# 第 1 章

# 進化と環境

## 1.1 概要

システムが漸進的に開発される際，新しい利用者要求へ対応するために実際に変更されるのは，システムの中に定義されているクラスである．このとき開発者は，単に求められた新たな要求を満足させることを考えるのではなく，将来必要となるクラスの再利用性や拡張性を考慮した設計を行うことが多い．ここで，利用者の要求や開発者の設計意図といったオブジェクトの変更要因をオブジェクトの環境と考えると，オブジェクトは，これらの環境に対して適切な構造や仕様を持つように変更されているとみなすことができる．オブジェクトのこのような変化に対して，生物学のアナロジを適用するならば，オブジェクトは環境の変化に適応して進化すると言うことができる．

以下で，オブジェクトの進化のための環境，そして進化を定量化するための進化メトリクス，および進化率メトリクスを定義した後，本研究で用いた 3 つのシステムの概要と，進化メトリクスを適用した結果の一部を紹介する．

## 1.2 オブジェクトを進化させる環境

本研究では，オブジェクトを進化させる環境として次に示す二種類の環境を考える．

1. 利用者環境

M. Lehman と L. Belady は，利用者と相互作用を持つシステムを E 型システムと呼んだ [32]．このようなシステムでは，システムが利用者に提供されること自体が利用者の作業環境に変化をもたらし，新たな要求を発生させる原動力となる．そして，新たな要求を受け取ることによってシステムは再び進化する．これを利用者とシステムの相互作用と言う．本研究で調査対象としたシステムも E 型システムである．そこで，オブジェクトを進化させる「新しい要求がシステムに到着する」という事象が利用者の要求変化に由来する点に着目し，この事象が発生するオブジェクトの環境を利用者環境と呼ぶことにする．

2. 技術的環境

利用者環境が変化した際，開発者は，新しい要求に合致するようにオブジェクトを変更するだけでなく，将来の漸進的開発の生産性についても配慮した設計変更を行うことがある．たとえば，与えられた利用者環境の変化や将来予想される利用者環境の変化に対して，既存のクラスを拡張する，分割して新たなクラスを定義する，あるいは統合する，継承構造を変更するといった設計方針を決定する開発者の技術的な意図は，いずれもオブジェクトの進化に影響を与える環境となり得る．そこで，開発者の設計方針が変化するという事象が開発者の技術的意思決定に由来する点に着目し，この事象が発生するオブジェクトの環境を技術的環境と呼ぶことにする．

## 1.3 進化メトリクス

オブジェクト指向メトリクスに関しては 1990 年以降活発に研究が進められてきた [61]．特に Chidamber と Kemerer らが提案したオブジェクト指向メトリクス（以降 CK メトリクス) [8] は，以降のオブジェクト指向におけるメトリクスの研究に大きな影響を与えた．たとえば，Basili らはエラー発生度と CK メトリクスを用いた計測値との相関関係を示し [2]，Sharble らは，方法論の違いによる成果物を複雑度で比較する際に，CK メトリクスを適用した [52]．

また，Lorenz と Kidd らは，プロジェクト管理および設計の視点からメトリクスを定義し [37]，生産性や見積りへの適用，設計妥当性評価への適用を提案

している．Henderson-Sellers らは，従来のサイクロマティックバリューやファンクションポイントといったシステムの複雑度を評価するメトリクスを，オブジェクト指向へ適用する研究を行った [22]．このようなメトリクスに関する基礎的な研究が進むなかで，再利用率を定義してクラスの再利用率に対する費用対効果のシミュレーションを行った結果も報告されている [21]．中西らは，オブジェクトの操作の抽象化に要する労力と，それによって得られた効用を定量化するメトリクスを提案し，ET++，InterViews を用いて，改版による労力と効用の関係が変化する様子を時系列で示した [41]．

プロジェクト管理における見積り，品質管理へのメトリクスの適用は従来から進められてきた [13, 30, 25]．プロジェクト管理では，計測した結果を解釈し，スケジュールの見直しやリソースの再割り当ての意思決定を行うためにも使われている [56, 60]．

