

ソフトウェアカンバン手法の概要

長瀬 嘉秀



カンバン

ソフトウェア開発の変革

Improving Service Delivery
in Technology Businesses

David J. Anderson (著)

長瀬 嘉秀・永田 渉 (監訳) / テクノロジックアート (訳)

「カンバン」の先駆者：デビッド J. アンダーソンの最高傑作！

世界で読まれ続けている名著
「KANBAN」の本邦初訳

ムダを無くし生産性を飛躍的に向上させる
トヨタの「かんぱん」方式を、ソフトウェア開発に適用。
注目の的となった手法の核心を解説。

リックテレコム

「カンバン(邦訳版)」の刊行に寄せて

トヨタの「かんばん」の本質は、モノの流れの滞留(異常)を検知することであり、またトヨタ生産方式の狙いは、異常(カイゼン・ニーズ)に対して前向きに取り組む組織文化・風土を醸成することにある。

本書では、今までに出版されているアジャイル関連の書籍以上に組織文化やマネジメント(経営)に触れると同時に、「かんばん」の考え方をういてより「儲かるIE(Industrial Engineering)」に近づいてきた取組みであると言える。マネジメントを行っていく上で大変重要でかつ根源的な西洋文化と改善文化(日本文化)という部分にも触れ、よりトヨタ生産方式の本質を理解した上での「カンバン」(本書)の取組みであることが伝わって来る。

本書の「カンバン」は、アジャイル関連の書籍には少なかったトヨタ生産方式の本質を意識した内容になっており、アジャイルの枠を超えた「儲かるIE」とソフトウェア工学の新たな融合を提唱する一冊である。

私が指導した IT 職場では、サービスを提供するまでに2ヶ月の期間を要していたが、カードレベルで見ると、現実には、20時間前後の時間しか消費しておらず、大半が、仕掛りの量と優先順位の問題により遅延を引き起こしていた。仕掛りというのは、トヨタ生産方式では、手待ちのムダとして扱われるが、この仕掛りを無くすために、一個流しで整流化するためには、本書のような「カンバン」のしくみが必要になってくる。

企画、開発、運用などのチームを跨がった場合には、チーム毎に複数の案件を抱えているため、仕事の優先順位がチーム毎に勝手に設定されている。したがって、経営者から見ても、品質の向上やリードタイムの短縮を行うためには、本書に書かれているような取組みは有効であり、組織文化を変えるための古くて新しい取組みであるといえる。既にアジャイル開発に取り組んでいるチームの次のステップとして、営業～企画～開発～運用までのリードタイムの短縮などに是非この「カンバン」(後工程引取り)方式を用いて実践し、研究されることを推奨する。

2014年6月

株式会社 豊田マネージメント研究所

副社長 エグゼクティブ・コンサルタント 高木 徹

万能な開発プロセスはない

「フリーサイズの」開発方法論がうまくいかない理由

チームは以下で異なる。

- ・スキルセット
- ・経験レベル
- ・能力レベル

プロジェクトは以下で異なる。

- ・予算
- ・スケジュール
- ・スコープ
- ・リスクプロファイル

組織は以下で異なる。

- ・バリューチェーン
- ・ターゲットマーケット

開発プロセスのトレンド

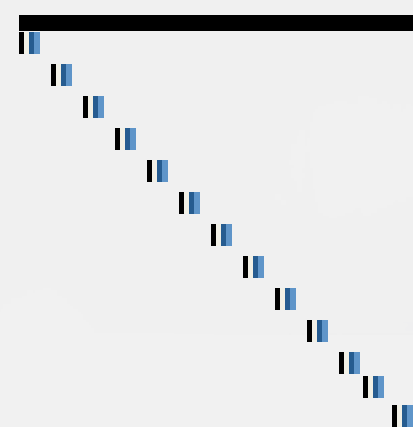
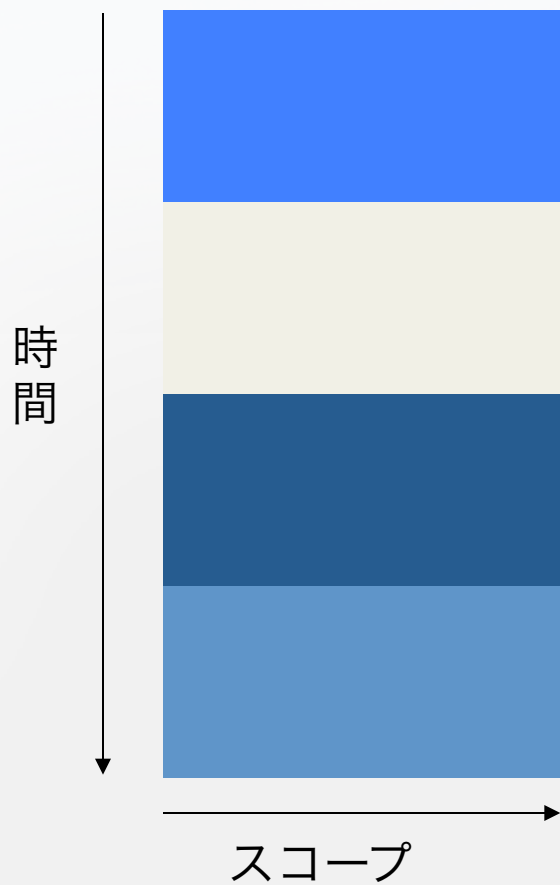
- ウォーターフォール
- スパイラル
- イテレーティブ
 - RUP(ラショナル)
- アジャイル
 - XP
 - スクラム
 - FDD
- リーン
- カンバン

アジャイルの特徴: 短期間の反復開発

ウォーターフォール

イテレーティブ

アジャイル



ObjectMentor資料より



ゴールドドラットの「制約条件の理論」

ボトルネックを発見し、それがパフォーマンスの制約とならなくなるまで、その解消方法を考えるというものである。ボトルネックが一つ解消すれば、新しいボトルネックが出てくるので、サイクルを繰り返す。ボトルネックを発見し除去することで、パフォーマンスを体系的に改善するという反復的な手法である。

- ドラム・バッファ・ロープ
- 開発の状況をフローとしてとらえトラッキングする
- プル型システム(後工程引き取り方式)
- 仕掛り(WIP)が事前に同意したある一定量に制限
- カンバンシステムへ

カンバンシステムへのアイデア

- ワークフローの見える化
- 作業項目型
- リズム(フローの独立、平準化)
- 優先度分類
- 具体的な管理レポート
- 改善のための検討

カンバンとは

- 皇居東御苑訪問
- 入園票で制御する
- 入場者を制限して、観光(処理)をできるようにする



カードウォール



出典：「カンバン ソフトウェア開発の変革」(リックテレコム)より

5つのプロパティ

- ワークフローの見える化
- 仕掛り(WIP)の制限
- フローの測定と管理
- プロセスポリシーの明示化
- 改善機会を認識するためのモデルの使用



出典：「カンバン ソフトウェア開発の変革」〈リックテレコム〉より

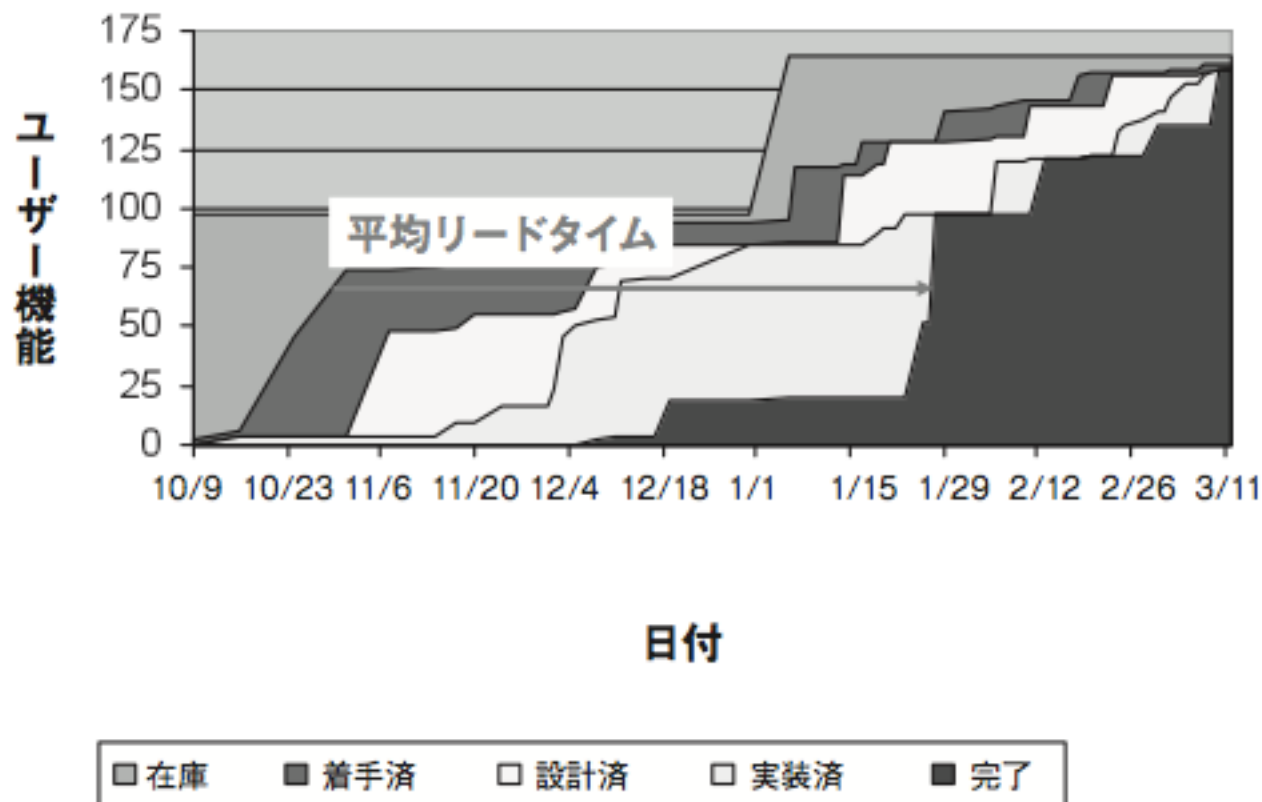
カンバンの手順

- 品質に集中する。
- 仕掛り(WIP)を減らす。
- デリバリーをひんばんに行う。
- 要望とスループットのバランスを取る。
- 優先順位を付ける。
- 予測可能性を向上させるために、ばらつきの原因を解消する。

