JFP
metaphor Lab.

SLP 記述ガイドライン

2015 年 7 月 1 日
株式会社ジェーエフピー

1　はじめに

この文書の目的は、SLP_vReq 2（以下 SLP）を用いて要求仕様書を記述するための、標準的な記述方法を示すことです。想定される読者は、SLP で要求仕様書を記述する人と、SLP で書かれた要求仕様書を管理する人です。

1.1　この文書の構成

このガイドラインでは第 2 章で SLP 単独での使い方を説明します。第 3 章では SLP をソフトウェア開発にどのように応用していくかを説明します。

2.1 節では SLP 文書を作成する方法を説明します。要求を SLP 文法に従って記述するためのアドバイスも、この節で示します。2.2 節では既存の SLP 文書を編集する方法を説明します。2.3 節では、SLP 文書を分析することで、要求仕様書の品質向上に役立てるためのヒントを示します。

第 3 章のうち 3.1 節では、要求仕様書を最初から SLP で作成する方法を説明します。3.2 節では、Microsoft Word や同 Excel などで書かれた従来の要求仕様書と、SLP 形式の要求仕様書を併用する方法を説明します。3.3 節では、既存の要求仕様書を修正し SLP を用いて新しい要求仕様書を作成する方法を説明します。3.4 節では、SLP 文書と他の文書との対応関係をトレースする（トレーサビリティ）方法を説明します。3.5 節では、形式手法の一種である VDM（Vienna Development Method）という開発手法と SLP とを関連付ける方法を説明します。ただし、VDM そのものについての説明を省き、要点のみを示します。3.6 節では、SLP を用いてテスト計画を作成する方法を説明します。

1.2　アドバイスの略語

この文章には、通常の説明文に加えて、その簡潔な要約である「略語」が含まれています。アドバイスには、利便性のため、連番の略語を作成しました。略語はグループごとに接頭語を付けて区別しています。各グループの接頭語を以下に示します。

接頭語	意味
NM	名前の付け方
LD	論理記述欄を記述する方法
FN	SLP の機能の紹介
GN	全般的な心構え
UI	操作方法

また、巻末にはアドバイスの一覧を収録しています（第 4 章）。

1.3　SLP とは

日本科学技術連盟ソフトウェア品質管理研究会の調査によると、ソフトウェア開発のライフサイクルの中で、要求仕様書に起因する問題が 40 %を占めています。また、受発注間において仕様の共有が困難であるという悩みが往々にして聞かれます。要求仕様書の品質が悪いと、以降の全ての工程に影響するため、要求仕様書の問題は深刻です。

SLP は、仕様書の用語の統一と、論理整合性によって、これらの問題の解決を目指します。ここで、用語の統一とは、文要素（メンバー名と状態名）が、余分な助詞や助動詞を交えず適切な語幹から成っていることを指します。また、論理整合性とは、複数の文の関に矛盾がないことと、条件に漏れがないことを指します。

SLP を導入することによって、仕様書の用語が整理され、仕様書の論理的な構造が明確になると共に、仕様についての誤解と思い込みが減り、仕様書の変更を素早く正確に行うことができます。そして最後に、レビューの効果が確実になります。

SLP の導入により特に顕著な効果が認められるものとして、箇条書きの要求事項や構想などの開発初期の文書や分析が不十分な仕様書があげられます。一方、それらが十分と思われる仕様書でも検査の対象として十分な効果が認められます。

SLP の基本的構造について以下に述べます。まず、要求と仕様を文の集合としてアプローチし、その文を主語と述語、または目的語と述語（単位文）に構成し直します。続いて、単位文を論理学的に関係付け、その上で主語、述語、目的語の用語をさらに分割し、新たな単位文を作ります。これは意味を精義化（refine）し、文の論理関係（条件）を明確にしていく作業でもあります。また、SLP では用語の検査と論理矛盾の検査が可能です。これらの方法により、最終的に文の集合である要求仕様文書は用語が統一され、論理矛盾なく構成されます。

1.3.1 要求の構文化手法

要求開発には図のように４つの工程があります。開発プロセスは前後、またそれ以前の工程と行き来しながら進みます。

最後は要求の妥当性を確認しなければなりませんが、そのためには、要求や（非）機能を正確な文（コトバ化）にし 文をSLPの構文規則に従い、文を形式的な関係で結びます。SLPはこの構文規則をツールとして実現していますから、ユーザーはSLPの求めるままに、主語や述語等を入力することで、文は構文に合った正しい形式的な関係となります。
ただし、形式が正しくても意味が異なっては妥当な要求仕様書といえません。意味の判定はユーザー行わなければなりませんが、形式が自然に整う分だけ、意味の正誤の判断に集中することができます。

コトバ化や文の形式的な関係づけはSLPでは「構文化」手法と名付けられています。

形式手法の適用に際しても、その前に用語の意味を確定することが重要であることが提唱されています。このような事前の意味確定プロセスの重要さを、形式手法のラーセン氏らも唱えています。形式手法の前段階として、“Pre-Formal”と呼ばれています。

１．要求獲得：
- 要求を様々挙げる。
- 図や表なども用いる。
- 要求項目別に分類整理する（章立てする）。

２．要求分析：
- 関連する要求が他に無いか明確にする。
- 要求の意図、理由を明確にする。
- 非機能要件を明確にする。
- ソフトウェアの稼働環境を明確にする。

３．要求仕様化
- 要求や機能の意味を考え、　意味同士を関係づける。
- 意味を最小単位の文（単位文）にする。
- 文を条件に応じた論理関係（含意、連言、、、）で結ぶ。
- 文の用語を統一する。

４．要求妥当性確認：
- すべての要求の論理関係（条件）を確認し、要求の意味が実現性や採算性等からも適切かを確認する。
- この開発の初期段階での要求は、開発の最終の成果物に比べて具体性（外延性）は同じではないが、抽象性（内包性）は同じでなければならない。

2　SLP を使う

2.1　SLP 文書を作成する

2.1.1　ヘルプの見方

SLP はオンラインヘルプを備えています。キーボードの F1 キーを押下するとヘルプが表示されます。また、以下の方法で操作すると、オンラインヘルプの特定の項目を開くことができます。メインメニューの項目にマウスカーソルが乗っている状態で F1 キーを押下すると、そのメニュー項目についてのヘルプが表示されます。メインウィンドウ以外のウィンドウを表示しているとき F1 キーを押下すると、そのウィンドウの操作方法が表示されます。

UI-01 キーボードの F1 キーを押下するとオンラインヘルプが表示される。

ソフトウェアの具体的な操作方法は「取扱説明書（操作編）」で知ることができます。
本ドキュメントでは操作方法の詳細は省略し、より基本的な概念について説明します。

2.1.2　SLP 文法で要求を記述する

SLP のもっとも基本的な機能の一つは、要求を「SLP 文法」に従って記述できることです。
SLP 文書は複数の「単位機能」の集まりです。1 個の単位機能は 1 個の論理記述欄を持ちます。論理記述欄には、要求を SLP 文法で記述します。

