ソフトウェア工学 98−12
（1994. 5. 25）

組み込み型マイクロプロセッサ用ソフトウェアにおけるコストモデル

藤村　直美
九州芸術工科大学

ソフトウェアの生産性を向上するためには、生産性に影響を与える要因が明確であること、それらを定量的に計測・評価できることが大前提である。これまで大規模なソフトウェア開発についてはいくつかのコストモデルが提案されている。しかしながら高機能、高品質、小規模、小人数で短期間に開発といった特徴を有する組み込み型マイクロプロセッサ（MP と略す）用ソフトウェアの生産形態に対応するコストモデルはほとんど検討されていない。ここでは数年間に渡って行なったアンケート調査に基づいて、組み込み型 MP 用ソフトウェアにおけるコストモデルを提案し、CPU ビット、記述言語、製品用途による影響などについても議論する。

A Cost Model for Software in Embedded Microprocessos

Naomi Fujimura
Kyushu Institute of Design
Fukuoka, 815 Japan

It is important to measure and evaluate software productivity quantitatively. Several cost models were proposed for large scale software projects. However, few cost models have been proposed yet for small scale software projects in which a few people develop software in a short period as in embedded microprocessors. We have been collecting data about software productivity in embedded microprocessors. We proposed a new cost model for the small scale software projects. We also discuss the characteristics from the view point of CPU bits, description languages, and product purposes in detail

1 はじめに

高機能、高品質のソフトウェアを効率良く生産できることは重要である [1]。一般的に工業製品の製造において生産性を向上するためには生産性に影響を与える要因が明確であること、それらを定量的に評価できることが大前提である。しかしながらソフトウェアは、従来の工業製品と異なり、人間の知的精神活動によってのみ実現される部分が多いために、生産性に影響を与える要因が明確でなく、また定量的に評価することも容易ではない。そうした中で、大規模なソフトウェアの生産性を定量的に評価するための試みとして、COCOMO モデル [2, 3] や Putnam モデル [4] のようなコストモデルが 1970 年代から提案されている [5]。

一方、組み込み型マイクロプロセッサ (MP と略す) 用のソフトウェアは小規模、小人数で短期間に開発という特徴を有し、製品の種類も数量も多く、この分野におけるソフトウェアの生産性向上は社会的にも経済的にも大きな影響を与える。しかしながらこれらの生産形態に対応するコストモデルはこれまでほとんど検討されていない。従来のコストモデルは大規模ソフトウェアを対象にしていること、提案されてから時間が経過していること、開発環境も変化していることなどから、現在の組み込み型 MP 用ソフトウェアにおいても適用可能であるかどうか全く不明である。

ここでは組み込み型 MP 用ソフトウェアの開発に関したアンケート調査を行い、それらの調査結果に基づいて、組み込み型 MP 用ソフトウェアに適用可能なコストモデルを提案している [6, 7, 8, 9]。これによって従来は困難であった組み込み型 MP 用ソフトウェア開発における生産性を定量的に議論することを可能にする。さらにここでは CPU ビット、記述言語、製品用途によるコストモデルの相違についても検討する。

2 調査とデータ

2.1 アンケート調査

本研究で行なったアンケート調査の調査項目は表 1 の通りである。ソフトウェアの生産性に関連する生産期間と生産高は次のように考える [10]。まず、生産期間は表 2 のようなライフサイクルを考え、このうちのシステム設計から総合テストまでを生産期間とする。ソフトウェアのライフサイクルは JIS や ISO 等で定義されているばかりでなく [11]、他にも各種の定義が提案されている。しかしながらこれらの定義は類似してはいるが、統一されておらず、しかも組み込み型 MP 用ソフトウェアの開発現場の実体に必ずしもあっていない。

表 1: 調査項目

CPU ビット	用途	見積もり工数
実績工数	開発期間	ソフトウェアの規模
記述言語	再利用率	開発環境
プロジェクト特性	その他	

表 2: ライフサイクルの定義

工程	内容
設計	要求分析＋システム設計
製作	プログラム作成
試験	単体テスト＋組み合わせテスト＋総合テスト
保守	保守＋廃棄

次に生産高として、ここでは注釈行を除いたソースプログラムの行数を採用する。これは直感的に分かり易く、またアンケート調査において回答し易いことを考慮したためである。実際にはすべてのプログラムを新しく開発する場合ばかりでなく、既存のプログラムを再利用する場合もある。そこで完成規模と開発規模は次のように考えるものとして、調査用紙に明確な説明を付けた。

$$\text{完成規模} = \text{開発規模} + \text{流用行数}$$
$$\text{開発規模} = \text{新規行数} + \text{改造行数}$$

