JaSST ‘13 Tokyo

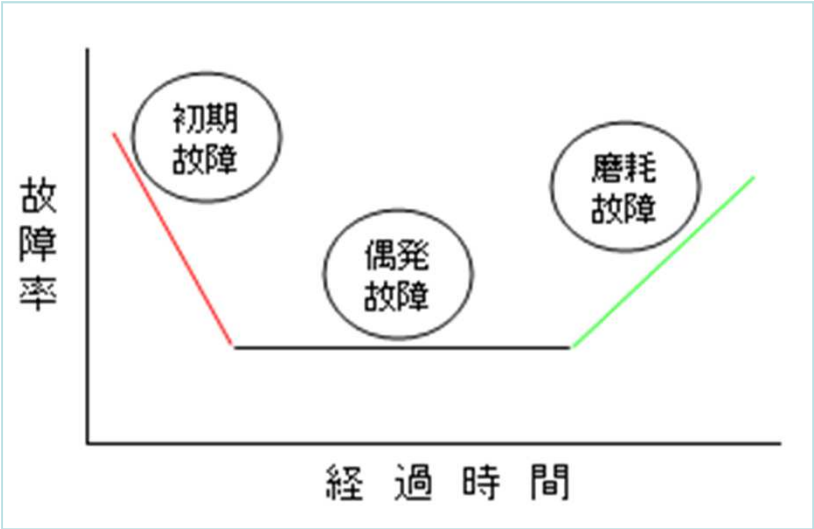
データベースメトリックスの活用
ーシステム内部品質の向上に向けてー

日本アイ･ビー･エム㈱
蔭山泰之

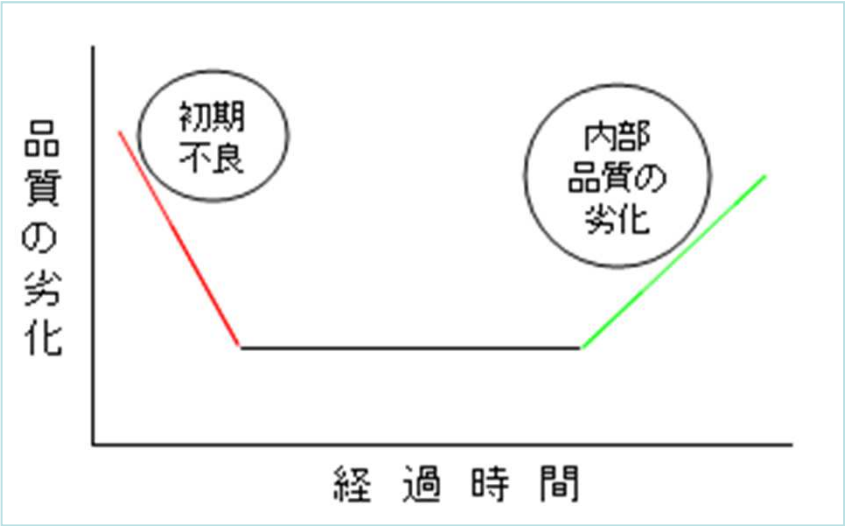
ソフトウェアのバスタブ曲線