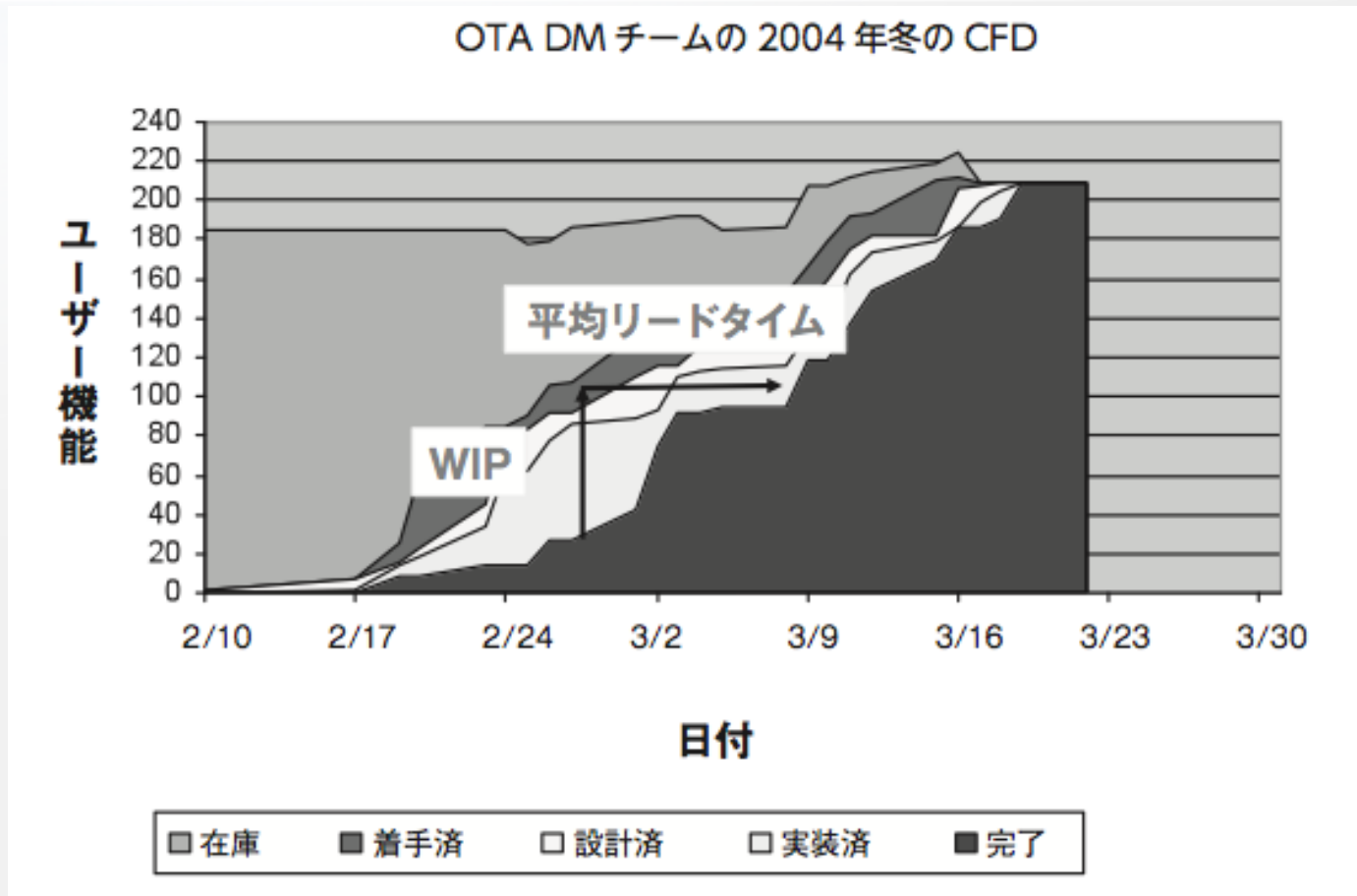
仕掛り(WIP)の減少とひんばんなデリバリー

- 累積フローダイアグラム

OTAダウンロードチームの2003年秋から2004年冬のCFD



- 累積フローダイアグラムの例2

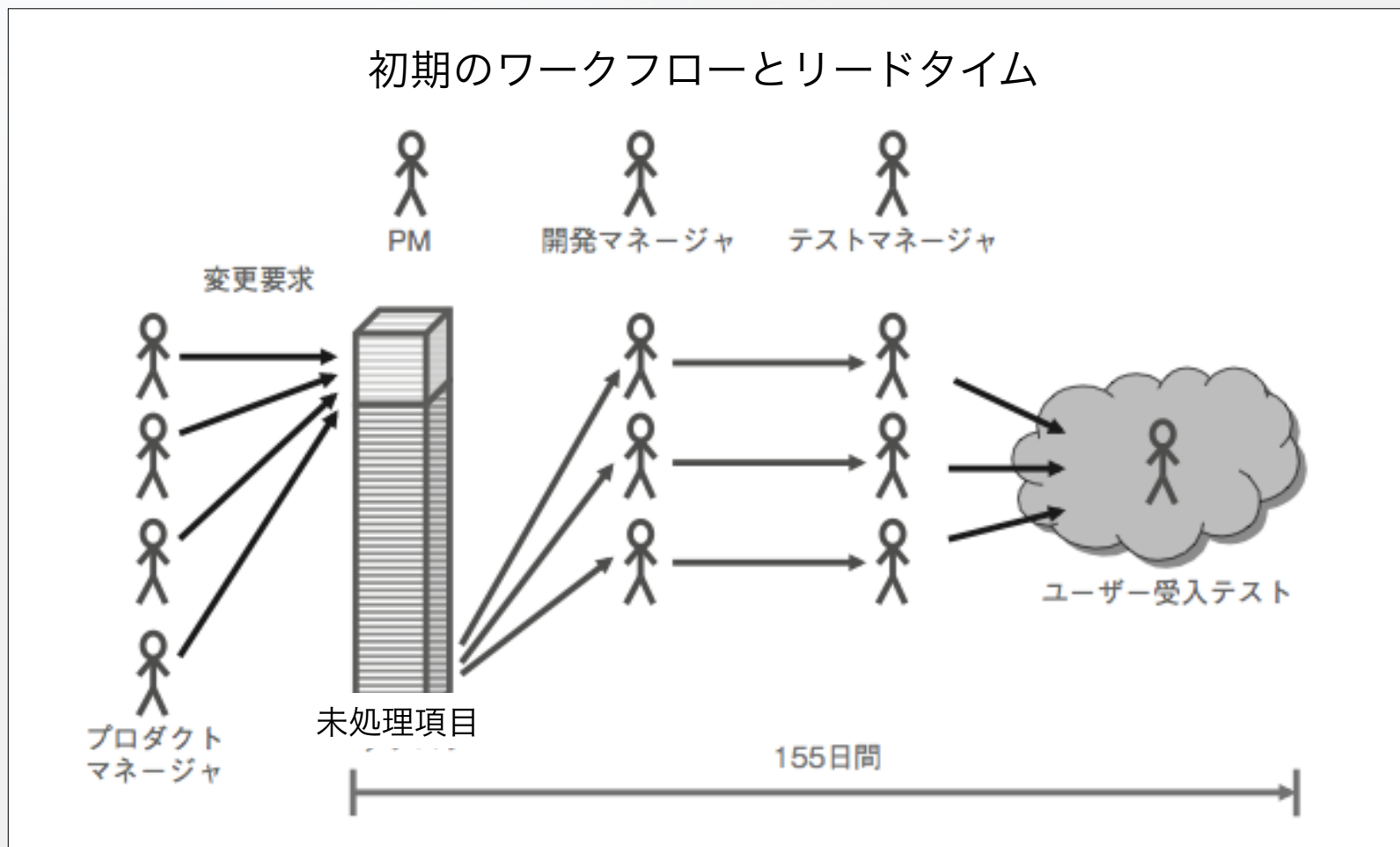


出典：「カンバン ソフトウェア開発の変革」〈リックテレコム〉より

仕掛りの量が減ると、平均リードタイムが短くなる

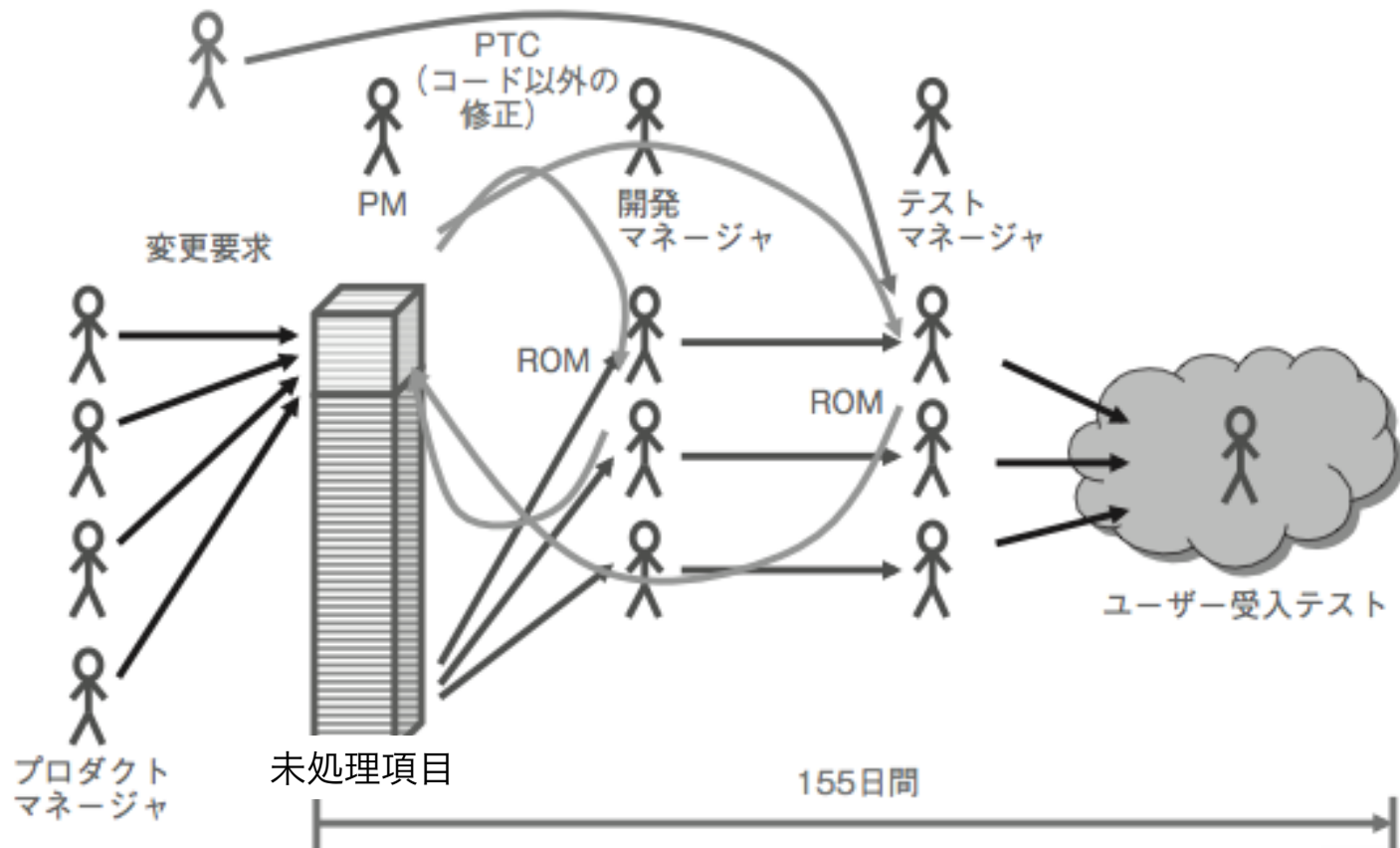
ワークフローとリードタイム

MS社での例

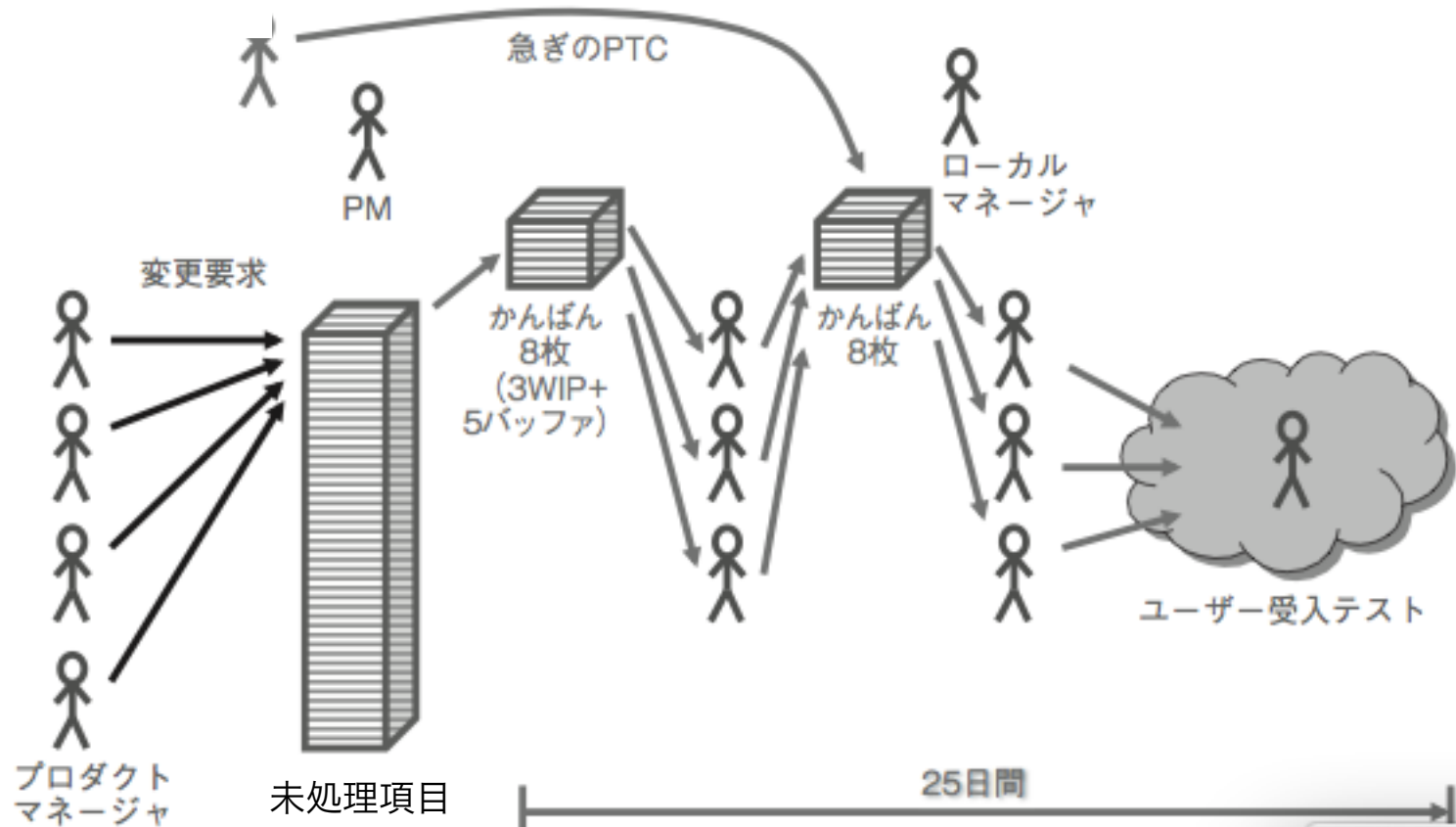


概算値見積り作業とPTC（割り込み作業）が邪魔をする

概算値見積りとPTC入力を含むワークフロー



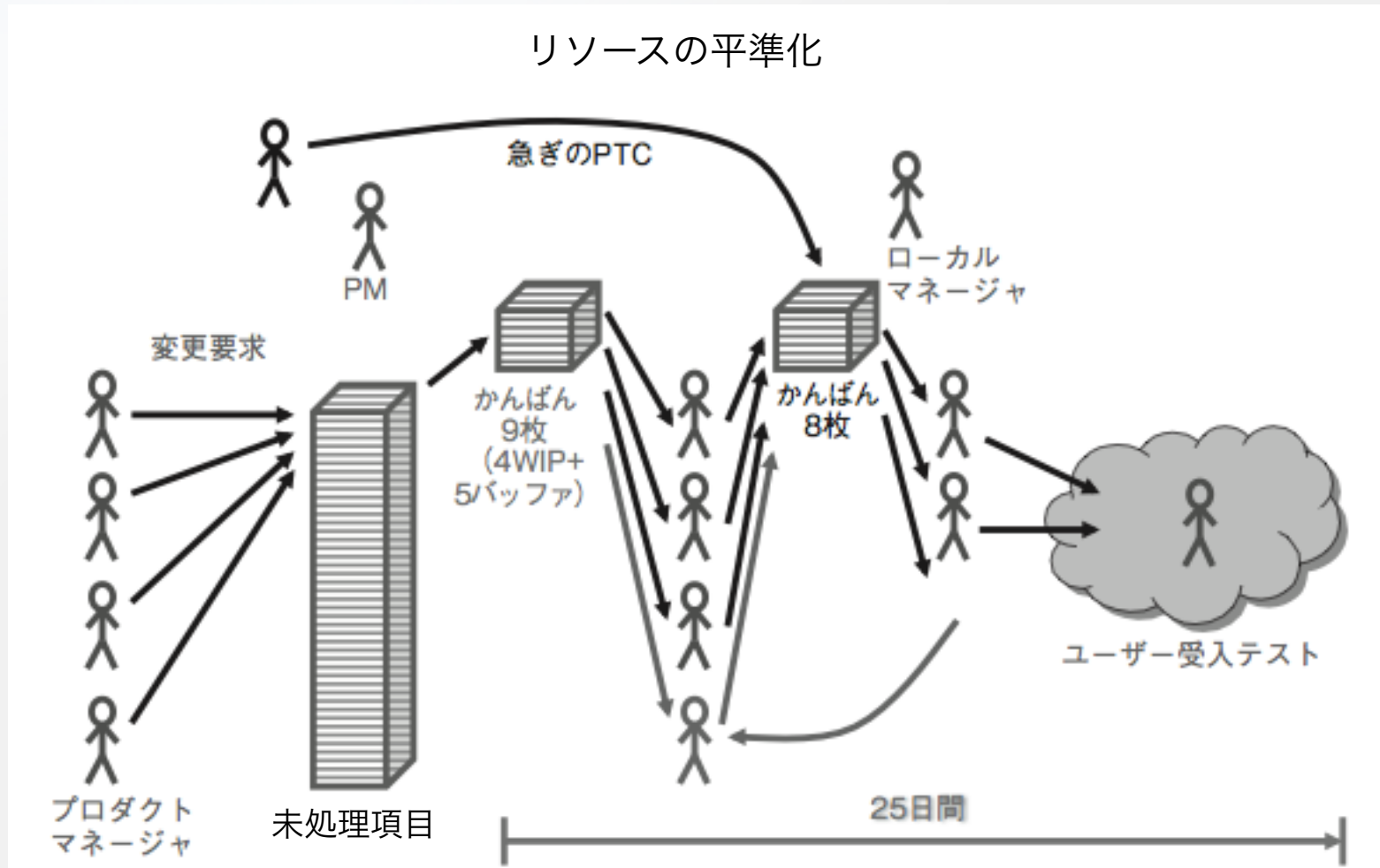
