# 現代的再利用で気をつけるべきことは何か

門井　仁†

**概要**：古典的再利用の時代と比較して，現代的再利用の時代はプログラムの再利用が容易になっていると考えられる．このことは短期のアプリケーション開発を可能にすることに繋がるが，その中でデータの取り回しの設計が以前に増して重要になるのではないだろうか．

# What is important when reusing software in the modern period ?

HITOSHI KADOI†

***Abstract***: Compared with the era of classical reuse, it is considered that the reusing software is easier in the age of modern reuse. This leads to short-term application development, but in the meantime, it seems that the design of data handling will be more important than before.

## 1. はじめに

古典的再利用の時代は，サーバーやソフトウェアなどの情報システムを自前で賄う，100%オンプレミスの時代であった．開発期間は比較的長く，また開発されたアプリケーションの寿命も 10 年以上と長い．（図 1）

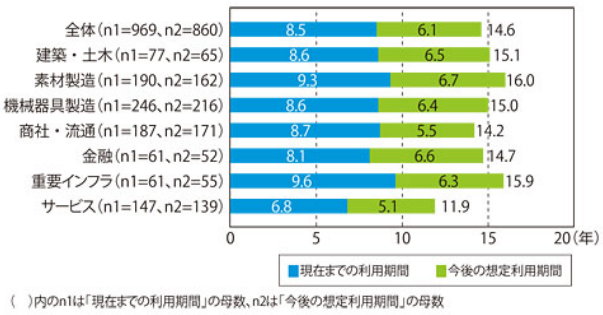


図 1　代表的な基幹システムの寿命（2012 年調査）[1]

一方で，メインフレーム中心の時代からダウンサイジング，そして Web 中心へと，システムの基盤に大きな変更が加わっていった．

この状況の中，再利用技術は研究，実装されていったが，再利用可能な資産を作成する側と利用する側の双方に難しさがあった．

作成側の難しさとは，将来的に幅広く利用できるように作成しなければならないことであり，利用側の難しさとは，仕様を正確に理解した上で，アプリケーションに適用しなければならないことである．長期のシステム寿命を求められる場合，一般的に遠い将来の予測は難しいことから，特に難しくなる．

これを克服するのは，SIer および発注側のレベル次第であったが，再利用を考えずに新たにコピー作成する方が安易でもあったため，再利用は思うように進まなかったと考えられる．

このことを踏まえた上で，現代的再利用について考えてみる．

## 2. 現代的再利用で良くなったのか

まず環境や，再利用可能な資産の作成側がどのように変化してきたかを確認し，その後で利用側について論じたい．

### 2.1 ハードウエアの変化

ハードウエアの低価格化により，メモリやディスク量を以前より多量に使用することができるようになり，また CPU 能力の高いものが安価に手に入るようになった．このことは，ソフトウエアの能力的問題をハードウエアがカバーできるということになった．加えて，クラウドの普及により，ハードウエアを保持する必要がなくなり，必要に応じて手軽に能力増強や削減を実施できるようになった．このことは，計画的にハードウエアの能力を見積もって調達しなくても良くなったことを意味する．結果，アプリケーションのパフォーマンスに対する要求レベルが緩められる等，開発ひいては再利用に対する制約が緩和されたと考えられる．

### 2.2 OSS(Open Source Software)の普及

OSS が世間的に認知され，既存メーカーもその潮流に乗ってきた．OSS の開発については，「伽藍とバザール」[2] で述べられているように，公開することにより品質の高いソフトウエアを生み出すことに成功し，多様な種類の OSS が世の中に溢れる結果となった．また，関連情報は internet を通じて簡単に手に入るようになり，利用者はもはやメーカーの既製品を利用することなくアプリケーションを開発することができるようになった．

† 門井　仁,Advisory Architect
Engineering and Cognitive Innovation
Global Business Services, IBM Japan, Ltd
email: hkadoi@jp.ibm.com

©2019 Information Processing Society of Japan

また，開発言語も多様化する中で，Python における機械学習系のライブラリのように，利用価値の高いライブラリを組み込み，それにより言語の価値を高めるといった動きになっている．

### 2.3 クラウドネイティブ

AWS に代表されるクラウドベンダーは，クラウド上に豊富なサービスを投入することで，競争力を付ける戦略を採っている．そして，クラウドネイティブ[3]という言葉が出来たように，クラウドを前提としたアプリケーション開発ができる状態になってきた．

### 2.4 利用側はどうなのか

これまで見てきたように，利用側にとって使えるものが多くなり，ソフトウエア品質も上がっているため，古典的再利用の時代と比較して再利用しやすくなってきたと言える．

一方で，古典的再利用の時代に存在した，システム寿命の長さに起因する利用側の難しさについては，大きくは改善していないのではないかと思われる．

しかしながら，調査[5]によると，アプリケーション開発に対する課題の上位に，開発期間の削減と迅速・柔軟な開発が挙げられている（図 2）．このことから，長期のシステム寿命は必ずしも求められていないと推測できる．

このように利用側にとって，再利用に対する敷居は低くなってきているが，The twelve-factor app [4]にまとめられているような，従来のオンプレミスとは違ったクラウドならではのベストプラクティス等，利用に対する新たな考慮事項が出てきている．

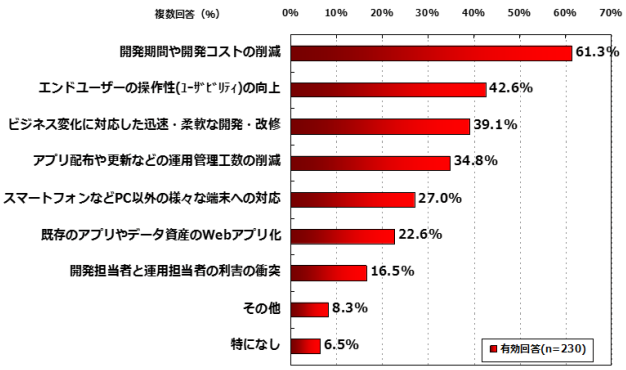


図 2　業務アプリケーション開発の課題[5]

## 3. 議論

2 章の最後に触れた，長期のシステム寿命が必須ではないことを前提とした場合，逆に言えばアプリケーションを短期的に更改していくことが重要ということになる．ということは，再利用できるだけでなく，如何に開発サイクルを早めることができるかが重要となる．

このように，アプリケーションは短期的に変化していくが，逆に変化が少ないものの代表例はデータであると考える．古典的再利用の時代は，アプリケーションの寿命が長かったため，アプリケーションを刷新するタイミングで，データ移行などの扱いを考えれば良かった．しかし，現代的再利用においては，システム更改時に従来のデータをそのまま使う，若しくは必要最小限の変更に止めた状態で使うようにする，といったデータの取り回しやすい設計にすることの重要性が増したのではないだろうか．

また，データをクラウド上に置く場合，ベンダーロックインを回避するため，どの程度クラウド側に置き，どの程度ベンダー特有のサービスを利用するか検討すべきではないかと考える．

## 参考文献

[1] 日経コンピュータ
https://tech.nikkeibp.co.jp/it/article/COLUMN/20120511/396127/
[2] エリック・レイモンド　伽藍とバザール
http://www.catb.org/~esr/writings/cathedral-bazaar/
[3] クラウドネイティブ　https://www.cncf.io/
[4] The twelve-factor app　https://12factor.net/
[5] @IT 業務アプリ Insider 読者調査レポート
http://www.atmarkit.co.jp/ait/articles/1604/05/news028.html

©2019 Information Processing Society of Japan