2.1.2.1　論理記述欄の記述方法

論理記述欄に構文を追加する方法は、右クリックメニューで行います。論理記述欄を右クリックして、メニューから選択することで、論理記述欄に構文を追加することができます。また、構文を選択した状態で「削除」メニューを選択すると、行単位やブロック単位で構文を削除することができます。
右クリックメニューで、記述可能な構文、以前に記入や登録したメンバー名、状態名、機能名が各々の入力領域で表示されます。構文は選択、項目名は選択か自ら記述します。

if 文などで用いられる仕様内容を記述する文は、主語と述語に分けて記述することとしています。これを陳述文と呼びます。命令文（Do）の場合は、目的語と（命令）述語です。if 文や Do 文はこれらの用語の各々 1 つずつをもって、陳述文、命令文を構成します。これらを合わせて単文と呼ぶこととします。

また主語と目的語の指す対象をメンバーと呼び、その名称を**「メンバー名」**と名付けます。
述語（陳述述語と命令述語）の指す状態や属性を総称して**「状態名」**と呼びます。
メンバー名は“< >”で括り、状態名を“{}”で括ります。つまり、メンバー名、状態名はそれぞれ“< >”、“{}”の中に記述されなければなりません。

FN-01 SLP 文法で要求を記述するには論理記述欄を用いる。

UI-02 論理記述欄に構文を追加するには右クリックメニューを用いる。

2.1.2.2　構文の意味と使用法

SLP 文法の構文のうち、要求を記述するために不可欠なものは「if」「Do」「Fn」の 3 種類です。それ以外に、用途によっては便利な構文が、「switch」「Do nothing」「コメント」「改行」「loop」「exitlp」「for」「while」の 8 種類です。

LD-01 「if」「Do」「Fn」の 3 種類の構文は不可欠である。他の構文は用途によっては便利である。

2.1.2.3 条件分岐

if 構文は「○○ならば○○せよ、そうでなければ○○せよ」という条件分岐を表します。if 構文は以下のような形式です。

```
if <メンバー名>が{状態名}ならば
    (肯定側の記述)
else
    (否定側の記述)
endif
```

論理記述欄に if 構文を追加すると、「if」「else」「endif」の 3 行が自動的に作られます。肯定側だけでなく否定側も記述するよう心掛けることで、要求を網羅的に記述することができます。

```
（例）
 if <信号機の色>が{青}ならば
    (車を進行せよ)
 else
    (車を停止せよ)
 endif
```

LD-02 if 構文を使用するときは、肯定側と否定側の両方を記述するよう心掛けることで、要求を網羅的に記述することができる。

論理記述欄に if 構文を追加すると、助詞に「が」が自動的に挿入されます。ここで言う助詞とは、メンバー名と状態名の間の入力エリアのことです。if 構文の助詞は「が」とすることを推奨しますが、必要に応じて変更しても構いません。これは、一般的な日本語文法でいうと、メンバー名は主語、状態名は述語に相当します。{青}以降の「ならば」のような助動詞についても候補リストから選択入力できますし、直接入力も可能です。

LD-03 if 構文の助詞（メンバー名と状態名の間）は「が」にする。

LD-04 if 構文では、メンバー名を主語、状態名を述語にする。

2.1.2.4 命令または平叙文

Do 構文は「○○せよ」という命令を表しますが、場合によっては「○○である」という平叙文に用いても構いません。Do 構文は以下のような形式です。

```
Do <メンバー名>を{状態名}せよ
（例）
  Do <車>を{停止}せよ
  Do <車>は{停止中}である
```

LD-05 Do 構文は「○○せよ」という命令を表す。

LD-06 Do 構文を「○○である」という平叙文に用いる。

論理記述欄に Do 構文を追加すると、中置助詞に「を」、後置助詞に「せよ」がそれぞれ自動的に挿入されます。ここで、中置助詞とは、メンバー名と状態名の間の入力エリアのことです。また、後置助詞とは、状態名の後の入力エリアのことです。Do 構文の中置助詞と後置助詞はそれぞれ「を」と「せよ」にすることを推奨しますが、必要に応じて変更しても構いません。これは、日本語文法の用語でいうと、メンバー名が目的語、状態名が述語に相当します。

LD-07 Do 構文の中置助詞（メンバー名と状態名の間）は「を」にする。

LD-08 Do 構文の後置助詞（状態名の後）は「せよ」にする。

LD-09 Do 構文では、メンバー名を目的語、状態名を述語にする。

2.1.2.5 機能を分割する

Fn 構文は機能を複数の単位機能に分割する役割を持ちます。Fn 構文は以下のような形式です。

Fn [単位機能名]

例えば「○○せよ」という命令があるとき、これを論理記述欄で「Fn [○○せよ]」と記述して、その具体的な内容を「○○せよ」という別の単位機能に書くことができます。論理記述欄の記述が長くなったり、ネスト（階層）が深くなったりしたとき、Fn 構文を使うことで、記述の一部を他の単位機能に移すことができます。

LD-10 論理記述欄の記述が長くなったり、ネストが深くなったりしたとき、Fn 構文を使うことで、記述の一部を他の単位機能に移すことができる。

2.1.2.6 複数の選択肢を持つ条件分岐

switch 構文は複数の選択肢を持つ条件分岐を表します。switch 構文は以下のような形式です。

```
switch <メンバー名>が
case {状態名 1}
    (メンバー名が状態名 1 である場合の記述)
case {状態名 2}
    (メンバー名が状態名 2 である場合の記述)
case {状態名 3}
    (メンバー名が状態名 3 である場合の記述)
elsw
    (メンバー名が状態名 1 でない、かつ、メンバー名が状態名 2 でない、かつ、メンバー名が状態名 3 でない場合の記述)
endsw
```

論理記述欄に switch 構文を追加すると、「switch」、3 行の「case」、「elsw」、「endsw」の 6 行が自動的に作られます。1 個の switch 構文につき 2 個以上の case 構文を入力できます。
switch 構文のメンバー名と状態名は、それぞれ、if 構文のメンバー名と状態名と同じように入力します。また、switch 構文のメンバー名の後の入力エリアは助詞、case 構文の状態名の後の入力エリアは助動詞です。これらはそれぞれ、if 構文の助詞（メンバー名と状態名の間）と助動詞（状態名の後）と同じように入力します。

2.1.2.7 ブロックに何も記述しないことを明示する

SLP の「構文検査」では、ブロックに何も構文が書かれていないと警告を発します。ここで言うブロックとは、if と else の間、else と endif の間、case と次の case の間、最後の case と elsw の間、elsw と endsw の間のことです。ブロックに何も構文が書かれていないことが間違いではないことを明示するためには、ブロックを Do nothing 構文で埋めてください。