カンバンによる仕掛け (WIP) 制限とキューのあるワークフロー



出典：「カンバン ソフトウェア開発の変革」(リックテレコム)より

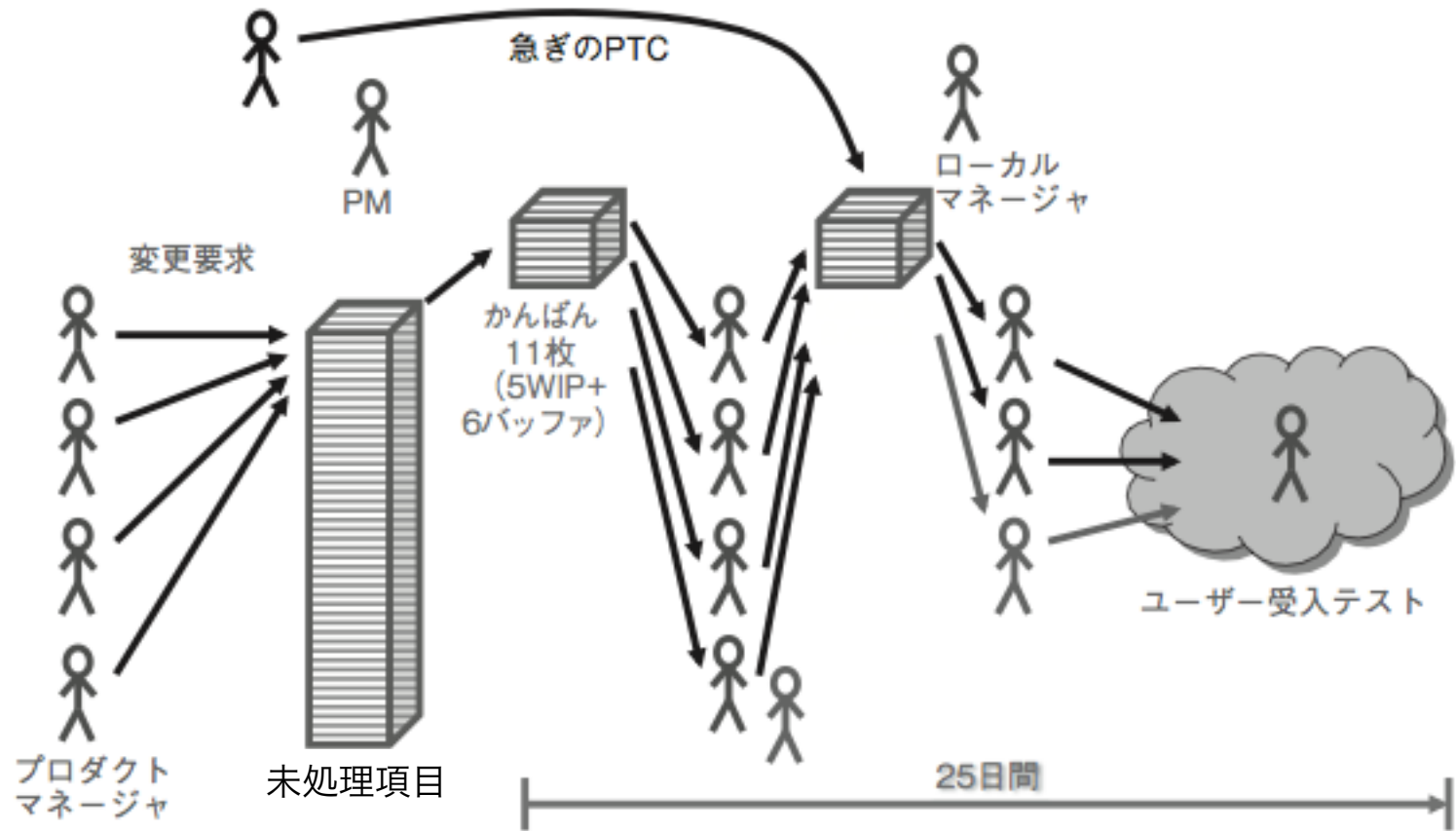
更なる改善

テストチームを 3 人から 2 人に減らし、開発者を 1 人追加する



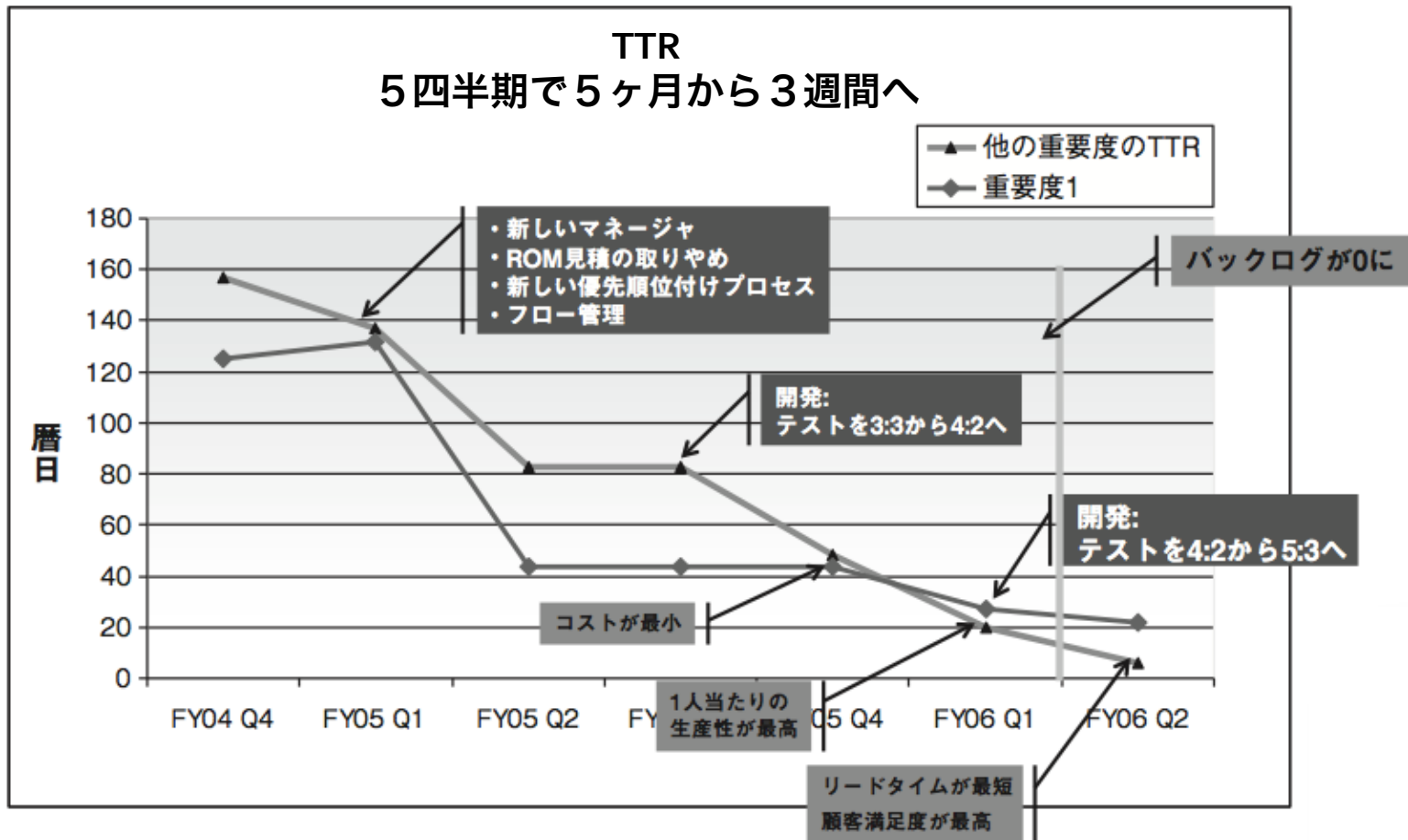
出典：「カンバン ソフトウェア開発の変革」〈リックテレコム〉より

リソースの追加



出典：「カンバン ソフトウェア開発の変革」(リックテレコム)より

TTR(解決時間) Microsoft社の会計年度で表示

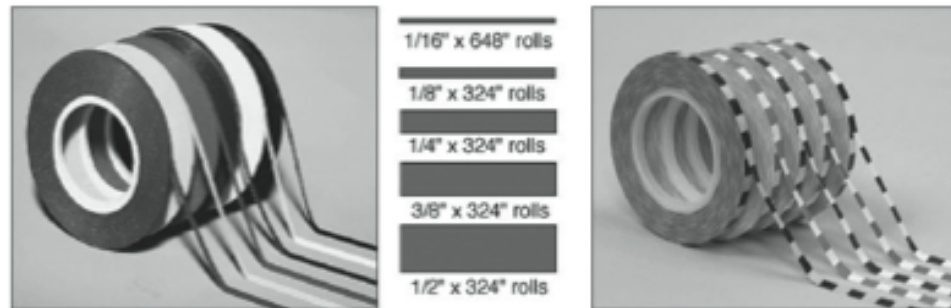


カードウォール

カードウォールによるワークフローの表現 (左から右に流れる)



図 6.2 ホワイトボード用罫線テープ