このうち流用行数とは既存のプログラムを全く修正せずにそのまま再利用した行数、改造行数とは既存のプログラムを部分的に修正して再利用した行数とする。改造の場合はプログラムの中身を理解する必要があるので、開発に要する工数は新規開発に準じて考える。

調査用紙は組み込み型 MP 用ソフトウェアを開発している各社（100～200 社、年度による）に発送した。参考として回答の回収件数を表 3 に示す。なお 1993 年度はアンケートの調査主体が変わったために回収件数が減少している。

表 3: 回答件数

年度	回答件数
1990	357
1991	490
1992	394
1993	130
90-93	1371

2.2 データと検証

コストモデルを提案するためにはソフトウェアの規模、工数、開発期間のデータが必要である。開発期間は回答をそのまま利用して特に問題はないが、規模と工数は次に述べるように補正が必要である。

ソフトウェアの規模では再利用率が問題になる。再利用率を (完成規模 − 新規開発)/完成規模 と定義する。この値が 100 %に近い方が再利用率が高い。例えば 1992 年度では、既存のソフトウェアを再利用せずにすべて新規に開発したという回答 (流用、改造ともに 0 行のもの) は 134 件 (34 %) であり、残りは程度はともかくとして何らかの再利用を行なっている。全く新規開発を行なわずにすべて既存ソフトウェアの改造と流用で済ませたという回答も 16 件 (4 %) ある。これらのことから組み込み型 MP 用ソフトウェアの開発においては再利用が活発に行なわれていることが分かる。したがってソフトウェアの規模として完成規模（出荷行数）を選択すると実績工数に比べて相対的に大規模なソフトウェアを開発していることになり、具合が悪い。ここではソフトウェアの規模は開発規模とする。

次に工数であるが、『人月』で回答してもらっている実績工数の基準が各社ともまちまちである。各社における 1 人月の分布の様子を図 1 に示す。この図から一人当たり 160 時間の労働時間を 1 人月として計算しているところが多く、ついで、200 時間、180 時間が多いことが分かる。ここでは 1 人月は 160 時間とみなして、すべての実績工数を換算する。

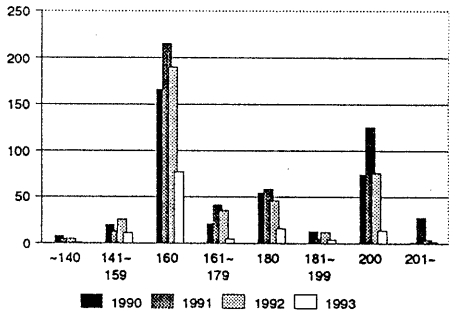

図 1: 1 人月の時間数の分布

最後に上述の補正を行なった上で、1 人月当たりの開発規模（これを開発効率と呼ぶ）を検討する。表 4 に開発効率の基本統計値を示す。これによると 1 人月で 20 行も開発が進まないものや数十キロ行のソフトウェアを開発したことになるものなど、常識的には考えられないような回答がアンケートの回答には含まれていることを示している。そこで開発効率の対数値を適当な区間で棒グラフにして表示すると図 2 のようになる。図 2 から開発効率の対数値は直感的には正規分布をしているように見える。これがどの程度正規分布に近いかを検定するために、開発効率の対数値を正規 QQ プロット図として表示する。

表 4: 開発効率の基本統計値

年度	平均	標準偏差	最小	最大
1990	1.45	2.95	0.073	40.7
1991	1.96	7.08	0.016	111.1
1992	1.64	4.66	0.008	61.6
1993	1.38	1.86	0.058	10.7
90-93	1.68	5.18	0.008	111.1

単位：千行／人月

ここでは例として 1992 年度の正規 QQ プロット図を図 3 に示す。図 3 から判断して、開発効率の対数値は平均を中心に標準偏差の 2 倍以内の

データがほぼ正規分布にしたがっていると考えられる。他の年度においてもほぼ同様である。したがって今後の議論においては開発効率の対数値が平均から標準偏差の2倍以内に限定したデータに基づいて解析を行なう。