2.1.2.8 コメント

「コメント」構文を使うと、SLP 文法では記述しにくい事柄を自由に記述することができます。

2.1.2.9　　改行

「改行」構文を使うと、論理記述欄の意味の区切りを視覚的に表すことができます。

2.1.2.10　　ループ

loop 構文、for 構文、while 構文はループを表します。論理記述欄に loop 構文を追加すると、「loop」「endlp」の 2 行が自動的に作られます。論理記述欄に for 構文を追加すると、「for」「endfor」の 2 行が自動的に作られます。論理記述欄に while 構文を追加すると、「while」「endwhile」の 2 行が自動的に作られます。また、exitlp 構文はループを抜けることを表します。ループの構文（ loop, endlp, for, endfor, while, endwhile, exitlp ）は、単にループを表現する「コメント」構文です。ループの回数の記述は必要なく、後述するようにループの回数に依存した検査はありません。for 構文、while 構文は以下のような形式です。通常、for 構文のメンバー名 1 とメンバー名 2 とメンバー名 3 は、ループの条件を記載するため、同じメンバー名になります。

for <メンバー名 1>を{状態名 1}せよ; <メンバー名 2>が{状態名 2}; <メンバー名 3>を{状態名 3}せよ
　(メンバー名 2 が状態名 2 である場合の記述)
　※メンバー名 2 が状態名 2 のとき、上述の処理を繰り返す。初期設定として、メンバー名 1 を状態名 1 にし、繰り返しのたびにメンバー名 3 を状態名 3 にする。
endfor

while <メンバー名>が{状態名}
　(メンバー名が状態名である場合の記述)
　※メンバー名が状態名のとき、上述の処理を繰り返す。
endwhile

2.1.2.11　　用語の命名規則
2.1.2.12　　メンバー名

メンバー名は if 構文、Do 構文、switch 構文に含まれる入力エリアです。メンバー名は if 構文では主語、Do 構文では目的語の役割を担います。そのため、メンバー名は名詞や名詞句、名詞節で命名します。メンバー名は文書全体で一意な名前を付けるのが原則です。そのために、必要に応じて修飾語を付加して、長さにこだわらず最も適切なメンバー名を付けることを推奨します。

NM-01 メンバー名は名詞（名詞句、名詞節でもよい）にする。

NM-02 メンバー名は文書全体で一意な名前にする。

NM-03 メンバー名は長さにこだわらず、一意な名前になるまで修飾語を加える。

メンバー名は「状態を持つもの」にすることを推奨します。例えば、「信号機の色」は「信号機」よりもメンバー名に適しています。なぜなら、状態名（後述）として「赤」「黄」「青」「正常」「停電」「故障」があるとすると、これら全てをメンバー名「信号機」に結び付けるよりも、メンバー名「信号機の色」を状態名「赤」「黄」「青」に、メンバー名「信号機の動作状況」を状態名「正常」「停電」「故障」にそれぞれ結び付けた方が、文要素（メンバー名と状態名）の関係が明確になる為です。

NM-04 「状態を持つもの」をメンバー名とする。

メンバー名には論理学の記号である量化記号を含めることができます。例えば、「すべての本」「ある本」などのメンバー名は、「すべての」と「ある」が量化記号と認識されます。量化記号は「量化記号」ウィンドウで設定できます。「通常文に変換」「条件の論理性と用語の確認（構文化決定表作成）」などの操作で否定文（else または elsw）を出力するとき、量化記号を含むメンバー名は、量化記号を対記号（否定語）に置き換えて出力されます。先ほどの例で言えば、「すべての本」の否定は「ある本」となります。

ここで、量化記号を含むメンバー名として、例えば「すべての男と女」というように複数の概念を含むように記述すると問題が起きます。「すべての男と女」の否定は「ある男と女」として出力されますが、これは正しくありません。正確には「すべての男と女が○○である」の否定は、強いて言えば、「ある男が○○でない、または、ある女が○○でない」です。このような問題を防ぐには、「すべての男と女」のようなメンバー名を使わず、「すべての男」と「すべての女」のように複数のメンバー名に分けることを推奨します。

NM-05 量化記号を含むメンバー名を使うときは、「すべての男と女」のように書かず、「すべての男」と「すべての女」に分ける。

2.1.2.13 状態名

「状態を名詞」として記述できる場合は、状態名は名詞（名詞句、名詞節でもよい）で命名します。そうでないときは、状態名を動詞で命名します。ただし、その動詞の活用語尾の扱いは、動詞がサ行変格活用であるか、状態名が使用されているのが if 構文であるか、もしくは Do 構文であるかによって異なります。

まず、動詞がサ行変格活用であるときは、語幹までを状態名にして、活用語尾を省きます。このとき、状態名が使われているのが if 構文であれば、助動詞（状態名の後の入力エリア）に「した」「している」などの活用語尾を入力します。また、状態名が使われているのが Do 構文であれば、後置助詞（状態名の後の入力エリア）に命令形の活用語尾「せよ」を入力します。（詳細は後述）

さらに、動詞がサ行変格活用でないときは、if 構文と Do 構文とで状態名の命名方法が異なります。if 構文では「0 より大きい」などのように平叙文で状態名を記述します。一方、Do 構文では、「閉じよ」などのように命令文で状態名を記述します。いずれの場合にも、助動詞と後置助詞（いずれも状態名の後の入力エリア）には何も入力しません。

（例）動詞がサ行変格活用であるとき（「到達する」、「表示する」）

```
if <回答メッセージ>が{到達}している
    Do <回答メッセージ>を{表示}せよ
else
    Do nothing
endif
```

NM-06「状態を名詞」として記述できるならば、状態名を名詞（名詞句、名詞節でもよい）にする。

NM-07 状態名を名詞で表現できないときは、状態名を動詞にする。ただし、その動詞がサ行変格活用であるときは、語幹までを状態名にして、活用語尾を省く。

NM-08 状態名を名詞またはサ行変格活用の動詞で表現できないとき、if 構文では、「0 より大きい」などのように平叙文で状態名を記述する。

NM-09 状態名を名詞またはサ行変格活用の動詞で表現できないとき、Do 構文では、「閉じよ」などのように命令形で状態名を記述する。

2.1.2.14 時制表現

ところで、if 構文や switch 構文の状態名の後ろには、状態が「たったいま変化した」したなどのような、その時の時間を規準において状態の変化を表現することができます。これを時制の変化と称し、その表現を助動詞で表しています。

これにより、例えば、「信号が青になった」ことと、「信号が青である」ことと、また「信号が青であった」こと、さらに「信号が青であったなら（実際は青ではなかった）」などの時制に応じた表現を区別することができます。このような区別の必要性は、道路を渡る人（ドライバの場合でもいいのですが）がどんな状態にいるか、ということに関係して、その人の行動を決めるからです。

もし信号が青「になって」、そのときその人が信号を待っていたなら、歩き始めよ、ということになるでしょうし、もし信号が青「である」なら、その人は歩き「始める」までもなく、歩いてよい状態が

以前から続いており、他方またその人が今たまたま信号の下で立っていたなら歩き始めてもいいし、信号が変化する前から歩いていた人であるならば、そのままとどまらず歩いてよいことになります。

またこの後者の人（信号が変化する前から歩いていた人）は、ちょうどタイミングよく青「になった」ことに出くわし止まらずに済んだ、などのちょっとした状況の違いが存在します。SLP では、このようなわずかな違いも表現できるようになっています。

SLP は時制を元にした状態の変化を表現し、それに応じて要求仕様を記述できるようになっています。また SLP が扱う状態の変化の表現は、以下の３つです。

① 状態変化（現在の状態の変化）・・・「信号が青になった」

② 現在状態・・・・「信号が青である」

③ 過去状態・・・・「信号が青であった」

上にあげた③のケースで「信号が青であったなら（実際は青ではなかった）」は、SLP の範囲外とします。要求仕様の記述には、過去の出来事に反する仮定はないと考えられるからです。

また過去ではなく、現在が「信号が青であったなら（実際には現在は青ではない）」も、SLP は範囲外とします。SLP では現在の状態において記述することを規則とします。この条件は、「信号は（現在）青ではない」と記述する必要があります。