なおアクティビティのステップに関しては、実行中と完了の両方をモデ

出典：「カンバン ソフトウェア開発の変革」
〈リックテレコム〉より

バッファとキューを追加したワークフロー



キューまたはバッファの列を菱形の管理票で示しているカードウォール
(写真提供：Liquident Holdings, Inc.)

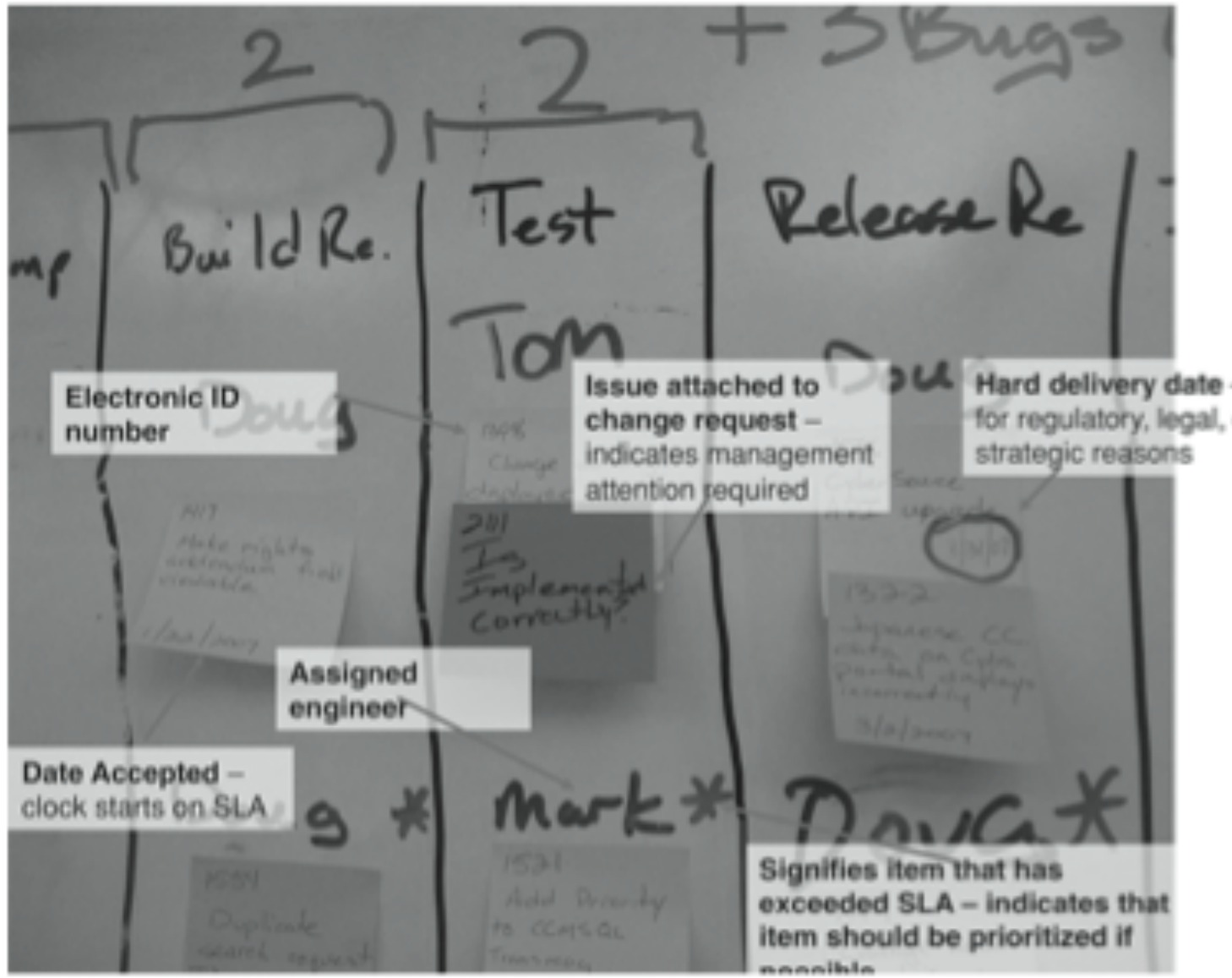


出典：「カンバン ソフトウェア開発の変革」〈リックテレコム〉より

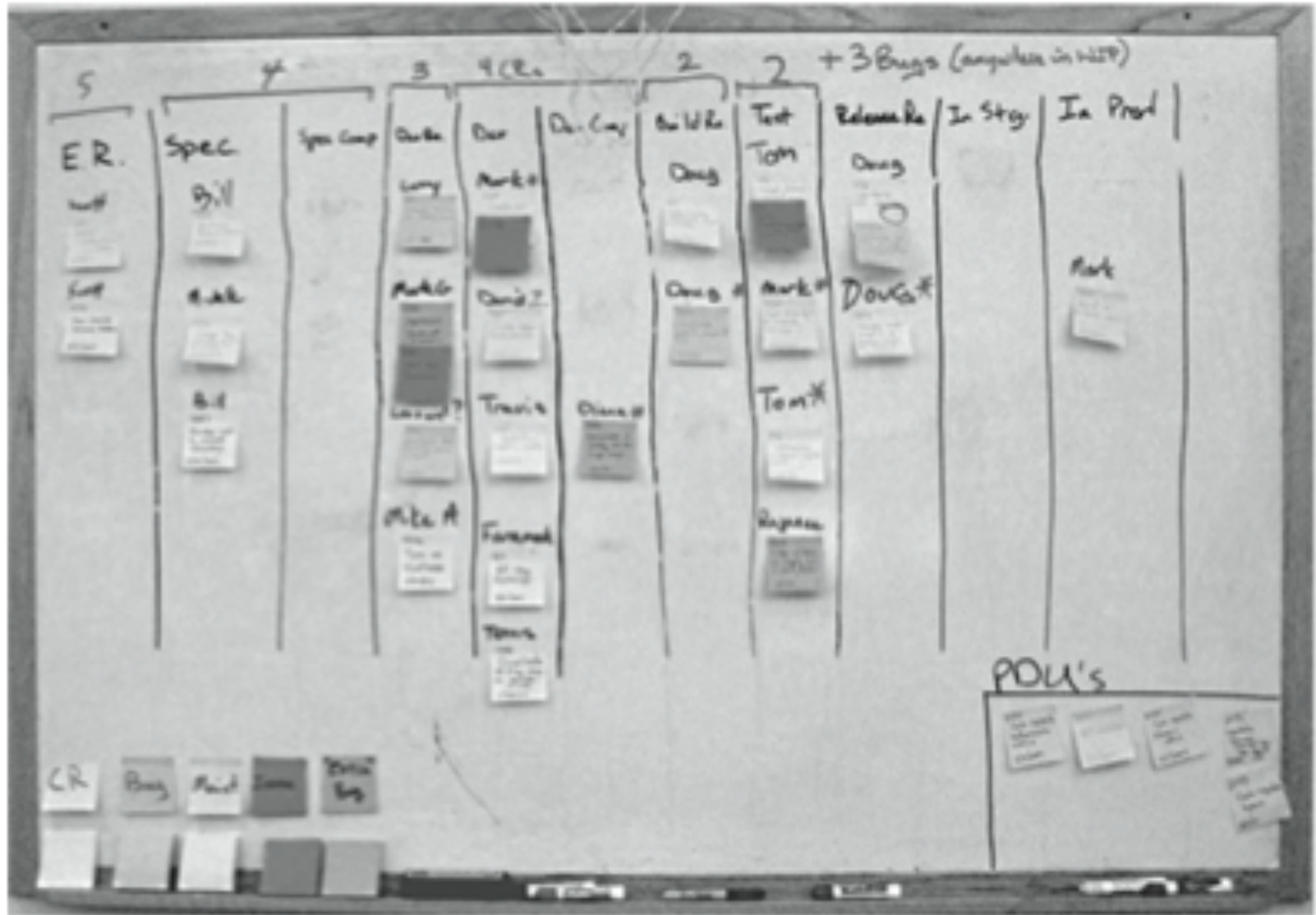
キャパシティ割当てを示すタイプごとのスイムレーンを持つカンバンボード

	入力 キュー	分析 実行中	分析 完了	開発 準備完了	開発 実行中	開発 完了	ビルド 準備完了	テスト	リリース 準備完了	...
変更要求 60%		⋮			⋮					
内部保守 10%		⋮			⋮					
PTC 30%		⋮			⋮					