Lorenz と Kidd らはメトリクスの取扱いに関して，一般的な計測値と，ある時点のシステムの状態を計測した結果を比較してシステムやプロジェクトを評価する指針を示しているが，このような評価の方法には疑問がある．個々のプロジェクトが置かれた状況や対象問題領域の特徴も異なるため，開発中のシステムの評価を一般的な傾向と比較して評価するのは適切ではないだろう．本研究では，同一のシステム開発に対して，時系列で計測を行うことによって，その変化の傾向をオブジェクトやシステムの進化として捉え [44, 43]，その進化過程から設計上の問題点を発見することを試みた．進化過程を観測する対象には，次に挙げる 3 つのレベルを設定する．

1. システムレベル

   システムレベルでは，以下に示すメトリクスを用いた計測値の他に，システム内の全クラスや全メソッドに対して計測した値の分布や分布の変化の傾向を参照することで，システムの進化を捉えることが可能である．システムレベルの計測結果は，オブジェクトの進化過程を議論する際，システムがどのような進化過程にあったかを確認するためにも利用した．

   - NCD: クラス数 (the number of classes)
   - SLOC: 全クラスの行数をシステム全体で合計した値 (system lines of code)

- SNOM: 全クラスに定義されているメソッド数をシステム全体で総計した値 (the number of methods defined in the system)
- NMN: システムに定義されたメッセージの名前の総数 (the number of message names defined in the system)
- NCT: ライブラリクラスの直下に定義されているクラス継承木の数. (the number of class trees)
- NCBT: クラス継承木に属しているクラスの数 (the number of classes belonging to a class tree)

2. クラスレベル

Chidamber らは，クラスの可読性，変更容易性，結合度，凝集度を計測するためのメトリクスとして WMC (Weighted Methods Per Class)，DIT (Depth of Inheritance Tree)，NOC(Number of Children)，CBO(Coupling between object classes)，RFC(Response For a Class)，そして，LCOM(Lack of Cohesion in Methods) を定義した [8]．本研究では，オブジェクトの進化を定量的に表現できることを評価基準とし，CK メトリクスのうちの 4 つのメトリクスを進化メトリクスに取り入れた．

クラスの役割の変化を定量化するためには，WMC の代わりにクラスのメソッド数:CNOM を用いる．CNOM は，メソッドの重みづけを 1 とした場合の WMC と考えることもできる．

また，DIT については，直近のライブラリクラスから求めた継承の深さと定義する．CK メトリクスでは，DIT を最上位のクラスから数えた継承の深さと定義しているが，クラスの進化を議論する場合には適切ではない．DIT を用いてクラスの進化を議論する際には，技術的環境の変化が及ぶ範囲に着目して，その範囲の継承の深さを計測すればよいはずである．一般に，アプリケーションの開発者がクラスの継承木を変更する際，ライブラリクラスの仕様を変更することが許されていることは少なく，開発者が自ら定義したクラス継承木だけを設計変更の対象とすることが多い．したがって，進化メトリクスで用いる DIT では，深さの起点をアプリケーションのクラスから見た直近のライブラリクラスと定める．

クラスの結合度を定量化するメトリクスである CBO は，プログラムの稼働中に収集したプロファイルデータからオブジェクト間のメッセージ送受信の回数を集計することで，クラスではなく，インスタンス間の CBO を求めることができる．計測対象システムが Smalltalk のように型のないプログラミング言語で開発されている場合，ソースコードから求めることは困難であるため，CBO は，インスタンス間で交されるメッセージ送受信回数は動的メトリクスとして定義する必要がある．

- CLOC: クラスに定義されている全メソッドの行数をクラス内で合計した値 (class lines of code)
- NIV: クラスに定義されているインスタンス変数の数 (the number of instance variables)
- CNOM: クラスに定義されているメソッド数 (the number of method definitions)
- DIT: 直近のライブラリクラスから数えた継承の深さ (depth of the inheritance tree)
- NOC: 直下のサブクラスの数 (the number of children)
- 動的メトリクス：
  - NSM: 送信したメッセージ名と送信先別の送信回数 (the number of sent messages)
  - NRM: 受信したメッセージ名と発信元別の受信回数 (the number of received messages )

3. メソッドレベル

クラスの役割を表すメソッドの規模は，クラスの規模にも大きな影響を与える．クラスの行数は，ここで定義されるメソッドの行数:MLOC をクラス内で合計した値である．行数の数え方については，Lorenz らが設計を評価するためのメトリクスの中で議論しているように [37]，メッセージ送信文の数を行数とするものなどがある．本研究で計測した行数は，計測方法が簡単な改行の数と定義し，次のメトリクスを用いてオブジェクトの進化過程を定量化することにした．

- MLOC: メソッドの行数 (lines of code of a method)

本論文では，オブジェクトの進化とは，主にクラスの進化を指し，メッセージの進化および動的メトリクスを用いたオブジェクトの進化は，本論文の対象外となっている．また，計測対象システムは Smalltalk で開発されていたため，Smalltalk のメタクラス機構を用いて計測を行うことができた [31].