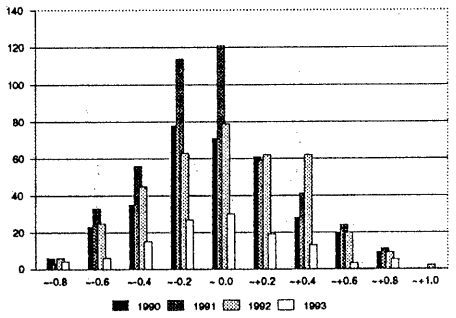

図 2: 開発効率の分布

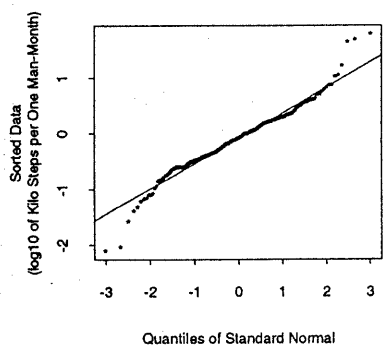

図 3: 開発効率の正規 QQ プロット図

対数をとって散布図として表示したものを図5に示す。他の年度のデータにおいてもほぼ同様の図になるのでここでは省略する。これらの図において、"+" で示している点が開発効率の値に基づいて解析から除外したデータである。

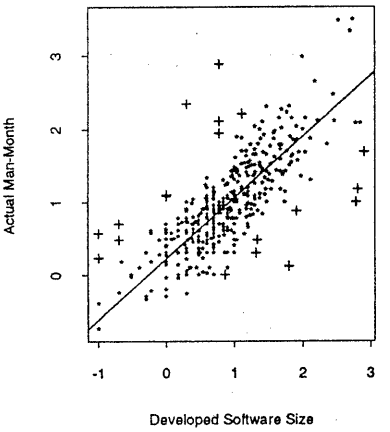

図 4: 規模と実績工数の散布図 (1992)

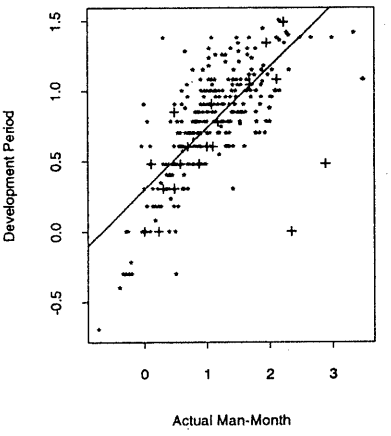

図 5: 実績工数と開発期間の散布図 (1992)

3 コストモデル

3.1 コストモデルの提案

例として 1992 年度における開発規模と実績工数をそれぞれ対数をとって散布図として表示したものを図4に示す。同様に 1992 年度におけるソフトウェア開発の実績工数と開発期間をそれぞれ

図4と図5を見ると、それぞれ二つの変数間には強い関連があることが分かる。したがってこれ

らの図からそれぞれ1次回帰直線の係数を求めることで、組み込み型MP用ソフトウェアのコストモデルを次のような式として表せることが分かる。

$$M = a \times S^b$$
$$T = c \times M^d$$

ここで、Sは出荷ソース行数、Mは開発工数（人月）、Tは開発期間、a、b、c、およびdはプロジェクトの困難度で決まる生産性に関連した係数である。このコストモデルをJEIDAモデルと呼ぶことにする。COCOMOのBASICモデルに対応するJEIDAモデルの係数a、b、c、およびdを表5に示す。

表5: 新しいコストモデルにおける係数

年度	件数	a	b	c	d	DE	AM
90	331	1.95	0.76	2.44	0.41	1.05	1.85
91	462	1.92	0.82	1.71	0.51	1.00	2.62
92	373	1.66	0.84	1.98	0.44	1.18	4.26
93	122	2.00	0.79	1.97	0.50	1.09	2.29
90–93	1291	1.87	0.81	1.98	0.46	1.07	2.84

DE：開発効率、AM：平均開発人員

ここではJEIDAモデルとCOCOMOのBASICモデル（表6参照）を比較して議論する。表5の1990 ～ 1993年の各年度の係数と表6のOrganicモデル（係数の値がJEIDAモデルの対応する値に一番近い）の係数を元に、図6と図7にそれぞれ規模と実績工数、実績工数と開発期間の関係をグラフ化したものを示す。これらの値はCPUビット、記述言語などによって変動するが、詳細は後述するので、ここでは詳しい議論をしない。