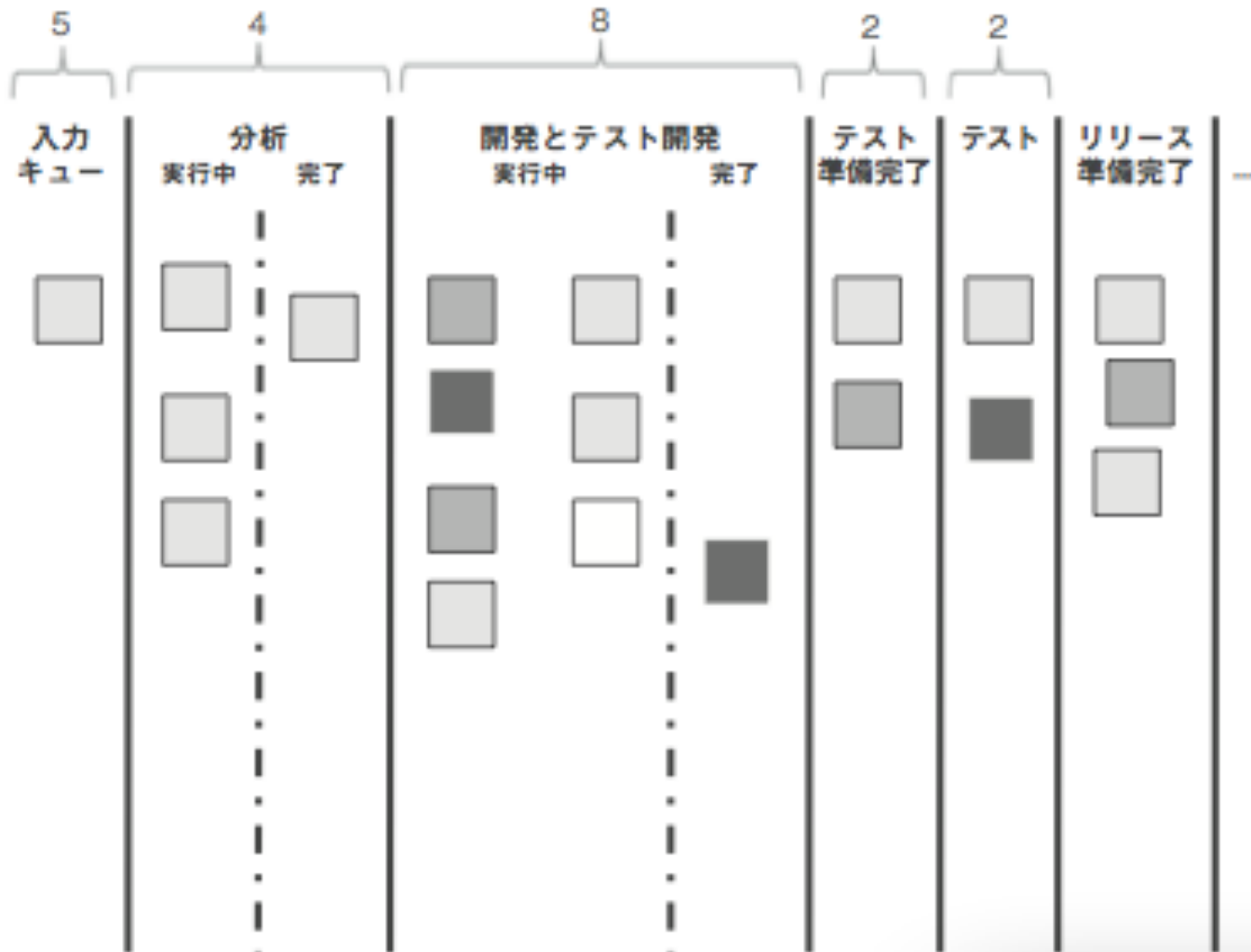
作業項目カードの内部詳細を示すカードウォール



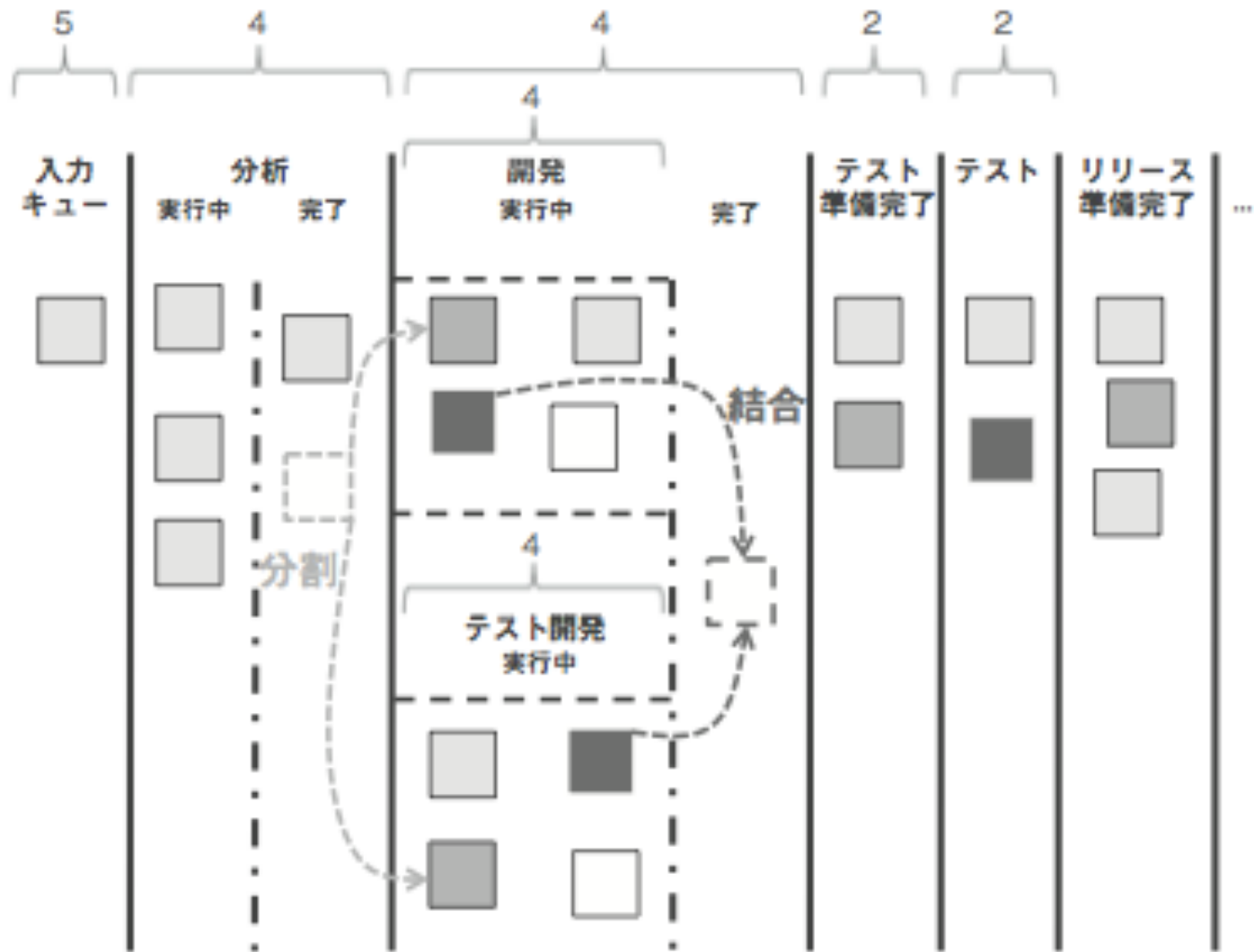
E.R. (エンジニアリング準備完了) 入力キューの表示



同時実行される アクティビティ用の開放された列

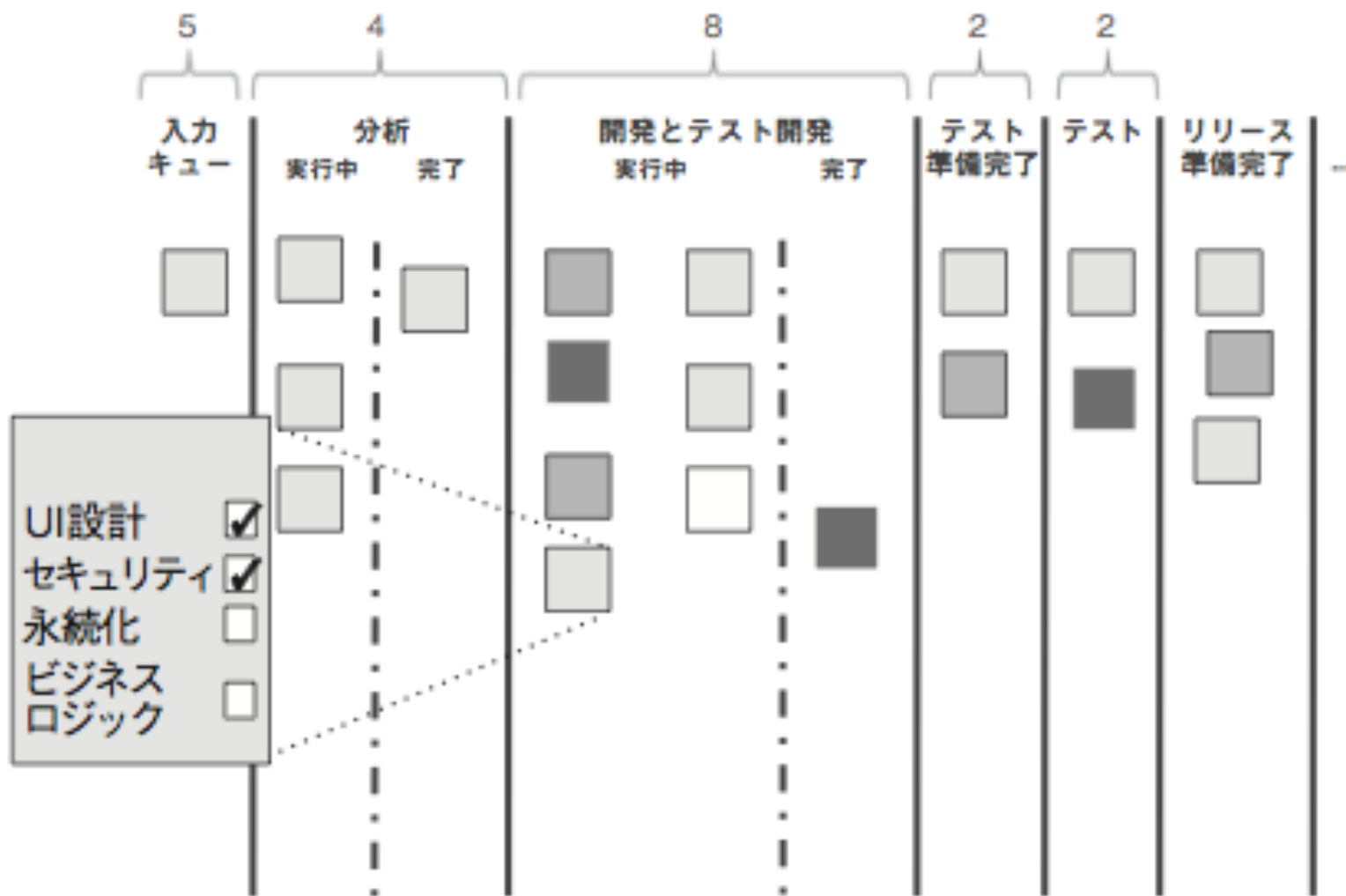


同時実行される アクティビティ用の分割された列

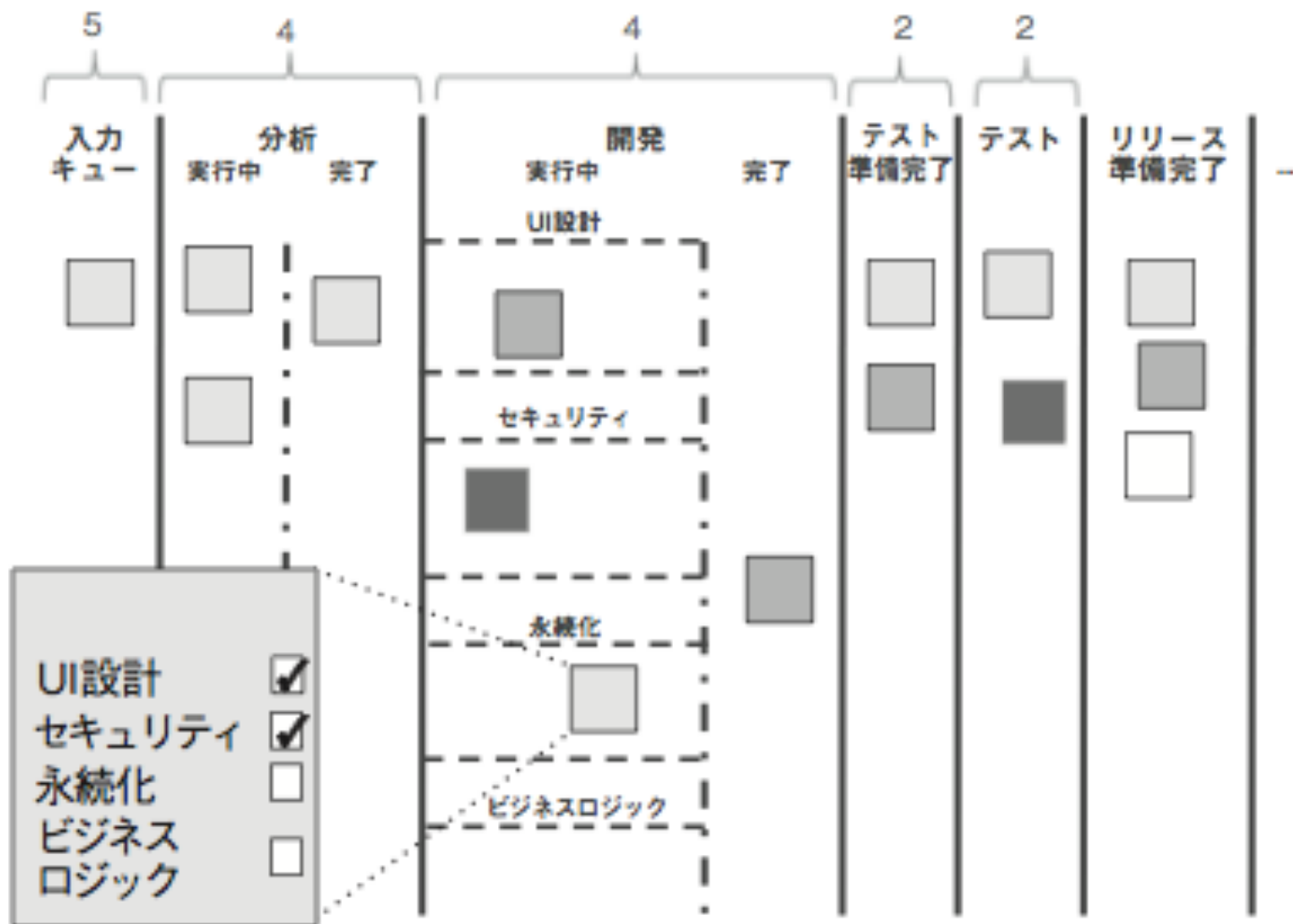