## 1.4 進化率メトリクス

クラスの進化を解析する際，クラスの役割に着目したクラスの時系列の進化率を用いる．クラスの進化率を以下に定義する．

$$\rho CNOM_i = \frac{CNOM_i - CNOM_{i-1}}{CNOM_{i-1}} \log_{10} CNOM_{i-1}$$

ここで，$CNOM_i$ は，第 i 版のクラスのメソッド数を表す．役割に着目したクラスの進化率は，第 i 版で計測したメソッド数 $CNOM_i$からひとつ前の版のメソッド数 $CNOM_{i-1}$を引き，その値を $CNOM_{i-1}$で割った後，$CNOM_{i-1}$の $\log_{10}$を掛けた値と定義する．

$\log_{10}$を乗算しているのは，$CNOM_{i-1}$の値が小さいとき全体の変化率が大きくなるのを防ぐためである [45].

## 1.5 調査対象システムの概要

オブジェクトの進化を調査する対象として選択したシステムは，次の 3 システムである．

1. SystemA: 熱交換シミュレーションシステム

   SystemA は 1 人の開発者によって開発され，そのリリース間隔は 1 か月であった．利用者はシステム試用後に要求変更を提出し，再び 1 か月後にリリースされたシステムを評価した．したがって，開発は漸進的に進められたと言うことができる．進化のためのデータは，利用者へ提供された版を対象に集計した．

2. SystemB: 入金消し込みシステム

   SystemB は 1 人の開発者によって開発され，そのリリース間隔は 2 か月であった．開発者は，数週間の試用期間後に利用者から要求変更を受け取り，次の開発を進めた．したがって，SystemB も漸進的に開発されたとみなすことができる．

3. SystemC: 証券管理システム

   SystemC は 4 人の開発者によって開発された社内開発システムである．開発途中の要求変更はなかったので，このシステムの開発形態は落水型と言うことができる．

以上の 3 システムはいずれも VisualSmalltalk を用いて開発されていた．次に，システムの開発状況を詳しく説明する．

### 1.5.1 SystemA

SystemA は全部で 6 版の計測対象を得ることができ，第 0 版のクラス数は 10 クラスであり，最後の第 5 版では 52 クラスであった．8 か月の全体の開発期間中に行われた開発内容は次のとおりである．

第 0 版 実現可能性評価用プロトタイプ開発

第 0 版はプロトタイプとしてシミュレーションシステム構築に対する VisualSmalltalk の実現可能性を検討するために開発された．このプロトタイプは使い捨て型のプロトタイプであり，以降のシステムとの関連は少ない．

第 1 版 基本構成システム構築

シミュレーションを行うために必要な最小限の機能と設備に該当するオブジェクトが開発された．

第 2 版 熱交換アルゴリズムの複雑化

熱交換のためにいくつかのアルゴリズムが提供され，シミュレーション設備の種類が増やされた．

第 3 版 シミュレーション設備の多様化とシミュレーション方式の変更

さらに設備の種類が増加し，熱交換のための冷却媒体の流れを制御する機構が追加された．同時に，シミュレーション設備を構成するエディタの操作性向上が要求された．

第 4 版 シミュレーション設備の追加と整理

新しい設備が追加されると同時に不要な設備を削除するように要求変更があった．

第 5 版 操作性改善

操作上の不具合の改善が要求された．

オブジェクトの進化過程を観測するにあたり，第 1 版から第 4 版までを観測対象として選択した．第 0 版と第 5 版を観測対象から外したのは，第 0 版が使い捨て型のプロトタイプであったためであり，第 5 版は第 4 版と計測結果に違いが認められなかったためである．

SystemA に各メトリクスを適用して得られた計測結果を，表 1.1および図 1.1 に示す．図 1.1の横軸はシステムが利用者に公開された版を表し，縦軸は第 1 版を 1 としたときの各版におけるそれぞれの計測値の相対度数を表している．SystemA の 4 つの版のうち，第 4 版のシステムに定義さたクラス数，メソッド数，総行数はそれぞれ 52 クラス，927 メソッド，8677 行で，第 1 版の 3.25 倍，4.80 倍，5.56 倍となった．図 1.1は，SystemA が S 字の曲線を描きながら規模を増加させていたことを示している．第 2 版から第 3 版へ至る期間が S 字の顕著な増加傾向を示す期間である．