とすると、過去の状態「信号が青であった」はどう解釈すべきか。これは、過去の経験を現在に持ち越した状態、といえます。信号では例が悪いので、クルマの例を取ります。「そのクルマは事故を起こした」という文は、過去のことを記述していますが、現在時制でいえば「そのクルマは事故車である」となります。このように SLP では過去の状態で表現されてとしても、現在がどうであるかと判断することになります。

また未来に関しても、①状態変化（現在の状態の変化）として解釈されます。「もし先々大地震が起きたならば」という条件は、ずっと先のあるか無いか不確かな条件ですが、SLP では、単に「大地震が起きた」と（それが、いつか分からぬながら）「現在の状態の変化」として扱います。要求仕様の記述がそうであるからです。

2.1.2.15 文要素をどこまで書けばよいか

このように状態は時制と結び付いています。同じ（信号の）「青」でも、「なった」、「である」、「であった」を「青」の状態に含むと内容が異なります。

他方、状態が命令文に用いられる場合には、（信号を）青「にせよ」とか「とせよ」というように助動詞は命令形にとなっています。

SLP では記述された用語の検査を行います。記述した用語が一様に使用されているか否かを SLP 文書全般に渡りチェックします。状態名の検査においては、if 構文の状態名も Do 構文の状態名も区別なく同じようにチェックします。このとき、状態名に「青となった」と「青とせよ」が存在した場合には、SLP は異なる状態と判断します。

用語の検査ではなく、論理矛盾のチェックの場合には、このように記述していたとしても、この違いは論理矛盾のチェックには影響を与えません。従って、この場合には if 構文と Do 構文における状態名の違いは気にすることはありません。しかし複数の if 構文の場合にはきちんと統一を図っておく必要があります。SLP は現在文字列の比較でしか文字の異同を検査できませんので、語彙の異同にはご注意ください。

以上の制約を念頭に置いた上で、状態名の記述においては、「青」とか「赤」の属性や状態の記述にとどめ、if 構文の場合（や switch 構文）には助動詞を付記することをお薦めします。

なお、命令文の場合の助動詞は一様であるため、助動詞のガイドはありません。コメント文を自由に書き込むことができるようになっています。

2.1.2.16　単位機能名

単位機能名は動詞の命令形にすると、簡潔で分かりやすく命名できます。単位機能名は文書全体で一意である必要があります。そのため、必要に応じて修飾語を付加して、十分に長い名前を付けることを推奨します。特に、親単位機能の名前に条件を表す修飾語を加えて子単位機能を命名すると、単位機能名を容易に命名できます。例えば、親単位機能が「○○せよ」であるとき、子単位機能は「××のとき○○せよ」と命名します。最終的には、インデント目次を見ただけで文書全体の流れが分かるように単位機能名を付けるようにします。

注意を要するのは、既存の要求仕様書の階層構造をそのまま単位機能名に用いると、SLP 文書としては読みにくくなります。既存の要求仕様書は階層構造が合理的でないものが多くなっているからです。そのため、一つの意味のまとまりを理解するために、複数の節に分散して書かれた記述を読むことがしばしば必要になります。SLP 文書を作成するにあたっては、文書の階層構造を新たに作り直すことを推奨します。実用的には、第 1 段階で従来の要求仕様書の階層構造をそのまま用いた SLP 文書を作り、第 2 段階で独自の階層構造を作るという二段階方法を推奨します。

一方、論理記述欄ではなく機能内容欄を主体とした使い方では、従来の要求仕様書の階層構造を踏襲して単位機能名を付ける方法が現実的です。

NM-10 単位機能名は動詞の命令形で命名する。

NM-11 従来の要求仕様書の階層構造にとらわれず、独自の単位機能名を付ける。

NM-12 単位機能名は文書全体で一意にする。

NM-13 単位機能名を命名するとき、一意な名前が容易に思いつかないときは、親単位機能の名前に条件を表す修飾語を加えて子単位機能を命名する。

2.1.2.17　助詞と助動詞についての補足

前述のように、論理記述欄に if 構文を追加すると、助詞（メンバー名と状態名の間の入力エリア）に「が」が自動的に挿入されます。また、論理記述欄に Do 構文を追加すると、中置助詞（メンバー名と状態名の間の入力エリア）には「を」が自動的に挿入されます。そして、後置助詞（状態名の後の入力エリア）には「せよ」が自動的に挿入されます。原則としてはこれらの助詞をそのまま用いるのが望ましいですが、状態名の性質によっては、助詞を変更する必要があります。特に、状態名が名詞であるか、サ行変格活用の動詞であるか、それ以外の動詞であるかによって、望ましい直後の助詞が異なります。

また、if 構文には「助動詞」という入力エリアがあります。助動詞は if 構文または case 構文の、いずれも状態名の後に存在します。if 構文または case 構文を追加しても、助動詞が自動的に入力されることはありません。かわりに、助動詞が置かれるべき位置にカーソルがある状態で右クリックすると、「候補リストから選択」メニューで助動詞を選択することができます。また、メニューから候補を選択するのではなく、助動詞を直接入力することも可能です。いずれにせよ、望ましい助動詞は、直前の状態名の性質によって異なることに注意してください。

まず、状態名が名詞であるとき、if 構文の助動詞は、名詞に接続するように、「である」「になった」などとします。Do 構文の後置助詞は、命令文であれば「にせよ」、平叙文であれば「である」にします。

次に、状態名がサ行変格活用の動詞の語幹であるとき、if 構文の助動詞は「した」「している」などとします。Do 構文の後置助詞は「せよ」にします。

最後に、状態名がサ行変格活用ではない動詞のとき、if 構文の助動詞と、Do 構文の後置助詞は、ともに空にし、状態名に活用語尾を含めます。

LD-11 直前の状態名が名詞のとき、if 構文の助動詞は「である」「になった」などとする。

LD-12 直前の状態名が名詞のとき、Do 構文の後置助詞は「にせよ」「である」などとする。

LD-13 直前の状態名がサ行変格活用の動詞のとき、if 構文の助動詞は「した」「している」などとする。

LD-14 直前の状態名がサ行変格活用の動詞のとき、Do 構文の後置助詞は「せよ」にする。

LD-15 直前の状態名がサ行変格活用でない動詞のとき、if 構文の助動詞は空にする。

LD-16 直前の状態名がサ行変格活用でない動詞のとき、Do 構文の後置助詞は空にする。

2.1.3 要求を自由に記述する

SLP 文法に従って論理記述欄を記述するのは、要求を論理的かつ簡潔に書くためには有益です。しかし、従来の要求仕様書に慣れている人にとって、新しい方法論になじむまでの抵抗感も大きいかもしれません。SLP の特性をより手っ取り早く理解するには、論理記述欄ではなく機能内容欄を主体に使う方法もあります。

GN-01 SLP をより手軽に使うには、論理記述欄ではなく機能内容欄を主体に使っていく。

機能内容欄ではリッチエディットコントロールが有効であり、ユーザーが文字を自由に入力できます。また、フォント（フォントファミリー、サイズ、色、太字、斜体、下線）も変更することができ、他のアプリケーションから、画像、図形、表などをコピー・アンド・ペーストすることができます。また、OLE オブジェクトを挿入することもできます。
既存の要求仕様書と同じように、機能内容欄に自由に書いて、それをインデント目次または階層目次を使って階層的に整理することができます。また、OLE オブジェクトをファイルへのリンクにすることで、複数のファイルに分散した要求仕様書を、SLP 文書を中心にして階層的に管理することができます。