出典：「カンバン ソフトウェア開発の変革」(リックテレコム)より

順不同アクティビティ用の 開放された列

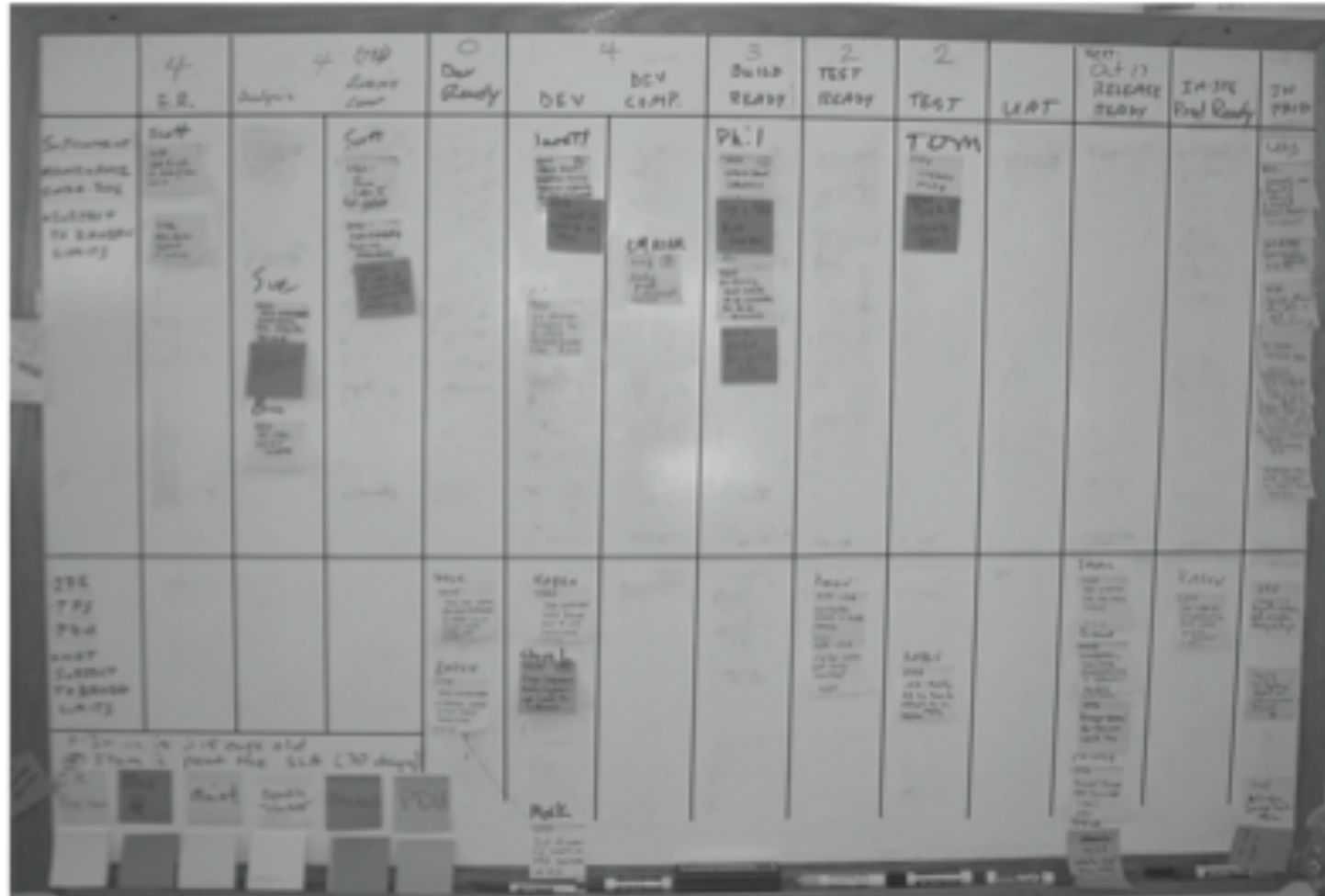


順不同アクティビティ用の 分割された列



カンバンシステムの調整

カードウォールの列の上部にカンバン制限を表示



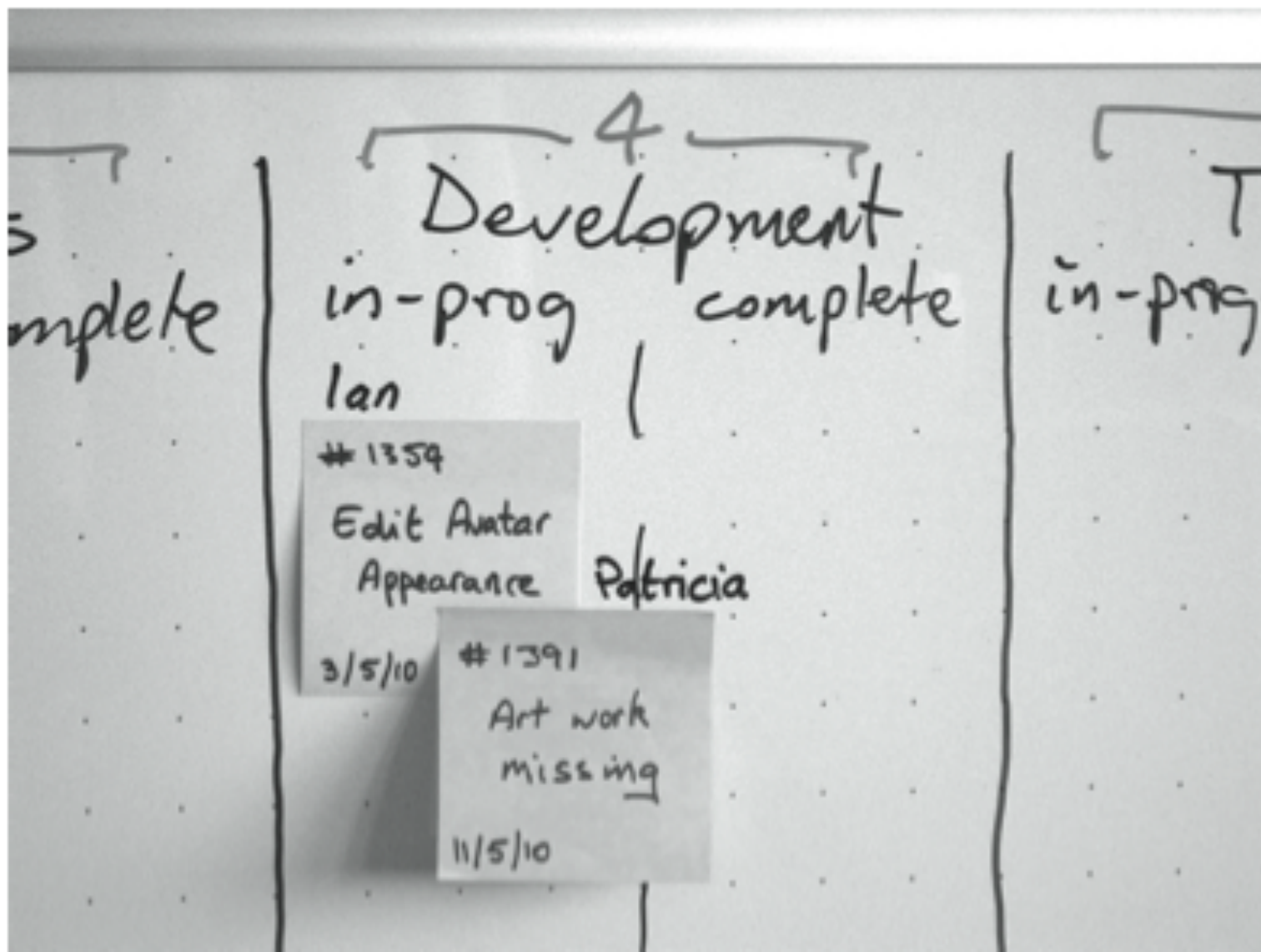
分析(Analysis)の上に4という数字が書かれていて、制限が4であることがわかる。