表6: COCOMO BASICモデルにおける係数[2]

モード	a	b	c	d
ORGANIC	2.4	1.05	2.5	0.38
SEMIDETACHED	3.0	1.12	2.5	0.35
EMBEDDED	3.6	1.20	2.5	0.32

図6から、数十キロステップ以上のソフトウェア開発において、COCOMOモデルではJEIDAモデルに比べて多大な実績工数を必要としていることが分かる。これはJEIDAモデルが比較的に小規模のソフトウェアを対象としておりCOCOMOモデルが大規模のソフトウェアを対象としているというだけでなく、二つのモデルがそれぞれ構築された時代におけるソフトウェアの開発技術や環境の違いも影響していると考えて良さそうである。一方、実績工数と開発期間の関係は図7からJEIDAモデルとCOCOMOモデルにあまり大きな差がないことが分かる。これは製品の納期や投入可能な人員といった制約から、必要な工数と開発期間との関係が両方のコストモデルにおいて類似の経営判断や工程管理の元で開発されていることを示唆している。

表5には開発効率(DE)と平均開発人員(AM)も示している。ここで開発効率(DE)の値は1.0を少し超える程度であることから1人月で開発できるソフトウェアの平均的な大きさは千行少しであることが分かる。平均開発人員はソフトウェアの開発期間を通じてプロジェクトにかかわった平均の要員数を表しており、この値は2名弱から毎年増加して4名強になっている。したがって毎年、ソフトウェアの規模が大きく、複雑になる傾向を有することが分かる。

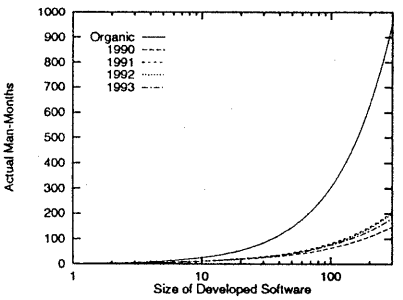

図6: ソフトウェアの規模と実績工数の関係

3.2 コストモデルの詳細

JEIDAモデルについて、CPUビット別、記述言語別、製品の分野別に分類した場合の特徴について検討する。CPUビット別のコストモデルの

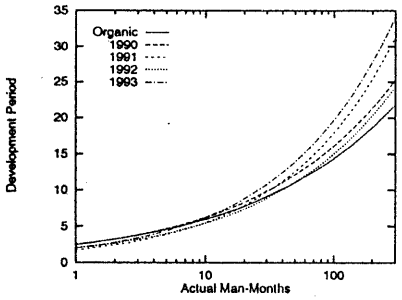

図 7: 実績工数と開発期間の関係

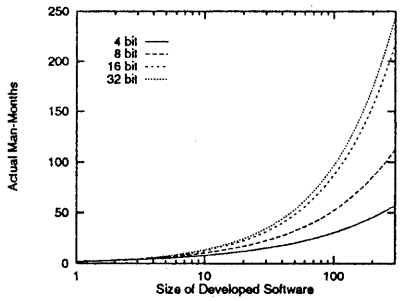

図 8: CPU ビット別ソフトウェアの規模と実績工数の関係

係数 a ～ d を元に規模と実績工数、実績工数と開発期間の関係をグラフ化したものを図 8 と図 9 に示す。記述言語別のコストモデルの係数 a ～ d を元に規模と実績工数、実績工数と開発期間の関係をグラフ化したものを図 10 と図 11 に示す。製品用途別のコストモデルの係数 a ～ d を元に規模と実績工数、実績工数と開発期間の関係をグラフ化したものを図 12 と図 13 に示す。なお、図 12 と図 13 では線種を分かりやすくするために、図の右端 (300 キロ行) における値の大きさの順に線種の説明を付けている。

図 8 と図 9 から CPU ビットによってそれぞれ規模と実績工数、実績工数と開発期間が異なる特徴を示していることが分かる。図 8 から同じ規模のソフトウェアなら CPU のビット数が増えるに従って工数が多く必要になることが分かる。一方、図 9 から工数と開発期間は 4 ビットと 8 ビットは 16 ビット、32 ビットとは異なる関係を示しているが、16 ビットと 32 ビットはほとんど同じ関係を有することが分かる。これは 32 ビットになっても 16 ビットの時と同様な開発を行なっており、特に 32 ビット CPU の特徴を考慮した開発を行なってはいないことを暗示している。