GN-02 既存の要求仕様書と同じように、機能内容欄に自由に書いて、それをインデント目次または階層目次を使って階層的に整理することができる。

GN-03 OLE オブジェクトをファイルへのリンクにすることで、複数のファイルに分散した要求仕様書を、SLP 文書を中心にして階層的に管理することができる。

論理記述欄ではなく機能内容欄を中心に使うときには、単位機能名は、従来の要求仕様書と同じように、体言止めで命名することを推奨します。従来の要求仕様書が階層構造を体言止めにしている場合が多く、また体言止めの方が単位機能名を短くできる為です。

NM-14 論理記述欄ではなく機能内容欄を中心に使うときには、単位機能名は、従来の要求仕様書と同じように、体言止めで命名する。

2.1.3.1 単位機能の作成方法について

単位機能の作成方法には、以下の２つのモードがあります。

(1) 同期モード（【要求仕様化フェーズ】に対応）
このモードでは、単位機能は目次欄と論理記述欄のいずれか一方に作るだけで自動的に両欄に作られます。これを「同期」的と呼びます。
論理記述欄の場合には、ユーザーが単位機能（Fn 構文）を作成すると、その単位機能は目次欄でも適切な項番が自動的に採番されます。これを「確定」的と呼びます。
他方、目次欄の場合には、その単位機能は論理記述欄にも自動的に記述されますが、その位置（行）が適切とは限りません。したがって、これを「暫定」的と呼びます。
「暫定」を「確定」にするには、ユーザーの論理記述欄での修正操作が必要となります。
（Fn 構文を、ブロック内の適切な位置に配置することです。）

なおこのモードを選択した場合、下記の半同期モードで作った「未定」単位機能が残っていた場合には「未定」をすべて論理記述欄に自動的に一括記述してから本モードを行います。ただし記述は「暫定」になります。

(2) 半同期モード　（【要求獲得・分析フェーズ】に対応）

ユーザーは単位機能を目次欄のみで作成します。この単位機能が論理記述欄に自動的に書き込まれることはありません。他方論理記述欄で作成された単位機能（Fn 構文）は目次欄にも自動的に書き込まれます。

このように書き込みが双方とも同じようになされる訳ではありませんので、これを「半同期」的と呼びます。

またこの場合、目次欄にのみ書き込まれた単位機能は論理記述欄には反映されませんので「未定」単位機能と呼びます。

他方、論理記述欄で書かれた単位機能は目次欄にも書かれ、かつ項番も適切に自動採番されますので「確定」単位機能と呼びます。

2.1.3.2　単位機能の同名と再利用

プログラムでは同名のラベルは同じ内容を指しますが、要求仕様書の場合はそうではありません。

目次にある項目名は同じでも、内容が異なっているものが多くあります。目次の例を見てみます。
「概要」、「目的」、「対象」、「用語説明」、「機能概要」、「機能一覧」、「インタフェース」
「非機能要件」、、、など。これらの項目はおなじみのタイトル名です。

要求Ａと要求Ｂを書き表すために、ＡとＢはそれぞれ下位に「概要」や「目的」を持ちます。

「要求Ａの概要」などと上位概念で逐一限定すれば、項目名はすべて固有にすることはできるかもしれませんが、大きな項目の下に、通常の視点となる項目群を持つことで、文書を構成します。

本ツールではこのような目次の特性を生かすために、単位機能の「同名」を許しています。「同名」でも内容は異なる訳です。いわば「同名異体」です。

異なる単位機能を「個別」単位機能と呼びます。「同名」でも「個別」が存在するという訳です。

また、同名同内容（「同名同体」）のものを他の箇所で利用（引用）することもあります。
この機能を「再利用」と呼びます。単位機能としては「共通」と呼びます。再利用される「共通」には
再利用の元となるオリジナルなものと、そうでないものとに分類できます。これを「正」、「副」とします。

これらの単位機能の違いを理解することは重要です。以下に整理した表を掲げます。

	単位機能の種類	同名	内容	状態
共通単位機能	正	○	○	同名同体
共通単位機能	副	○	○	同名同体
個別単位機能	×	○	×	同名異体
個別単位機能	×	×	―	異名

2.2　SLP 文書を編集する

SLP 文書の編集は、SLP 文書の作成とほぼ同じ操作です。論理記述欄のメンバー名、状態名、助詞、助動詞を変更するには、変更したい入力エリアにカーソルを移動して、文字を削除または挿入します。また、論理記述欄の構文を追加したり削除したりするには、論理記述欄を右クリックしてメニューを選択します。

2.3　SLP 文書の版を管理する

SLP 文書は、1 文書内で 10 版まで、版の管理をすることができます。
一般的な「メジャーバージョン番号＋マイナーバージョン番号」の型式で、版数をファイル保存時に自動更新することもできますし、手入力で明示的な版数を指定することができます。
また更新履歴を表示させることも可能ですので、作業の経緯を振り返ることもできます。

2.4　SLP 文書間の差分を表示する

SLP 文書の改版における差分を表示させ、視覚的に知ることができます。比較は前章の機能で付与された版数を基準に行い、結果を「検査・検索等結果」ウィンドウに表示します。
これにより、改版時のより具体的な修正項目を把握することができます。

2.5　SLP 文書間の単位機能を結合する

SLP 文書間で単位機能を相互にコピーすることができます。
派生開発などで、ある単位機能をそのまま移植したいようなときに使用します。
単位機能の項番を指定することによりコピー元を決定し、コピー先ではその位置を指定します。
目次に追加しないで単位機能一覧にのみ追加させて、後の編集操作で属性を決定することも可能です。

2.6　SLP 文書を分析する

SLP 文書を作成したら、今度はその SLP 文書を分析します。分析の結果をもとに SLP 文書を修正して、より厳密で分かりやすくすることができる為です。SLP 文書を分析する方法には、狭義の検査（構文検査、矛盾検査、冗長検査）、用語に関する機能（曖昧語検索、類似語検索、文要素のエクスポート）、その他の機能（条件の論理性と用語の確認、条件順序性確認）があります。

2.6.1　SLP 文書を検査する

SLP 文書を検査するには、メインメニューの「検査」以下の項目を使います。
構文検査は、文要素（メンバー名と状態名）が空文字列でないかどうかと、ブロックが空でないかどうかを検査します。ブロックを空にすることが正しいと確信でき、構文検査で警告を出す必要がないならば、ブロックに Do nothing 構文を挿入してください。
矛盾検査は、論理的な矛盾を検査します。「矛盾の可能性があります」という警告が出ますが、この警告を出したくないときは、「メンバー属性無矛盾化設定」を使用します。「メンバー属性無矛盾化設定」の具体的な操作方法はオンラインヘルプを参照してください。なお、「検査・検索等結果」ウィンドウの「矛盾の可能性があります」という警告メッセージをダブルクリックすると、すぐに「メンバー属性無矛盾化設定」

