activity
- Organization


While engineering software systems and in related domains (e.g. project management), a life cycle Activity is achieved by one or more Operations. SEE Service operations satisfy target life cycle activities. Actor as SEE Service supplier provides SEE Service to another actor as SEE Service acquirer. SEE Service supplier and SEE Service acquirer are Organizations or CASE tools.

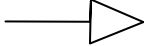


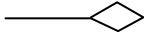
NOTE brief usages of UML notation are described here for readers benefit.

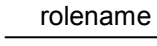
- 

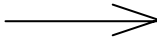
**class:** A class is drawn as a solid-outline rectangle with a class name. A class name preceding package names separated by double colons (::) indicate a class defined in another package.
- 

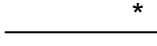
**association class:** An association that also has class properties (or a class that has association properties) shown as a class symbol (rectangle) attached by a dashed line to an association path.
- 

**association:** Binary associations are shown as lines connecting two class symbols.
- 

**inheritance:** A subclass "inherits" from the superclass, ("is a" relationship). It is represented in the image with a line starting at the subclass and ending in a white arrow at the superclass.
- 

**aggregation:** A superclass instance "uses a" subclass instance (or more than one). It is represented in the images with a line starting at the superclass with a white diamond and ending in the subclass.
- 

**rolename:** A name string near the end of the line indicates the role played by the class attached, if specified.
- 

**navigability:** An arrow may be attached to the end of the line to indicate that navigation is supported toward the classifier attached to the arrow. Normally one-way navigability is shown and two-way navigability is suppressed (no-way navigability is rare in practice).
- 

**multiplicity:** specifies the number of target instances that may be associated with a single source instance across the given Association. A single star "\*" denotes the unlimited nonnegative integer range multiplicity that means many.

Figure 1 — SEE Reference Model described in UML

## 5 Software engineering services

The services in this section support activities related to the specification, design, implementation, testing, and maintenance of software. The following services are defined and grouped in this section:

- Software requirements engineering services;
- Software reverse engineering service;
- Software re-engineering service;
- Software prototyping service;
- Software design service;
- Software modelling service;
- Software simulation service;
- Component based software generation service;
- Source code generation service;
- Compilation service;
- Debugging service;
- Software static/dynamic analysis service;
- Software testing services;
- Software verification service;
- Software integration services.

iTeh STANDARD PREVIEW  
(standards.iteh.ai)

[ISO/IEC 15940:2006](https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006)

<https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006>

### 5.1 Software requirements engineering service

#### 5.1.1 Service concept

This service provides the ability to capture, represent, analyse, validate, and refine those system requirements that are allocated to software components.

#### 5.1.2 Service operations

This service provides the ability to:

- Elicit and capture software requirements;
- Structure the software requirements;
- Create, modify, browse, and present software requirements;
- Group and prioritise software requirements;
- Check consistency of software requirements;
- Allocate software requirements for each software component;

- Conduct impact analysis for the addition, subtraction, or modification in a requirement against the project value, resources, and timeline;
- Validate and baseline the document specs based on stakeholders and developers.

## 5.2 Software reverse engineering service

### 5.2.1 Service concept

This service provides the ability to capture design information from source or object code, and produce structure charts, call graphs, and other design documentation to provide new functionality or support a new environment.

### 5.2.2 Service operations

This service provides the ability to:

- Generate design from source code;
- Generate source program from object code.

## 5.3 Software re-engineering service

### 5.3.1 Service concept

iTeh STANDARD PREVIEW

This service provides the ability to take a new or a modified set of software requirements and the existing design as input and produce a new or modified design.

### 5.3.2 Service operations

<https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006>

This service provides the ability to:

- Revise or restructure existing code;
- Perform impact analysis of new design on existing software components;
- Translate from one notation or language into another;
- Check that the new set of requirements is consistent with the existing system;
- Determine the impact of the altered design on the existing set of components.

## 5.4 Software prototyping service

### 5.4.1 Service concept

This service provides the ability to enable the production of a software system that reproduces the user interface and emulates the functionality and behaviour of the final system to be built.

### 5.4.2 Service operations

This service provides the ability to:

- Build a prototype from requirements;
- Invoke software modelling service if necessary and available;

- Produce a user interface from requirements;
- Execute a prototype;
- Conduct simulations if necessary and available.