SystemA では，第 1 版から第 2 版へ至る過程で，シミュレーションの設備を表示する機能を持っていたオブジェクトからシミュレーションの計算を行うオブジェクトを独立させることによって，それまで 2 つの主なクラス継承木によって実現されていたシステムのフレームワークは，3 つのクラス継承木が構成するフレームワークに変更された．NCT（クラス継承木の数）の増加は，このような設計変更を表している．

表 1.1: SystemA の成長

| メトリクス | 第 1 版 | 第 2 版 | 第 3 版 | 第 4 版 |
|---|---|---|---|---|
| NCD | 16 | 26 | 47 | 52 |
| 相対度数 | 1.00 | 1.63 | 2.94 | 3.25 |
| SLOC | 1560 | 3714 | 8044 | 8677 |
| 相対度数 | 1.00 | 2.38 | 5.16 | 5.56 |
| SNOM | 193 | 436 | 885 | 927 |
| 相対度数 | 1.00 | 2.26 | 4.59 | 4.80 |
| NMN | 115 | 241 | 452 | 463 |
| 相対度数 | 1.00 | 2.10 | 3.93 | 4.03 |
| NCT | 5 | 6 | 6 | 6 |
| 相対度数 | 1.00 | 1.20 | 1.20 | 1.20 |

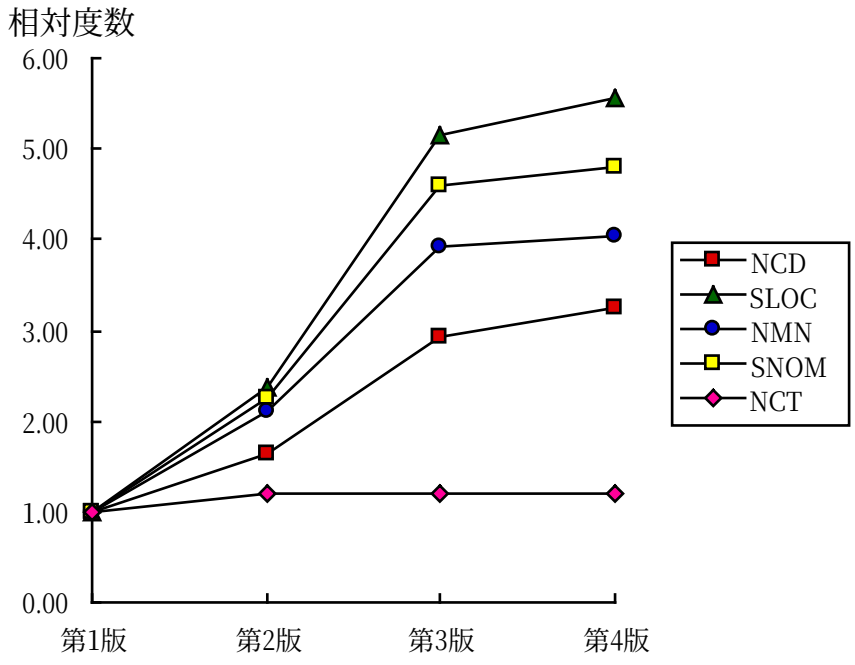


図 1.1: SystemA の定量的開発傾向の推移

### 1.5.2 SystemB

システム B の成長経過を表 1.2および図 1.2に示した．図 1.2は，SystemA と同様，第 1 版におけるそれぞれのメトリクスで計測した結果を 1 とし，以降の版の計測値の倍率として相対度数を求めて縦軸に表したグラフである．SystemB の 4 つの版のうち，第 4 版のシステムに定義されていたクラス数，メソッド数，総行数はそれぞれ 62 クラス，2644 メソッド，20470 行で，第 1 版の 1.68 倍，3.07 倍，3.51 倍となった．図 1.2が SystemA のような S 字の成長曲線を描いていないのは，SystemB と SystemA のシステムの性格の違いに由来するものである．SystemA は，シミュレーションシステムであり，システムの将来像を開発者が予想できない状況で漸進的に開発が進んだが，SysmteB の場合はビジネス系の定形処理であったため，開発の初期にシステムの将来像を開発者が想定できた．開発者へのインタビューから，第 1 版でフレームワークの妥当性を検証できたため，システム全体の構造は第 4 版へ至るまでに変更する必要なかったことも確認した．そのような状況下でも，開発期間中クラスの再構成は行われていた．これは，クラス数（NCD）が第 2 版以降減少している点から読みとることができる．システムの将来像が想定できいた場合でもクラスの再構成が起きていたということは，開発当初の設計が，利用者要求の変更や追加によって変更されたことを意味しており，利用者環境の変化がクラスを進化させる原動力となっていたことを示すものと解釈できる．