ところがカードは3枚だけである。 $4 - 3 = 1$ なので、入力キューであるエンジニアリング準備完了から、分析に項目を1つ引き取ることができるという信号が表示されている。

ここで入力キューの制限も4であるが、今あるのは2項目だけである。1つを分析に移すと、残りは1つだけになる。

したがって、次の優先順位会議では、新しく3つの項目を入力キューに入れることができる。

ブロックされている項目に付けられた問題チケットを示すカードウォールの拡大写真



出典：「カンバン ソフトウェア開発の変革」(リックテレコム)より

Corbis社で使った「Digital Whiteboard」アプリケーション



出典：「カンバン ソフトウェア開発の変革」〈リックテレコム〉より

デリバリーリズム

カンバン手法は、タイムボックスに区切られたイテレーションをなくし、優先順位付け、開発、デリバリーの工程(アクティビティ* 2)を分離する。それぞれのリズムに対し、自然な形での調整が可能である。カンバン手法では定期的なリズムという考え自体が不要である。カンバンチームはそれでもソフトウェアを定期的に、それも短い時間軸でデリバリーする。「アジャイルマニフェストの背後にある原則」も満たしている。しかし、不自然に物事をタイムボックスに入れると、さまざまな機能不全が生じるため、そのようにならないようにしている。

デリバリー効率

$$\text{デリバリー効率(\%)} = 100\% \times \frac{\text{トータルコスト} - (\text{調整コスト} + \text{取引コスト})}{\text{ソフトウェアリースのトータルコスト}}$$

コストだけの式

$$100\% \times \frac{1,500,000 \text{ドル} - 500,000 \text{ドル}}{1,500,000 \text{ドル}} = 66.7 \%$$

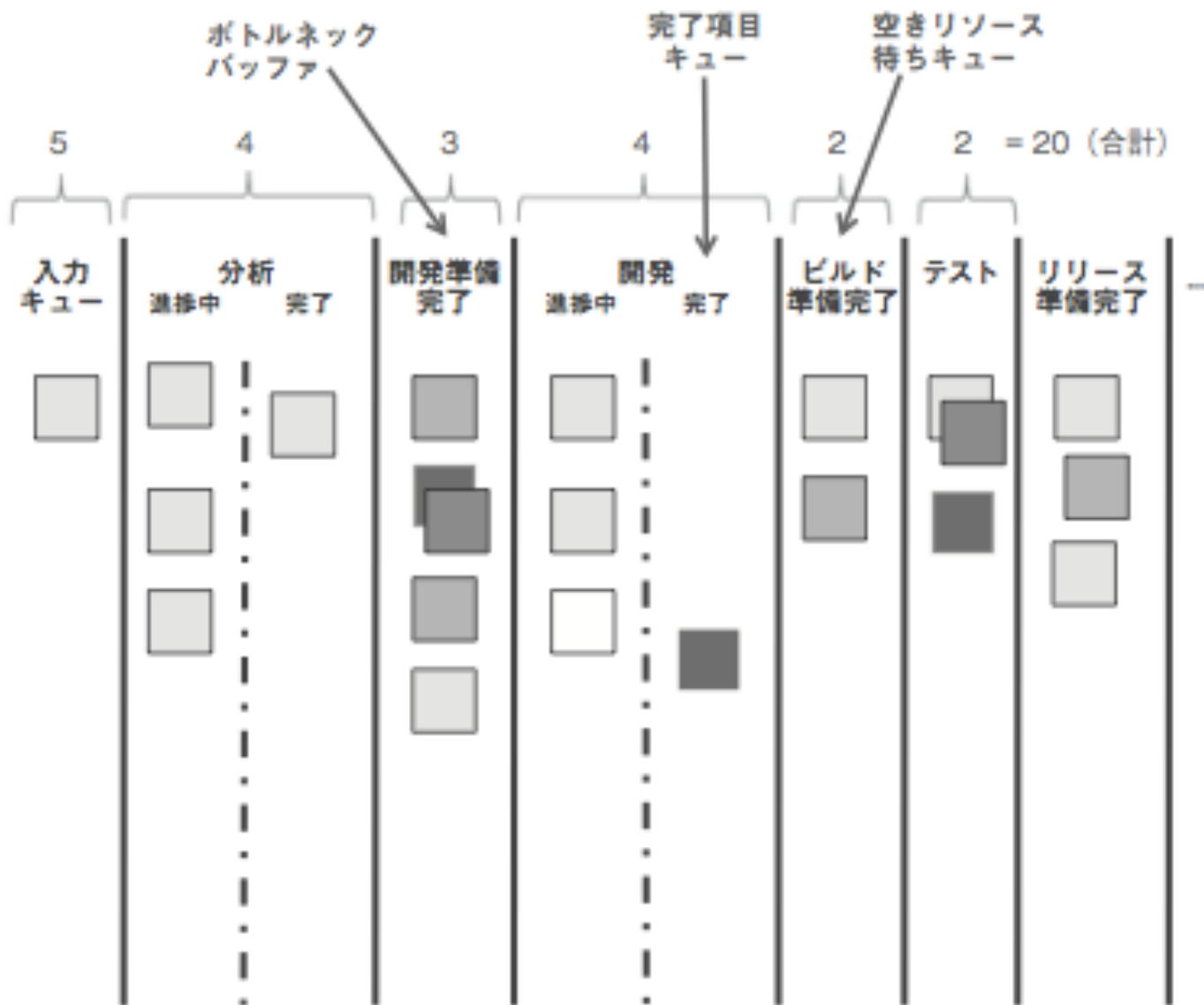
あるリリースでどれだけの価値がもたらされるかが分かれば、デリバリーの頻度をどれだけの長さにすべきかの判断をよりの確に下すことができる。毎月のソフトウェアのデリバリーで150万ドルのコストに対して、200万ドルの収益が見込まれるのであれば、デリバリーによって50万ドルの利ざやが得られる。

$$\text{デリバリー効率(\%)} = 100\% \times \left(1 - \frac{\text{調整コスト} + \text{取引コスト}}{\text{利ざや} + \text{調整コスト} + \text{取引コスト}} \right)$$

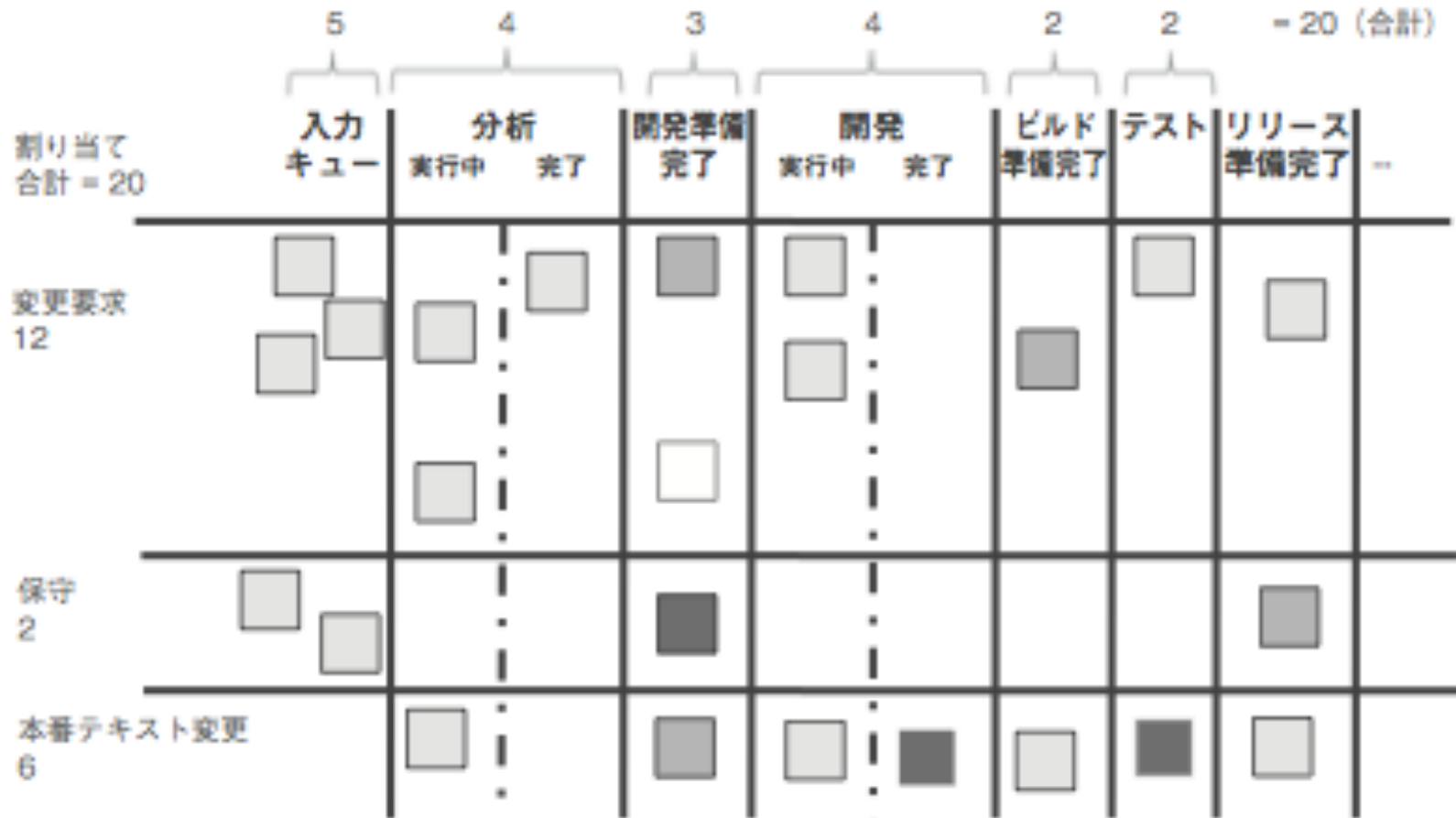
価値を考慮した式

$$100\% \times \left(1 - \frac{500,000 \text{ドル}}{500,000 \text{ドル} + 500,000 \text{ドル}} \right) = 50\%$$