ウィンドウが開きます。逆に、「文どうしの関係定義」を使用すると、ユーザーが指定した組み合わせで、明示的に矛盾を検出して、警告メッセージを出力させることができます。

FN-02 矛盾検査の結果をユーザーが制御するには、「メンバー属性無矛盾化設定」と「文どうしの関係定義」を用いる。

UI-03 「検査・検索等結果」ウィンドウの「矛盾の可能性があります」という警告メッセージをダブルクリックすると、すぐに「メンバー属性無矛盾化設定」ウィンドウが開く。

冗長検査は、論理的な冗長を検査します。

いずれの検査でも、検査の対象となるのは文要素です。SLP の検査は、文要素をそれぞれ機械的に比較しており、内容を理解した上で比較する能力は持っていません。そのため、同じ事柄を意味するメンバー名または状態名は、正確に同じ文字列で記述する必要があります。

NM-15 同じ事柄を意味するメンバー名または状態名は、正確に同じ文字列で記述する。

メンバー名と状態名を除く入力エリア（すなわち、助詞、助動詞、コメント）は検査の対象とはなりません。これらの入力エリアは、単に読む人にとって読みやすくする為にあります。

なお、loop 構文、for 構文、while 構文を使用した場合、ループの回数に依存した検査は行われないことに留意してください。

NM-16 メンバー名と状態名を除く入力エリア（すなわち、助詞、助動詞、コメント）は、単に読む人にとって読みやすくするためにある。

GN-04 loop 構文、for 構文、while 構文を使用した場合、ループの回数に依存した検査は行われない。

2.6.2 使用を禁止または制限したい用語を管理する

使用を禁止または制限したい用語を管理するには、そのような用語を「曖昧語」に登録します。曖昧語を登録するには「曖昧語検索」ウィンドウを使用します。「曖昧語検索」ウィンドウでは、その名の通り、曖昧語検索を行うことができます。曖昧語検索は、登録されている曖昧語を、機能内容欄と論理記述欄から検索する機能です。

FN-03 使用を禁止または制限したい用語を管理するには「曖昧語検索」を使用する。

2.6.3 表記が揺れている用語を発見する

表記が揺れている用語を発見するには「類似語検索」を用います。これは、論理記述欄の文要素（メンバー名と状態名）のうち、類似度が一定以上である用語の組を発見する機能です。二つの文字列がどの程度に類似しているかの決定が求められる、いわゆる類似度の判定には、レーベンシュタイン距離が用いられています。

FN-04 表記が揺れている用語を発見するには「類似語検索」を用いる。その判定基準はレーベンシュタイン距離である。

2.6.4 用語の一覧を作成する

SLP 文書で使われている用語の一覧を作成するには「文要素のエクスポート」を用います。メンバー名と状態名の一覧は、厳密には「メンバー名」と「メンバー名と状態名の組」の一覧です。なぜなら、状態名はメンバー名に付随して存在する用語の為です。状態名はメンバー名ごとに別の名前空間を持っていると言い換えることもできます。

メンバー名と状態名の一覧には、「メンバー名」と「メンバー名と状態名の組」のほかに、それらの用語の使用頻度が出力されます。また、それらの用語が「文要素候補リストの作成」で登録されているかどうかや、「曖昧語」に該当するかどうかも併せて出力（エクスポート）されます。
出力ファイルは CSV 形式ですので Microsoft Excel 等で開いて見ることがきます。
また、このファイルを入力（インポート）することで、ほかの SLP 文書に対してメンバー名と状態名を取り込むことができます。

FN-05 SLP 文書で使われている用語の一覧を作成するには「文要素のエクスポート」を用いる。

NM-17 状態名はメンバー名に付随して存在する用語である。すなわち、状態名はメンバー名ごとに別の名前空間を持っている。

2.6.5 構文化決定表を作成する

構文化決定表は、SLP 文書の論理記述欄に書かれている、条件（if 構文と switch 構文）と結論（Do 構文）との関係を、大胆に捨象した形式でまとめたものです。 「主語（目的語）と述語から成る構文化決定表」には論理記述欄の記述のうち文要素が反映され、他の入力エリア（助詞、助動詞、コメント）は構文化決定表には反映されません。

FN-06 構文化決定表は条件（if 構文と switch 構文）と結論（Do 構文）との関係の記述のみに特化したものである。

LD-17 構文化決定表には文要素が反映される。他の入力エリア（助詞、助動詞、コメント）は反映されない。

2.7 SLP 文書を印刷する

SLP 文書は「印刷設定」メニューにより、選択された項目（複数可）を印刷することができます。
ただし、機能内容欄ではオブジェクトのリンクと埋込み(OLE)に対応していますが、貼り付けられた画像データは印刷されないことに注意してください。

機能内容欄の画像データを印刷したいときは、ファイルメニューの[他ファイルへの出力]-[機能内容欄のエクスポート]機能を使用して、他のアプリケーションで開いて印刷してください。
（画像データが、正常に印刷されます。）

3 SLP を開発プロセスに取り入れる

3.1 要求仕様書を初めから SLP で書く

要求仕様書を初めから SLP で書くことも可能です。要求仕様書が、存在しないか、存在しても何らかの理由でまったく新たに書き直す必要がある場合には、最初から SLP で要求仕様書を書くことを推奨します。SLP に慣れるまでは、紙またはテキストファイルなどに下書きをしてから SLP 文書を書き始める方がよいかもしれません。慣れてくれば、SLP を用いて、下書きから正式なものまであらゆる文書を作成することが可能になります。

GN-05 既存の要求仕様書が存在しないか、質が悪すぎる場合には、最初から SLP で要求仕様書を書く。

GN-06 SLP に慣れていないうちは、紙またはテキストファイルなどに下書きをしてから SLP 文書を書き始める。

3.2　従来の要求仕様書と SLP を併用する

3.2.1　従来の要求仕様書を階層的に整理する

2.1.3 節「要求を自由に記述する」で既に説明しましたが、単位機能の機能内容欄から既存の要求仕様書にリンクを張ることができます。そのことにより、複数のファイルに分散した要求仕様書を、SLP 文書を中心にして階層的に管理することができます。

3.2.2　従来の要求仕様書をもとに SLP 文書を作成する

従来の要求仕様書をもとに SLP 文書を作成する方法には、既述のように機能内容欄を使う方法と、論理記述欄を使う方法とがあります。機能内容欄を使う方法とは、2.1.3 節「要求を自由に記述する」で説明したように、機能内容欄を従来の要求仕様書と同じように記述して、それをインデント目次、階層目次または水平目次を使って階層的に整理することです。

一方、論理記述欄を使う方法では、既存の要求仕様書の内容を、SLP 文法に当てはめて記述し直す必要があります。そのためには、以下のような思考過程をとることが推奨されます。

1. 条件（if 構文）と結論（Do 構文）に分割する。
2. 主語または目的語（メンバー名）と述語（状態名）に分割する。

ここまでの過程で、従来の要求仕様書の記述を SLP 文書の論理記述欄に移すことができます。ここまでを SLP 文書の「下書き」として、さらに続けて、SLP 文書の用語を整理します。

まず、2.1.2 節の命名規則に従って、メンバー名、状態名、単位機能名の用語を整えます。さらに、2.3 節で説明した手法で SLP 文書を分析して、不十分なところがあれば修正します。