どのようなシステムでも、長年使い続けていれば、その品質は経過時間にともなって劣化していく。

この点では、ソフトウェアの品質の推移は、ハードウェアが示すバスタブ曲線と似たようなかたちになる。



ハードウェアのバスタブ曲線

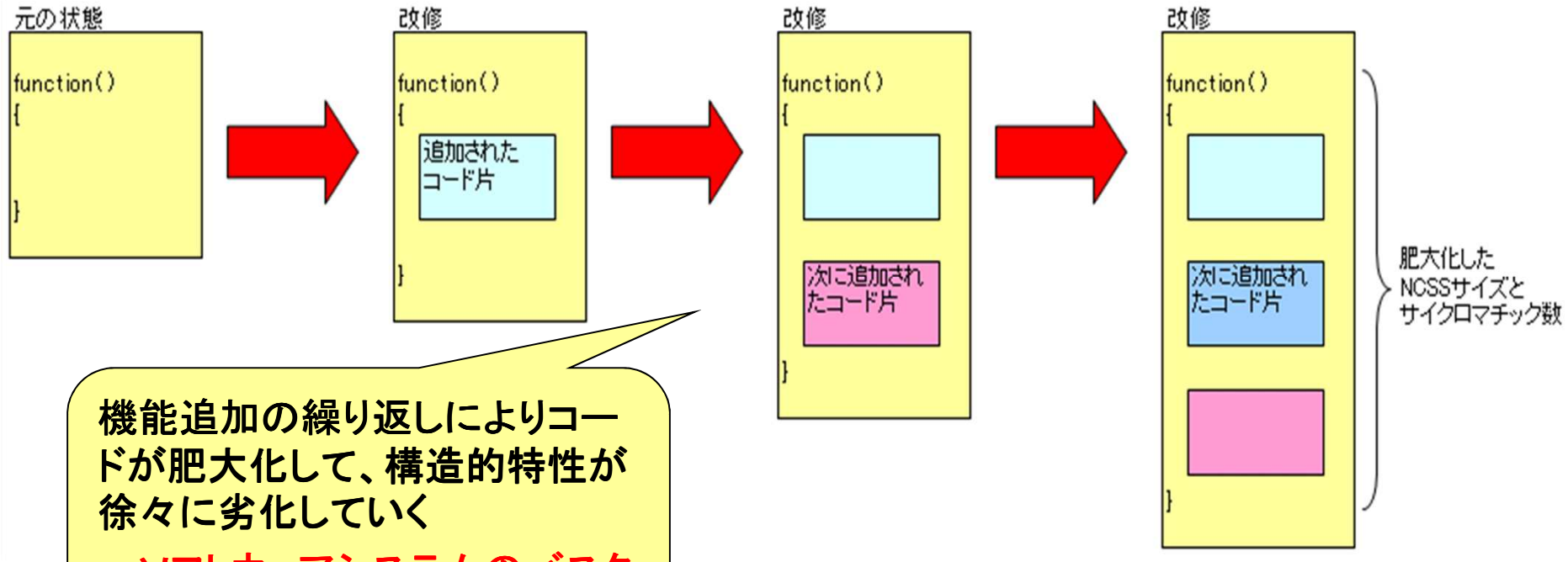


ソフトウェアのバスタブ曲線

ただし、ソフトウェア･システムの場合、ハードウェアの磨耗故障にあたるのは内部品質の劣化。