図 10 と図 11から記述言語による規模と実績工数、実績工数と開発期間の関係はそれぞれ異なる特徴を示すことが分かる。図 10 からアセンブリ言語によるものと C 言語やその他の高水準言語を主体にして記述するものとで明確に異なる特徴を示しており、アセンブリ言語の方が同じ規模に対しては工数が少なくて済んでいることが分かる。

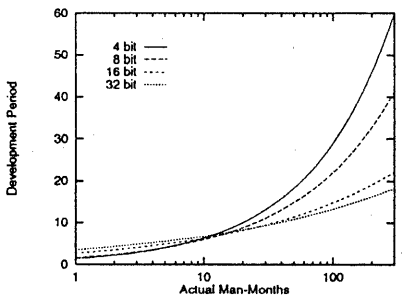

図 9: CPU ビット別実績工数と開発期間の関係

これは一見常識に反する結果のように見えるが、機能的にはアセンブリ言語の数ステップが C 言語の一ステップに相当することから特に異常な結果ではないと考えられる。一方、図 11から工数と開発期間は記述言語が異なるとそれぞれ異なる関係を示すことが分かる。およそ 30 人月以上では同じ工数に対してはアセンブリ言語で記述するものが開発期間が長くなっていることが分かる。これは平均開発人員 (AM) の割当が他の記述言語に比べて半分程度と少ないことから、必要な工数が大きくなると必然的に開発期間が長くなると考えて良さそうである。

図 12 から OA 関連 (OA)、通信関連 (Comm)、周辺端末 (Peri)、家電機器 (Home) の四分野は規模と工数の関係がそれぞれ異なる特徴を示しているが、制御関連 (Proc)、計測機器 (Meas)、その他 (Others) の三分野はほぼ同じ特徴を示しているこ

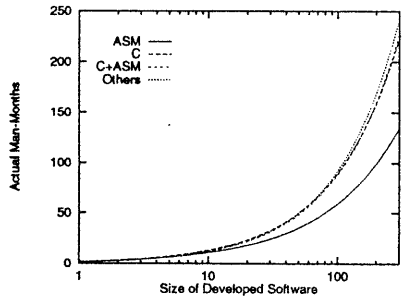

図 10: 記述言語別ソフトウェアの規模と実績工数の関係

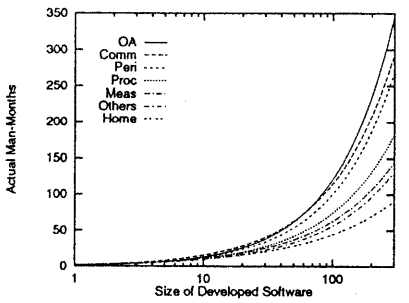

図 12: 製品分野別ソフトウェアの規模と実績工数の関係

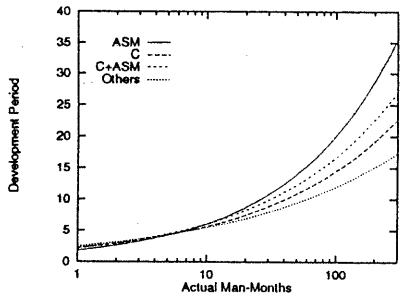

図 11: 記述言語別実績工数と開発期間の関係

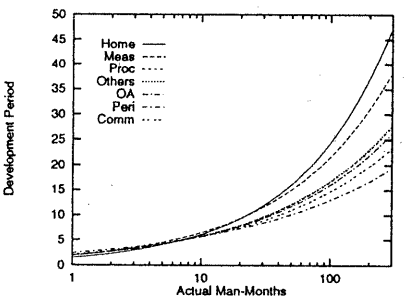

図 13: 製品分野別実績工数と開発期間の関係

とが分かる。一方、図 13 から家電機器 (Home) と計測機器 (Meas) を除いた他の五分野はほぼ同じ特徴を示しており、規模と工数の関係ほどばらばらになっていない点が興味深い。また開発効率 (DE) に大きな変動はないが、平均開発人員 (AM) は、3 年間の平均で 家電機器の 1.52 人から OA 機器の 6.14 人まで、製品分野によって大きく変動していることが分かる。