> LD-18 既存の要求仕様書を SLP の論理記述欄を使って書き直すには、まず条件と結論に分割し、次に主語または目的語と述語に分割する。続いて、用語の整理と SLP 文書の分析を行う。

3.2.3　SLP 文書から従来の要求仕様書を作成する

SLP 文書から従来の要求仕様書を作成するには、「通常文に変換」と「機能内容欄のエクスポート」を組み合わせる方法があります。まず、「通常文に変換」を行うと、論理記述欄の記述を自然言語に変換して、その結果が機能内容欄に貼り付けられます。続いて、「機能内容欄のエクスポート」を行うと、SLP 文書のすべての単位機能の機能内容欄を結合して、その結果をリッチテキスト形式のファイルに出力することができます。最後に、リッチテキスト形式のファイルを Microsoft Word などで体裁を整えれば、従来の要求仕様書と同様なものを得ることができます。

> FN-07 SLP 文書から従来の要求仕様書を作成するには、「通常文に変換」と「機能内容欄のエクスポート」を組み合わせる。

3.3　既存の SLP 文書を修正する

要求仕様書を SLP の論理記述欄を使って記述している場合、既存の要求仕様書を変更または加筆することは、他の形式の要求仕様書（Word、Excel など）よりも容易です。if 構文と Do 構文を用いることで、要求仕様書の構造が明確になっている為です。

> GN-07 要求仕様書を SLP の論理記述欄を使って記述しているならば、既存の要求仕様書を変更または加筆することは、他の形式の要求仕様書よりも容易である。

3.4　他の文書（または成果物）と SLP 文書との対応関係を記録する

3.4.1 SLP 文書から原要求を参照する

上流の要求仕様書が SLP 文書とは別に存在するとき、SLP 文書の記述が上流の要求仕様書のどの記述を根拠としているかを記録することができます。そのためには、単位機能ヘッダー欄の「原要求項目識別子」テキストボックスと「原要求項目識別子の変更」ウィンドウを用います。まず、上流の要求仕様書には、項目ごとに何らかのラベルを振っておきます。そして、SLP の側では、単位機能ごとに、その単位機能が上流の要求仕様書のどの項目を根拠としているか、先ほどのラベルを「原要求項目識別子」に追加することで記録します。それぞれの単位機能には複数の原要求項目識別子を記録できます。また、親単位機能の原要求項目識別子は、子単位機能の原要求項目識別子としても表示されます。

FN-08 SLP 文書の記述が上流の要求仕様書のどの記述を根拠としているかを記録するには「原要求項目識別子」を用いる。

SLP 文書に記録されている原要求項目識別子はトレーサビリティ・マトリックスという形式で一覧することができます。また、原要求項目識別子を変更する方法として、「原要求項目識別子」ウィンドウの他に、トレーサビリティ・マトリックスを経由する方法があります。以下は手順です。

1. SLP でトレーサビリティ・マトリックスをエクスポートする。
2. Microsoft Excel 等の適切なアプリケーションでトレーサビリティ・マトリックスを編集する。
3. SLP でトレーサビリティ・マトリックスをインポートする。

FN-09 原要求項目識別子を一覧するには「トレーサビリティ・マトリックス」を用いる。

3.4.2 下流工程の文書またはコードから SLP 文書を参照する

前節では、上流の要求仕様書の各項目にラベルを付けて、それを SLP の側から「原要求項目識別子」として参照する方法を説明しました。この節では逆に、SLP 文書の各単位機能にラベルを付け、そのラベルを下流工程の文書またはコードから参照する方法を説明します。

SLP 文書の単位機能にラベルを付けるには、単位機能ヘッダーの「ユニークラベル」テキストボックスと「ユニークラベル」ウィンドウを用います。インデント目次と単位機能名は、ある時点においてはユニークですが、ユーザーが SLP 文書を編集してこれらを変化させることができるため、外部から参照するためのラベルとしては使いにくい面があります。一方、ユーザーがユニークラベルを変化させようとすると、SLP は警告メッセージを発します。そのため、ユニークラベルは単位機能に対する恒久的なラベルとして使用できます。

FN-10 SLP 文書の単位機能に恒久的なラベルを付けるには「ユニークラベル」を用いる。

3.5　VDM（Vienna Development Method）と併用する

SLP の文要素は、VDM（Vienna Development Method）の型の定義に反映させることができます。例えば、メンバー名「信号機の色」が状態名「赤」「黄」「青」を持ち、メンバー名「信号機の動作状況」が状態名「正常」「停電」「故障」を持つとき、VDM の型の定義は以下のように記述することを推奨します。

```
types
信号機の色型 = <赤> | <黄> | <青>;
信号機の動作状況型 = <正常> | <停電> | <故障>;
信号機型::
        色: 信号機の色型
        動作状況: 信号機の動作状況型;
```

GN-08 SLP の文要素は、VDM の型の定義に反映させることができる。

3.6　SLP をテストに使用する

SLP をテストに使用するには「テスト仕様原案作成」を用います。これは、SLP の論理記述欄の記述からテスト仕様書を自動的に作成する機能です。SLP 文書は自然言語で書かれた要求仕様書に比べて用語の統一、および論理整合性の面で優れています。そのため、本機能は、テスト技法のひとつである CPM（コピー&テスト&モディファイ）法を利用した場合に特に効果を発揮します。また、テスト仕様原案を自動生成することにより、テスト仕様書のフォーマットが統一され、かつ要求仕様とのトレーサビリティを保てるため、管理が容易になる利点もあります。ただし、自動生成されたテスト仕様書をそのままテストに使えるかどうかは、用途と状況によります。例えば、テストを意識した記述を行う、論理記述欄で表現しきれない機能は機能内容欄に記述する、SLP から自動生成されたテスト仕様書を実際にテストできるように読み換える等の作業が必要になるかもしれません。テスト方法については、要求の作成者とテストの実施者との間で事前に共有しておくことが重要です。

3.6.1　テスト仕様原案作成時の注意事項

テスト仕様原案は「確定」の単位機能を対象として作成されます。
「確定」とはインデント目次で見たときに、■で示されている単位機能項目です。
■以外（灰色、白色）で示されている「暫定」や「未定」の単位機能については、Fn 構文を用いてそこに目次と同じ単位機能を記述したり、Fn 構文を適切な行に移動することで「確定」にすることができます。

FN-11 SLP をテストに使用するには「テスト仕様原案作成」を用いる。

GN-09 SLP から自動生成されテスト仕様書を実際にテストできるように、要求の作成者とテストの実施者との間でテスト方法を事前に共有しておくことが望ましい。

例えば、あるソフトウェアが Windows Vista, 7, 8 で正常に動作することを確認したい場合、以下のような SLP 文書を作り、「テスト仕様原案作成」を実行します。

```
switch <Windows のバージョン>が
case{Vista}である
   Fn[ ○○せよ ] (1)
case{7}である
   Fn[ ○○せよ ] (1)
case{8}である
   Fn[ ○○せよ ] (1)
elsw
```

```
    (:他のバージョンはテストの対象外とする。
    Do nothing
endsw
```