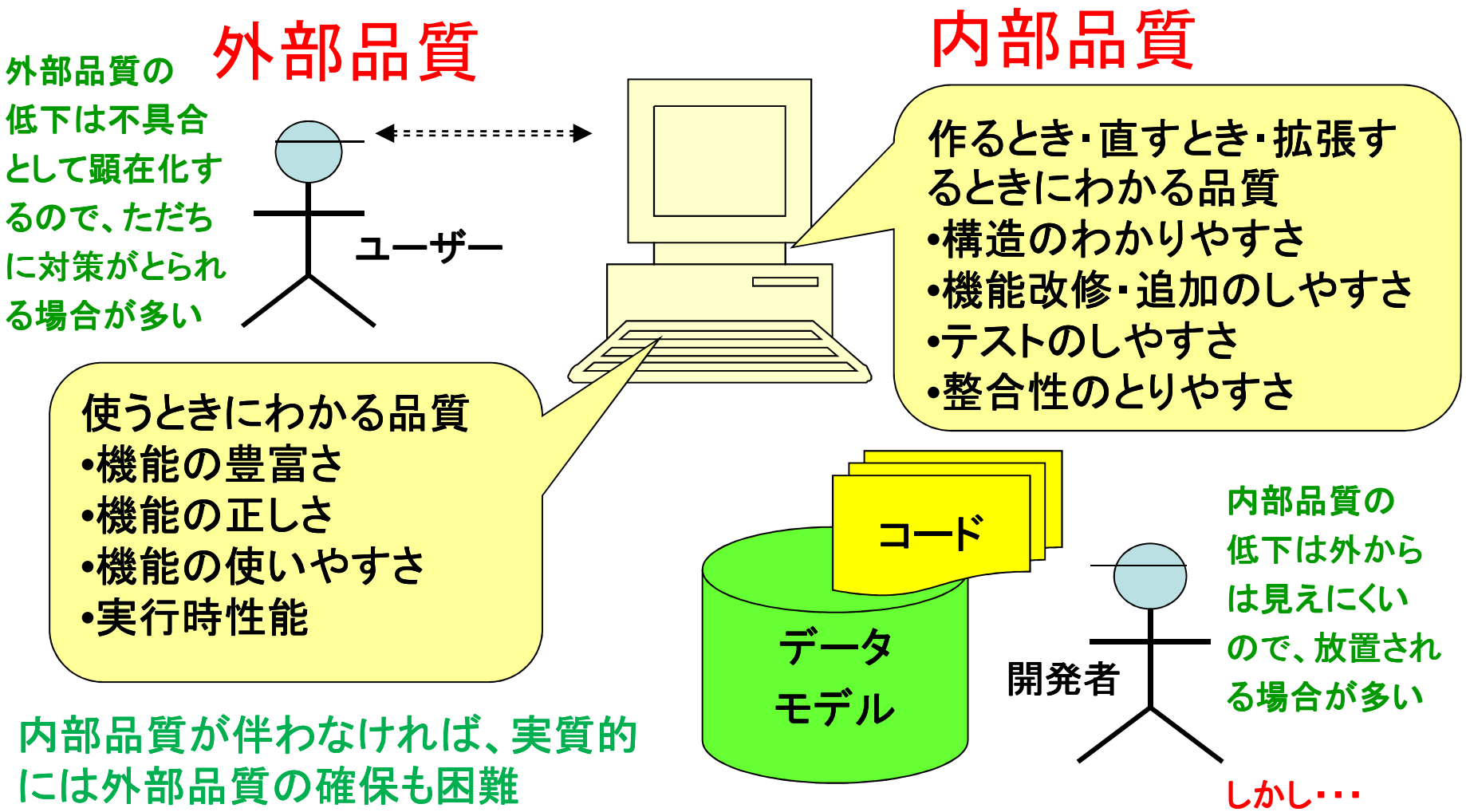
改修の積み重ねによるコードの肥大化

最初は保守しやすいコードだったとしても、保守による改修の際に徐々にコードの品質が低下していく可能性がある。とくに、機能的に重要なコードは改修頻度が高いので、重要な部分ほどコードの品質が低下するという傾向がある。