4 おわりに

これまで組み込み型 MP 用ソフトウェアにおけるコストモデルの可能性は十分に検討されていなかったが、本研究で JEIDA モデルと名付けた新しいコストモデルを提案することができた。これで組み込み型 MP 用ソフトウェアの生産性を定量的に議論するための基盤を実現できたと言える。今回の結果から、組み込み型 MP 用ソフトウェアのための JEIDA モデルでは従来のコストモデルとは異なる特徴を有すること、JEIDA モデルの係数は比較的安定した値を示しており、十分に実用になるものであることなども明らかにすることができた。

JEIDA モデルを使うことによって、見積り精度は粗いにしても、実際の現場においてソフトウェアの生産性に関連した予測や評価を行なうことができる。例えば、ソフトウェアの開発前に開発規模を見積ることができれば実績工数や開発期間を推定することができる。またここでは特定の組織に依存しないコストモデルの係数を示していることから、ソフトウェアの開発後に JEIDA モデルに基づいて開発規模から実績工数を、実績工数から開発期間を求めることによって、各社における

ソフトウェアの生産性の善し悪しを定量的に判断する目安として用いることも可能である。

JEIDA モデルの係数の信頼性を上げるには、今後も継続して調査を行なうこと、調査は行なっているがまだコストモデルに組み込んでいない開発環境やプロジェクトの特性をパラメータとして組み込むことなどが必要であろう。またこうしたコストモデルの係数の経年変化を把握することも重要であると考えており、今後も継続して調査を行なう予定である。

謝辞

本研究は文部省科学研究費重点領域研究 (1)(課題番号 02249110, 03235110, 04219110) によります。また日本電子工業振興協会のソフトウェア生産性尺度専門委員会 (1990～1992) との共同研究でもあります[12, 13, 14]。本論文をまとめるに当たって、九州大学工学部牛島和夫教授から有益なご助言を頂きました。ここに記して関係者の方々に謝意を表します。

参考文献

[1] 花田収悦：“ソフトウェア生産性の評価と管理”、情報処理、**21**、10、pp.1057 – 1064 (昭 55 – 10)

[2] B.W.Boehm ： “Software Engineering Economics”, IEEE Trans. on Software Enf., **SE – 10**, 1, pp.4 – 21 (Jan. 1984)

[3] B.W.Boehm ： “Software Engineering Economics”, Prentice – Hall (1981)

[4] L.H. Putnam and A. Fitzsimmons : “ Estimating Software Cost”, Datamation, pp.189 – 198 (Sep.), pp.171 – 178 (Oct.) and pp.137 – 140 (Nov. 1979)

[5] 大筆豊：“ソフトウェアのコストの見積り技術”、情報処理、**33**、8、pp.906 – 911 (平 4 – 08)

[6] 藤村直美：“組み込み型マイクロプロセッサ用ソフトウェア生産の現状”、情処学会第 39 回全国大会論文集、pp.1421 – 1422 (平 1 – 10)

[7] 藤村直美：“組み込み型マイクロプロセッサ用ソフトウェア開発の現状と生産性モデル式”、情報処理学会第 41 回全国大会論文集、**5**、pp.249 – 250 (平 2 – 09)

[8] Naomi Fujimura: “Software productivity in built – in microprocessors”, Microprocessing and Microprogramming, **28**, pp.169 – 172 (Feb. 1989)

[9] 藤村直美：“組み込み型マイクロプロセッサ用ソフトウェア開発の現状と生産性”、情報処理学会第 46 回全国大会論文集、**5**、255 – 256 (平 5 – 03)

[10] 上條史彦: “ソフトウェアのコストについて”、情報処理、**21**、10、pp.1050 – 1056 (昭 55 – 10)

[11] 菅忠義: “ソフトウェア工学における標準化動向”、情報処理、**28**、9、pp.1113 – 1126 (昭 62 – 09)

[12] 電子協：ME 知的ソフトウェア環境に関する調査報告書、pp.167 – 240 (平 3 – 03)

[13] 電子協：ME 知的ソフトウェア環境に関する調査報告書、pp.95 – 167 (平 4 – 03)

[14] 電子協：ME 知的ソフトウェア環境に関する調査報告書、pp.273 – 355 (平 5 – 03)