SystemB のもう一つの特徴として，利用者インタフェースを実現するクラスが 2 クラスだけであった点を挙げることができる．必要な利用者インタフェースは，VisualSmalltalk の開発環境として提供されている PARTS Workbench と呼ばれるコンポーネントを用いて開発された．

次に，全体の開発過程を説明する．

第 1 版 システムのフレームワーク評価用プロトタイプ構築

この版はフレームワークの評価を兼ねて開発された進化型プロトタイプであり，データベースと接続する前に顧客へリリースされた．

第 2 版 フレームワークに基づいた試用版システム構築

システムの試用向け機能が第 1 版のフレームワークを基に構築され，利

表 1.2: SystemB の成長

| メトリクス | 第 1 版 | 第 2 版 | 第 3 版 | 第 4 版 |
|---|---|---|---|---|
| NCD | 37 | 81 | 71 | 62 |
| 相対度数 | 1.00 | 2.19 | 1.92 | 1.68 |
| SLOC | 5839 | 18677 | 20579 | 20470 |
| 相対度数 | 1.00 | 3.20 | 3.52 | 3.51 |
| SNOM | 861 | 2491 | 2705 | 2644 |
| 相対度数 | 1.00 | 2.89 | 3.14 | 3.07 |
| NMN | 642 | 1762 | 1944 | 1928 |
| 相対度数 | 1.00 | 2.74 | 3.03 | 3.00 |

用者インタフェースの見直しも同時に行われた.

第 3 版 システム運用版の開発

利用者インタフェースの追加要求を受け取り，システムの運用版の仕様が決定され開発され，データベースと接続した版が利用者へリリースされた．ここで要求された利用者要求変更件数は 59 件で，そのうち 70%が利用者インタフェースの変更要求であった．他の 30%は機能の追加，変更，削除である．主な機能追加は，画面に表示された一覧表の操作に関するものである.

第 4 版 システムの使い勝手の改善

利用者要求変更は 6 件で利用者インタフェースと機能変更要求とが 50%づつであった.

### 1.5.3 SystemC

SystemC で観測された落水型の開発過程を図 1.3，その値を表 1.3に示す．図 1.3は第 00 版の各メトリクスを適用して求めた計測値を 1 としたとき，それぞれの版で求めた値を第 00 版に対する倍率を相対度数として縦軸にとり，システムの成長グラフを描いた．全体の成長が約 1 か月でほぼ収束しているのは，その後の開発で要求変更や仕様の変更が起こらなかったことを表している.