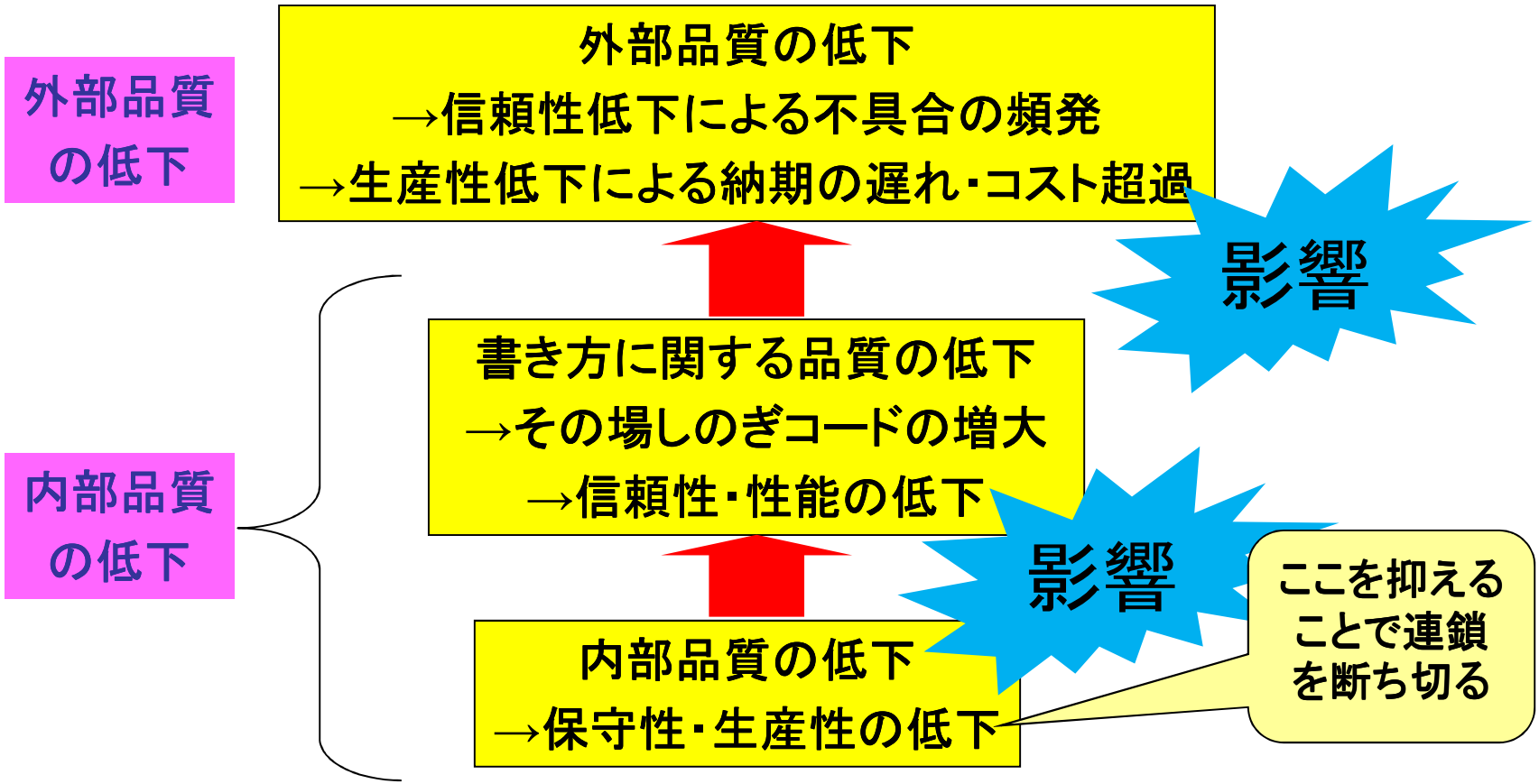


外部品質と内部品質

ソフトウェア・システムの品質は大きく外部品質と内部品質に分けられるが、中長期的な対策を施すべきは、内部品質の方。



質低下の連鎖

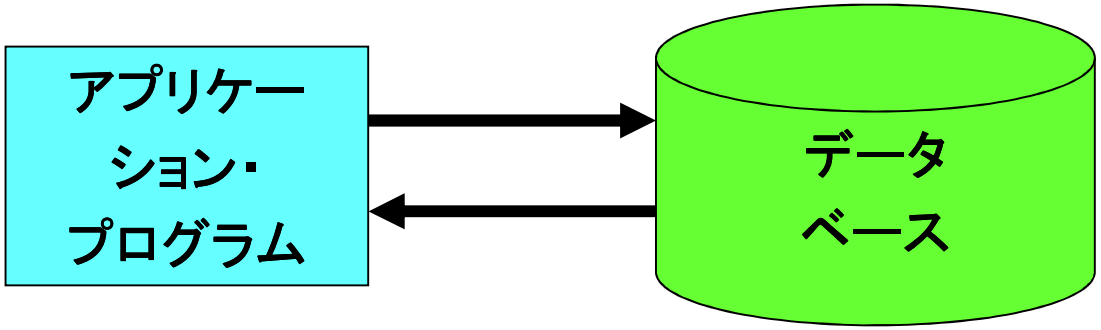


内的品質の構造に関する品質が低下すると、保守性・生産性が低下し、改修の影響を十分に調べ切れないうちに改修を加えたり、影響範囲が不明なままテストを実施したりするようになる。すると、内部品質低下が、結果として外的品質の低下として顕在化してくる。