## 5.5 Software modelling service

### 5.5.1 Service concept

This service provides the ability to model requirements and/or design in order to determine the effectiveness of alternative designs with respect to such attributes as user interface characteristics or execution flow.

### 5.5.2 Service operations

This service provides the ability to:

- Build a software model (graphical, logical, mathematical, formal, etc.) from requirements;
- Validate a software model;
- Map and/or transform one software model into another;
- Analyse a software model.

## 5.6 Software simulation service

### 5.6.1 Service concept

This service provides the ability to simulate models in order to determine the effectiveness of alternative designs with respect to such attributes as user interface characteristics or execution flow.

### 5.6.2 Service operations

This service provides the ability to:

- Build a simulation model by invoking the software modelling service if necessary and available;
- Execute a software model;
- Capture simulation results of software models.

## 5.7 Software design service

### 5.7.1 Service concept

This service provides the ability to capture, represent, create, analyse, and refine the design attributes of the software components of a system or subsystem. The outcome of the software design service includes the definition of the software components and sub-components.

### 5.7.2 Service operations

This service provides the ability to:

- Translate requirements into architecture and design elements;
- Create and modify software architecture and design representation;

- Validate architecture and design artefacts to requirements;
- Produce structure charts, graphs, screens or other design information from a design representation;
- Structure design specifications;
- Architecture and design documentation;
- Evaluate architecture and design representations.

## **5.8 Component based software generation service**

### **5.8.1 Service concept**

This service provides the ability to produce automatically and semi-automatically software components using existing components or component templates.

### **5.8.2 Service operations**

This service provides the ability to:

- Generate a parser from a syntactic language description;
- Generate a script for the composition and interconnection of software components;
- Generate a rule-based system from a set of rules;
- Generate a user interface component for a software system.

## **5.9 Source code generation service**

[ISO/IEC 15940:2006](https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006)

<https://standards.iteh.ai/catalog/standards/sist/9af023c1-e821-4df3-8f5a-5db2f1e9386c/iso-iec-15940-2006>

### **5.9.1 Service concept**

This service provides the ability to generate modules from design specifications.

### **5.9.2 Service operations**

This service provides the ability to:

- Generate modules from design specifications;
- Invoke the software static/dynamic analysis service and work-through source code;
- Provide for traceability to design specifications.

## **5.10 Compilation service**

### **5.10.1 Service concept**

This service provides the ability to support the translation (e.g., build, compile or interpretation) and linking of software components written in various programming languages. The principal outputs from this service are executable programs supporting the implementation of some target system.

### **5.10.2 Service operations**

This service provides the ability to:

- Find code and inheritance dependencies among a set of software components;

- Pre-process source code to produce modified source code;
- Apply macro expansions to source code;
- Translate a source program into some target object code language;
- Produce report on the translation; this may include source listings of various complexity, including cross-reference data, compilation speeds, CPU usage, etc.;
- Link the object code into executable images. When intended for use on a remote target, link code into loadable/bootable images;
- Update the compiled system incrementally to reflect new changes.

## 5.11 Debugging service

### 5.11.1 Service concept

This service provides the ability to locate and repair source code errors in individual software components by controlled or monitored execution of the code to track down errors and replace code.

### 5.11.2 Service operations

This service provides the ability to:

- Instrument source programs by inserting breakpoints, instruction traps, printing out data values, and modifying source text;
- Execute programs incrementally;
- Monitor and save execution output;
- Log and measure debugging results;
- Analyse properties of programs and their current data values.

## 5.12 Software static/dynamic analysis service

### 5.12.1 Service concept

This service provides for the static analysis, or source code analysis, of software components in order to determine execution structure within the component and for the dynamic analysis, or code in execution analysis, in order to determine execution behaviour characteristics.

### 5.12.2 Service operations

This service provides the ability to:

- Collect raw statistics from a software module/component;
- Compute complexity measures from a software module/component;
- Produce and graphically represent cross reference lists;
- Collect raw statistics from a software module/component in execution;
- Produce and graphically represent characteristics of the execution behaviour;
- Produce findings according predefined template or rules.