上記は単位機能「Windows のバージョンにかかわらず○○せよ」の論理記述欄の内容です。ここで、単位機能「○○せよ」には実際の動作を記述します。

> GN-10 条件だけを変えて結果が変わらないことを確認するテストをするには、同じ単位機能を条件を変えて複数回 Fn 構文で呼び出すようにする。

4　アドバイス一覧

4.1　名前の付け方

NM-01 メンバー名は名詞（名詞句、名詞節でもよい）にする。

NM-02 メンバー名は文書全体で一意な名前にする。

NM-03 メンバー名は長さにこだわらず、一意な名前になるまで修飾語を加える。

NM-04 「状態を持つもの」をメンバー名とする。

NM-05 量化記号を含むメンバー名を使うときは、「すべての男と女」のように書かず、「すべての男」と「すべての女」に分ける。

NM-06 文字通り「状態の名前」として記述できるならば、状態名を名詞（名詞句、名詞節でもよい）にする。

NM-07 状態名を名詞で表現できないときは、状態名を動詞にする。ただし、その動詞がサ行変格活用であるときは、語幹までを状態名にして、活用語尾を省く。

NM-08 状態名を名詞またはサ行変格活用の動詞で表現できないとき、if 構文では、「0 より大きい」などのように平叙文で状態名を記述する。

NM-09 状態名を名詞またはサ行変格活用の動詞で表現できないとき、Do 構文では、「閉じよ」などのように命令形で状態名を記述する。

NM-10 単位機能名は動詞の命令形で命名する。

NM-11 従来の要求仕様書の階層構造にとらわれず、独自の単位機能名を付ける。

NM-12 単位機能名は文書全体で一意にする。

NM-13 単位機能名を命名するとき、一意な名前が容易に思いつかないときは、親単位機能の名前に条件を表す修飾語を加えて子単位機能を命名する。

NM-14 論理記述欄ではなく機能内容欄を中心に使うときには、単位機能名は、従来の要求仕様書と同じように、体言止めで命名する。

NM-15 同じ事柄を意味するメンバー名または状態名は、正確に同じ文字列で記述する。

NM-16 メンバー名と状態名を除く入力エリア（すなわち、助詞、助動詞、コメント）は、単に人間にとっての可読性のためにある。

NM-17 状態名はメンバー名に付随して存在する用語である。すなわち、状態名はメンバー名ごとに別の名前空間を持っている。

4.2 論理記述欄を記述する方法

LD-01 「if」「Do」「Fn」の 3 種類の構文は不可欠である。他の構文は用途によっては便利である。

LD-02 if 構文を使用するときは、肯定側と否定側の両方を記述するよう心掛けることで、要求を網羅的に記述することができる。

LD-03 if 構文の助詞（メンバー名と状態名の間）は「が」にする。

LD-04 if 構文では、メンバー名を主語、状態名を述語にする。

LD-05 Do 構文は「○○せよ」という命令を表す。

LD-06 Do 構文を「○○である」という平叙文に用いる。

LD-07 Do 構文の中置助詞（メンバー名と状態名の間）は「を」にする。

LD-08 Do 構文の後置助詞（状態名の後）は「せよ」にする。

LD-09 Do 構文では、メンバー名を目的語、状態名を述語にする。

LD-10 論理記述欄の記述が長くなったり、ネストが深くなったりしたとき、Fn 構文を使うことで、記述の一部を他の単位機能に移すことができる。

LD-11 直前の状態名が名詞のとき、if 構文の助動詞は「である」「になった」などとする。

LD-12 直前の状態名が名詞のとき、Do 構文の後置助詞は「にせよ」「である」などとする。

LD-13 直前の状態名がサ行変格活用の動詞のとき、if 構文の助動詞は「した」「している」などとする。

LD-14 直前の状態名がサ行変格活用の動詞のとき、Do 構文の後置助詞は「せよ」にする。

LD-15 直前の状態名がサ行変格活用でない動詞のとき、if 構文の助動詞は空にする。

LD-16 直前の状態名がサ行変格活用でない動詞のとき、Do 構文の後置助詞は空にする。

LD-17 構文化決定表には文要素が反映される。他の入力エリア（助詞、助動詞、コメント）は反映されない。

LD-18 既存の要求仕様書を SLP の論理記述欄を使って書き直すには、まず条件と結論に分割し、次に主語または目的語と述語に分割する。続いて、用語の整理と SLP 文書の分析を行う。

4.3 SLP の機能の紹介

FN-01 SLP 文法で要求を記述するには論理記述欄を用いる。

FN-02 矛盾検査の結果をユーザーが制御するには、「メンバー属性無矛盾化設定」と「文どうしの関係定義」を用いる。

FN-03 使用を禁止または制限したい用語を管理するには「曖昧語検索」を使用する。

FN-04 表記が揺れている用語を発見するには「類似語検索」を用いる。

FN-05 SLP 文書で使われている用語の一覧を作成するには「文要素のエクスポート」を用いる。

FN-06 構文化決定表は条件（if 構文と switch 構文）と結論（Do 構文）との関係を捨象したものである。

FN-07 SLP 文書から従来の要求仕様書を作成するには、「通常文に変換」と「機能内容欄のエクスポート」を組み合わせる。

FN-08 SLP 文書の記述が上流の要求仕様書のどの記述を根拠としているかを記録するには「原要求項目識別子」を用いる。

FN-09 原要求項目識別子を一覧するには「トレーサビリティ・マトリックス」を用いる。

FN-10 SLP 文書の単位機能に恒久的なラベルを付けるには「ユニークラベル」を用いる。

FN-11 SLP をテストに使用するには「テスト仕様原案作成」を用いる。

4.4 全般的な注意事項

GN-01 SLP をより手軽に使うには、論理記述欄ではなく機能内容欄を主体に使っていく。

GN-02 既存の要求仕様書と同じように、機能内容欄に自由に書いて、それをインデント目次または階層目次を使って階層的に整理することができる。

GN-03 OLE オブジェクトをファイルへのリンクにすることで、複数のファイルに分散した要求仕様書を、SLP 文書を中心にして階層的に管理することができる。

GN-04 loop 構文、for 構文、while 構文を使用した場合、ループの回数に依存した検査は行われない。

GN-05 既存の要求仕様書が存在しないか、質が悪すぎる場合には、最初から SLP で要求仕様書を書く。

GN-06 SLP に慣れていないうちは、紙またはテキストファイルなどに下書きをしてから SLP 文書を書き始める。

GN-07 要求仕様書を SLP の論理記述欄を使って記述しているならば、既存の要求仕様書を変更または加筆することは、他の形式の要求仕様書よりも容易である。

GN-08 SLP のメンバー名と状態名は、VDM の型の定義に反映させることができる。

GN-09 SLP から自動生成されたテスト仕様書を実際にテストできるように、要求の作成者とテストの実施者との間でテスト方法を事前に共有しておくことが望ましい。

GN-10 条件だけを変えて結果が変わらないことを確認するテストをするには、同じ単位機能を条件を変えて複数回 Fn 構文で呼び出すようにする。

4.5 操作方法

UI-01 キーボードの F1 キーを押下するとオンラインヘルプが表示される。

UI-02 論理記述欄に構文を追加するには右クリックメニューを用いる。

UI-03「検査・検索等結果」ウィンドウの「矛盾の可能性があります」という警告メッセージをダブルクリックすると、すぐに「メンバー属性無矛盾化設定」ウィンドウが開く。