データモデルの品質維持・管理

プログラムとデータはシステムの両輪であり、システムの品質を維持するためには、プログラムの品質だけでなく、データの品質も管理しなければならない。

とくに、データベースは、複数のプログラムが共通にアクセスする情報なので、その品質の低下の影響は小さくない。



データの品質は、

（1）データの妥当性・精度などの内容に関する特性

（2）データの整合性・効率などの構造（データモデル）に関する特性

に分けることができる。構造に関する品質、とくにアプリケーションとの関係についての情報は、CRUDマトリックスなどによって管理することができる。

データベースの不吉な匂い

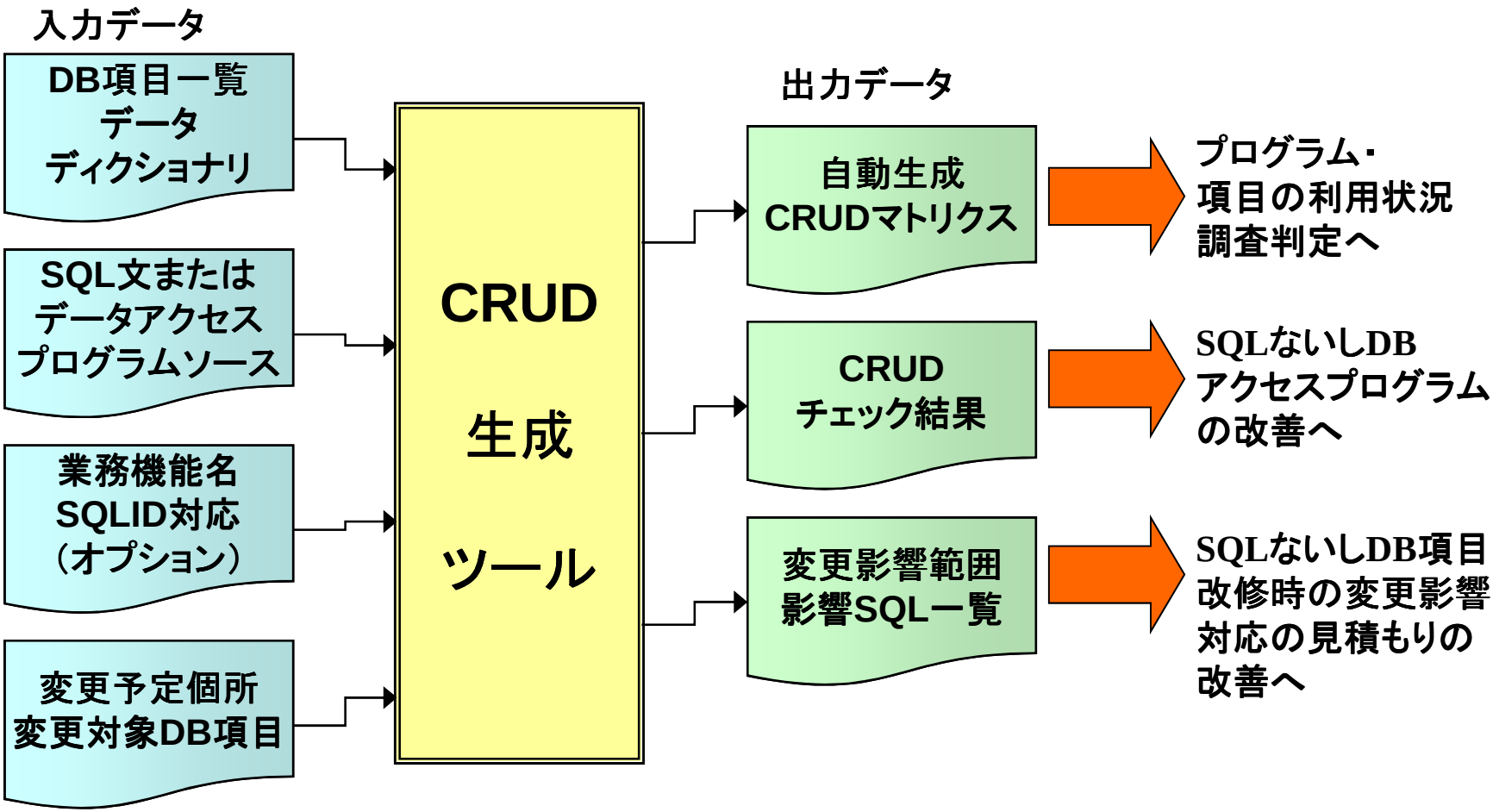
データベースの内部品質が良くない場合の比喩としての「データベースの不吉な匂い」。リファクタリングのきっかけ。