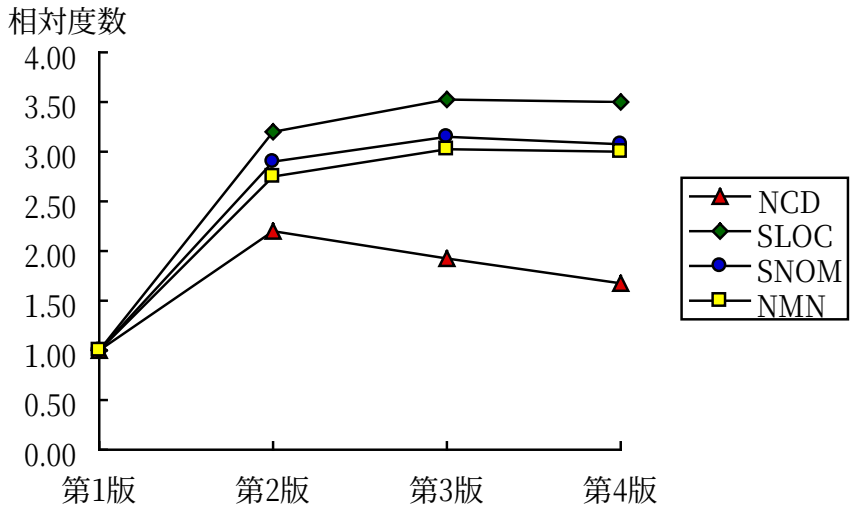


図 1.2: SystemB の定量的開発傾向の推移

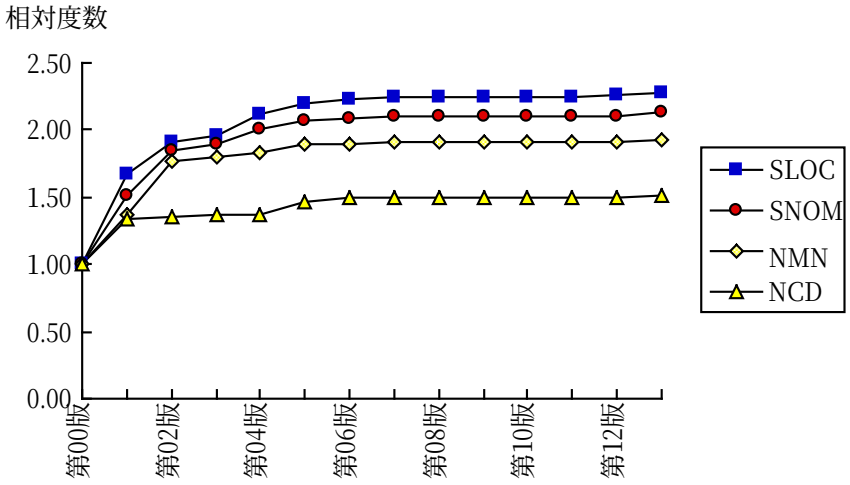


図 1.3: SystemC の定量的開発傾向の推移

表 1.3: SystemC の成長

| メトリクス | 第 00 版 | 第 01 版 | 第 02 版 | 第 03 版 | 第 04 版 | 第 05 版 | 第 06 版 |
|---|---|---|---|---|---|---|---|
| NCD | 88 | 118 | 119 | 120 | 120 | 129 | 132 |
| 相対度数 | 1.00 | 1.34 | 1.35 | 1.36 | 1.36 | 1.47 | 1.50 |
| SLOC | 6570 | 10966 | 12578 | 12846 | 13864 | 14405 | 14624 |
| 相対度数 | 1.00 | 1.67 | 1.91 | 1.96 | 2.11 | 2.19 | 2.23 |
| SNOM | 698 | 1060 | 1294 | 1319 | 1406 | 1449 | 1457 |
| 相対度数 | 1.00 | 1.52 | 1.85 | 1.89 | 2.01 | 2.08 | 2.09 |
| NMN | 510 | 695 | 904 | 914 | 933 | 964 | 968 |
| 相対度数 | 1.00 | 1.36 | 1.77 | 1.79 | 1.83 | 1.89 | 1.90 |
| メトリクス | 第 07 版 | 第 08 版 | 第 09 版 | 第 10 版 | 第 11 版 | 第 12 版 | 第 13 版 |
| NCD | 132 | 132 | 132 | 132 | 132 | 132 | 133 |
| 相対度数 | 1.50 | 1.50 | 1.50 | 1.50 | 1.50 | 1.50 | 1.51 |
| SLOC | 14738 | 14731 | 14735 | 14736 | 14737 | 14814 | 14934 |
| 相対度数 | 2.24 | 2.24 | 2.24 | 2.24 | 2.24 | 2.25 | 2.27 |
| SNOM | 1465 | 1466 | 1466 | 1466 | 1466 | 1466 | 1487 |
| 相対度数 | 2.10 | 2.10 | 2.10 | 2.10 | 2.10 | 2.10 | 2.13 |
| NMN | 976 | 976 | 975 | 975 | 976 | 976 | 981 |
| 相対度数 | 1.91 | 1.91 | 1.91 | 1.91 | 1.91 | 1.91 | 1.92 |

第 13 版のシステムの総クラス数， メソッド数，行数はそれぞれ 133 クラス，1487 メソッド，14934 行で，第 00 版から 1.51 倍，2.13 倍， 2.27 倍となっていた．利用者インタフェースは，SystemB と同様，PARTS Workbench を用いて開発されており，5 クラスだけがインタフェースに使われる部品として定義されていた．