- 複数の目的に使われるカラム。
- 複数の目的に使われるテーブル。
- 冗長なデータ。
- カラムの多すぎるテーブル。
- 行の多すぎるテーブル。
- 「スマート」カラム。
- 変更の恐怖。

S. W. アンブラー、P. サダラージ
『データベース・リファクタリング』より

CRUDマトリックス生成ツール

データディクショナリ（テーブルID、項目ID）とSQL文の情報を元に、
DB項目単位、SQL文単位にCRUDマトリックスを生成するツール



データベースメトリックスの比較対照

規模がかなり大きい

	今回のシステム	自動車メーカーA 設計部品表システム その1	自動車メーカーA 設計部品表システム その2	自動車メーカーA 設計部品表システム その3	自動車部品メーカーB 設計部品表システム その1	自動車部品メーカーB 設計部品表システム その2	自動車メーカーC 生産ライン管理システム	自動車メーカーD 設計部品表システム	精密機器メーカーE 特許情報管理システム	情報通信会社F 料金管理システム
全テーブル数	580	124	194	242	124	156	165	193	321	72
全項目数	15,791	1,619	2,648	4,455	1,996	3,010	1,406	2,928	5,350	2,027
全実質項目数	14,241	875	1,484	3,003	1,252	1,918	1,406	2,757	4,387	1,591
全主キー項目数	1,101	404	736	839	352	443	370	1,111	867	253
全実質従属項目数	13,140	471	748	2,164	900	1,475	1,036	1,646	3,520	1,338
主キー項目比率(対全実質項目数)	7.7%	46.2%	49.6%	27.9%	28.1%	23.1%	26.3%	40.3%	19.8%	15.9%
実質従属項目比率(対全実質項目数)	92.3%	53.8%	50.4%	72.1%	71.9%	76.9%	73.7%	59.7%	80.2%	84.1%
1テーブルあたり平均項目数	27.2	13.1	13.6	18.4	16.1	19.3	8.5	15.2	16.7	28.2
1テーブルあたり平均実質項目数	24.6	7.1	7.6	12.4	10.1	12.3	8.5	14.3	13.7	22.1
1テーブルあたり主キー項目数	1.9	3.3	3.8	3.5	2.8	2.8	2.2	5.8	2.7	3.5
1テーブルあたり実質従属項目数	22.7	3.8	3.9	8.9	7.3	9.5	6.3	8.5	11.0	18.6
アクセスSQL本数	10,188		3,140	1,063		1,755		3,265		457
CRUDマーク数	145,828		27,042	16,865		14,702		38,977		5,068
マトリックス上CRUDマーク発生率	0.091%		0.325%	0.356%		0.278%		0.408%		0.547%

1

2

3

- 凡例

実質項目：テーブル中の項目のうち、更新日時、更新者IDなどのシステム管理上設定されている項目を除いた、アプリケーショントランザクションで利用される項目。

他のシステムと比較した今回のシステムの特徴

1. 主キー項目の比率が低い

正規化があまり進められていないためか？

エンティティ・リレーションにおいてリレーションが深くない？

2. ひとつのテーブルあたりの項目数が多い

テーブルの凝集度が低い。正規化があまり進められていないか？

項目の繰り返し・重複がある？

主キーに従属しない項目がある？

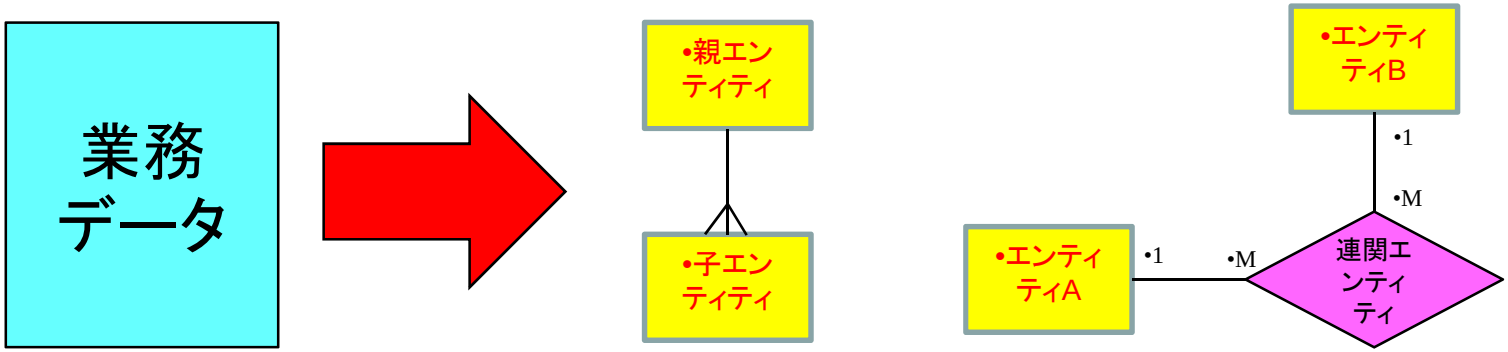
推移従属がある？

3. CRUDマトリックスが疎である

ある特定の目的のために設定されていて、複数のアプリで共有されていないテーブルが多い？

同じ意味の項目が複数のテーブルに散らばっている？

正規化の進展



正規化が進められると、ひとつひとつのエンティティのサイズが小さくなり、その代わりにエンティティ同士の間にリレーションが増えてくる。

リレーションが増えると、それを表現するための主キー項目が増える。

1. 親エンティティの主キー項目が外部キーとして子エンティティに降りてくる。
2. 連関エンティティには、連関を結ぶエンティティの主キー項目が外部キーとして含まれてくる。

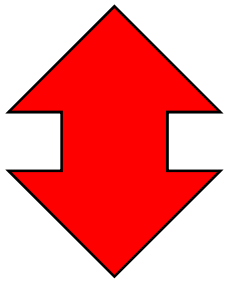
相対的に従属項目の比率は下がってくる。

正規化と項目数の多いテーブル

正規化を進めれば、主キー項目によって決まる項目だけが従属項目として残るので、ひとつひとつのエンティティは小さくなっていく。

つまり、そのエンティティに本当に必要な項目だけが残る。

結果としてエンティティの凝集度が高まる。



逆に、従属項目が多いエンティティは、そこに本来は必要のない項目が多く含まれている可能性がある。

必要のない項目は、複数のエンティティに重複して存在している可能性がある。

⇒ データ不整合の温床

CRUDマトリックスの密度

	アプリA	アプリB	アプリC	アプリD	アプリE	アプリF
テーブルA	■	■	■			■
テーブルB		■	■		■	
テーブルC			■			
テーブルD		■		■		■
テーブルE			■	■	■	
テーブルF	■	■	■			■

密なCRUDマトリックス

	アプリA	アプリB	アプリC	アプリD	アプリE	アプリF
テーブルA	■					
テーブルB		■				
テーブルC			■			
テーブルD				■		
テーブルE					■	
テーブルF						■

疎なCRUDマトリックス

- 複数のアプリケーションから共通にアクセスされているテーブルが多いと、CRUDマークは増える傾向にあり、CRUDとしては密になる。
- 各アプリケーションがそれぞれ固有のテーブルにアクセスしていると、CRUDマークは減る傾向にあり、CRUDとしては疎になる。
- 正規化が進められていないと、同じ項目でもアプリケーションごとの固有のテーブルに散らばってしまい、データの不整合が発生しやすくなる。

品質改善作業のポートフォリオ

データベースは複数のアプリから共通にアクセスされる基盤なので、リファクタリングの効果も大きいが、影響も大きい。

難易度			
	高	①	②
	低	③	④
		小	大
		効果	

改善効果の高い部分② 、④は、放置した場合に危険度の高い部分でもある。

⇒効果が大で難易度の低い④の部分から始めるのがよい。

④から始めて③に至り、最終的には②を目指す。

内部品質向上へ向けての対応策

- **未使用テーブルの確認・除去**

  物理的に定義されているが、現状ではアプリケーションから使われていないテーブルの有無をチェックし、あれば、それの必要性を検討して、不必要であれば削除する。

⇒不要なテーブル数が減ることにより、調査範囲が減り、影響範囲調査などの人的工数が低減する。

- **重複SQLの除去**

  同じCRUDマークのSQLを調査し、同じ内容のSQLであれば重複を除去する。

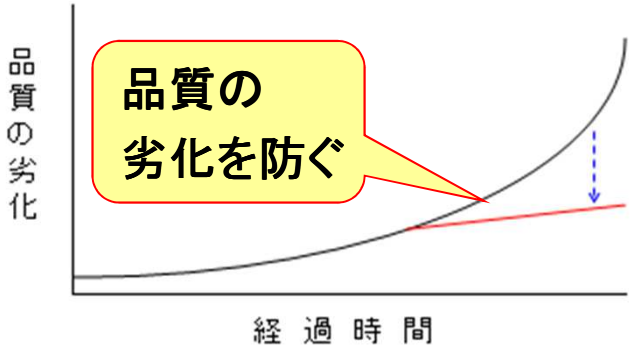
⇒SQLの共通利用により、SQLの全体本数が低減し、調査範囲を絞ることができて、調査工数を削減することができる。
SQLに対して改修・改善を施さなければならなくなった場合に、同じ複数のSQLを改修する必要がなくなるので、改修・テスト工数を削減することができる。

		SQL1	SQL2	SQL3	SQL4
TABLE1	FIELD1	R			R
TABLE1	FIELD2	R			R
TABLE1	FIELD3				
TABLE2	FIELD1		C	D	
TABLE2	FIELD2		C	D	
TABLE2	FIELD3	R	C	D	R
TABLE3	FIELD1				
TABLE3	FIELD2				
TABLE3	FIELD3	R			R

同じCRUDマーパターン

システム内部品質の向上に向けて

内部品質維持・向上のためにかける工数は、中長期的には外部品質維持の工数削減、あるいは機能拡張の生産性向上というかたちで効果が現われてくる。



放置しておけば時間とともに劣化していく内部品質について対策を施すことで、外部品質の低下を防ぐことにつながる。

品質維持・向上のための施策

1. 機能拡張・新規追加部分の品質維持管理（ルールとチェック）
   品質維持のためのルールを設けてそれが守られているかチェックする。
2. システム既存部分の品質向上（再設計とリファクタリング）
   すでに劣化している部分で将来に影響がありそうな部分を改善する。
3. システムの内部品質の継続的な計測と評価
   どちらについても内部品質の状態の推移を客観的に評価する必要がある。