さまざまなキューとバッファを示しているカードウォール



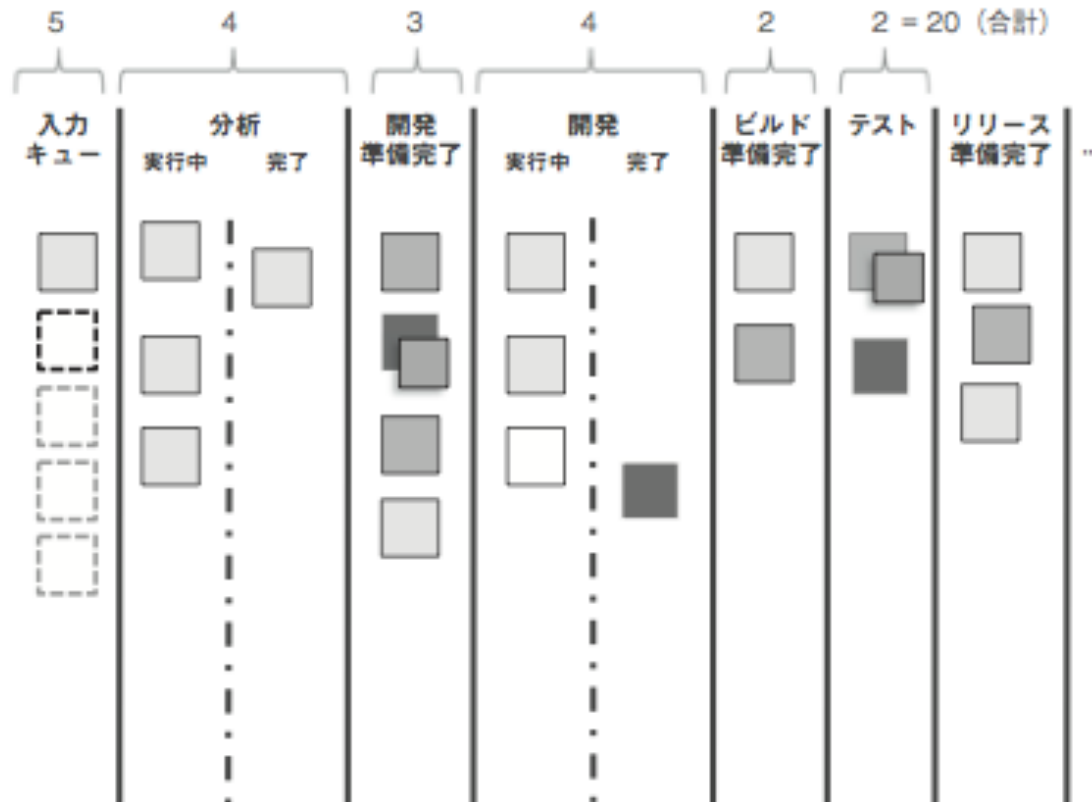
各レーンの仕掛け制限をはっきりさせた各作業項目分類のスイムレーンを示している
カードウォール



出典：「カンバン ソフトウェア開発の変革」(リックテレコム)より


優先順位分類

サービスクラスごとの容量割り当てを示すカードウォール




(割当て)

特急 (白) :  +1 = +5%

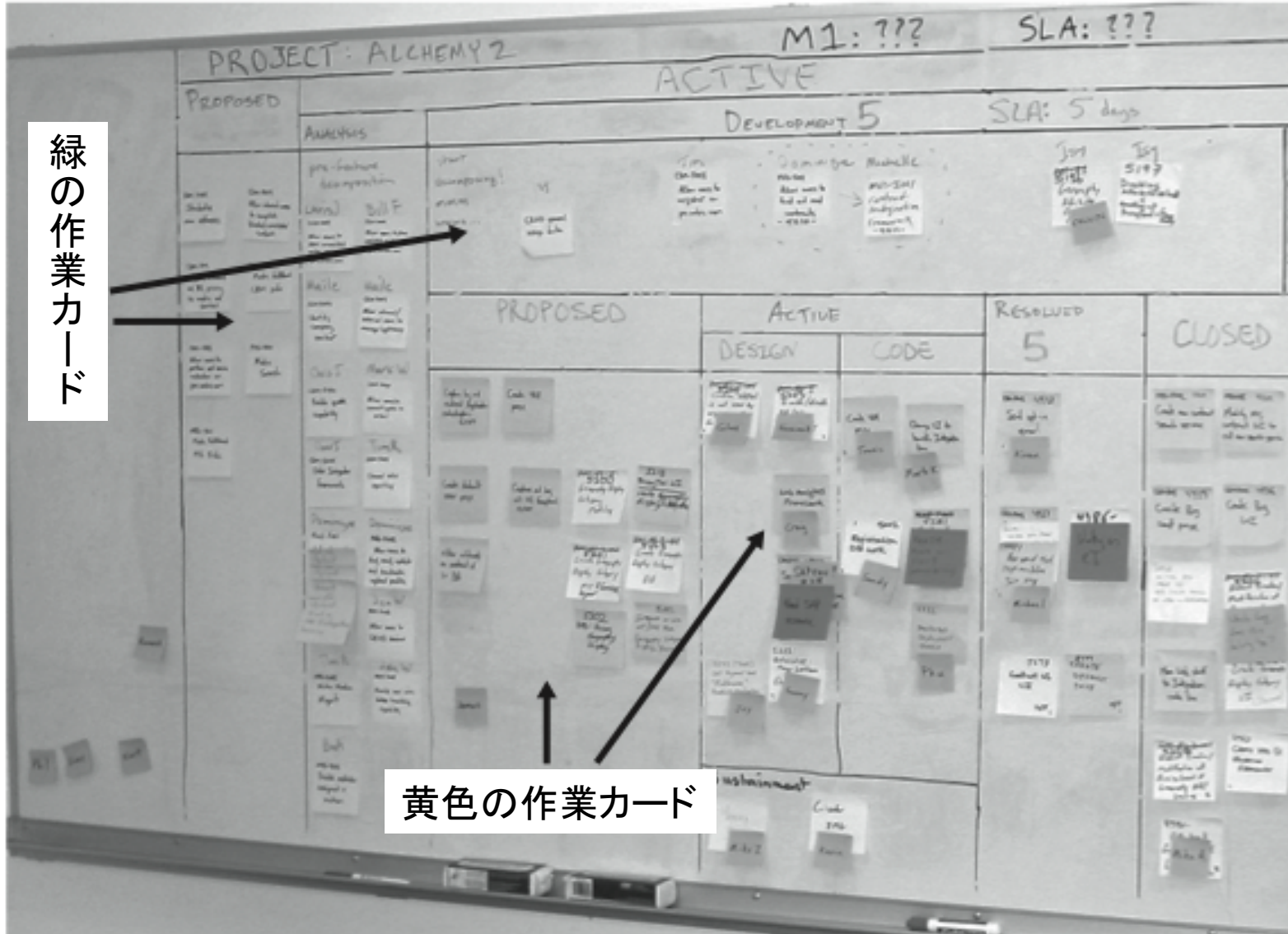
必須納期 (紫) :  4 = 20%

標準クラス (黄色) :  10 = 50%

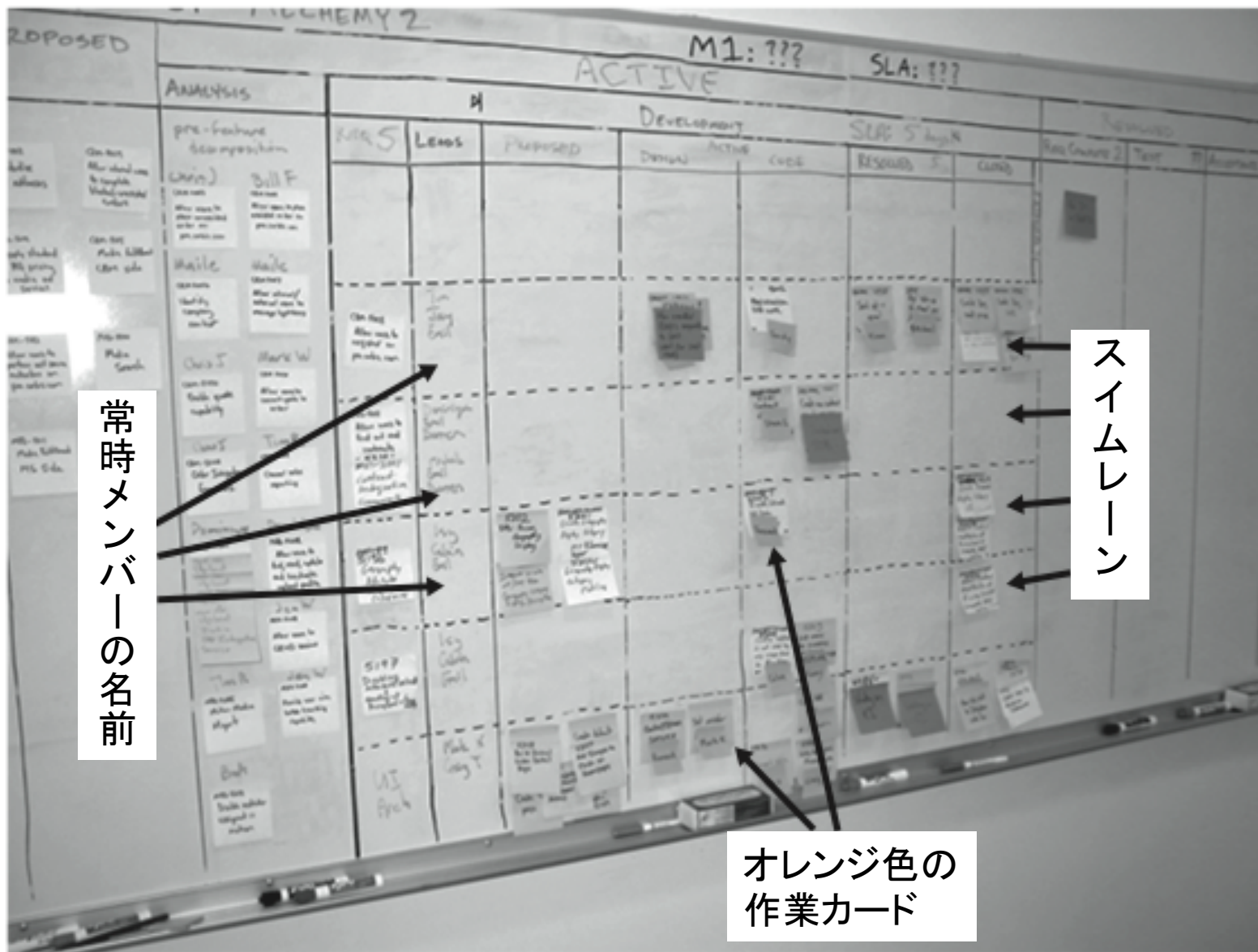
無形クラス (緑) :  6 = 30%

出典：「カンバン ソフトウェア開発の変革」
〈リックテレコム〉より

2階層ボードの写真



スイムレーンのある2階層ボードの写真



常時メンバーの名前

スイムレーン

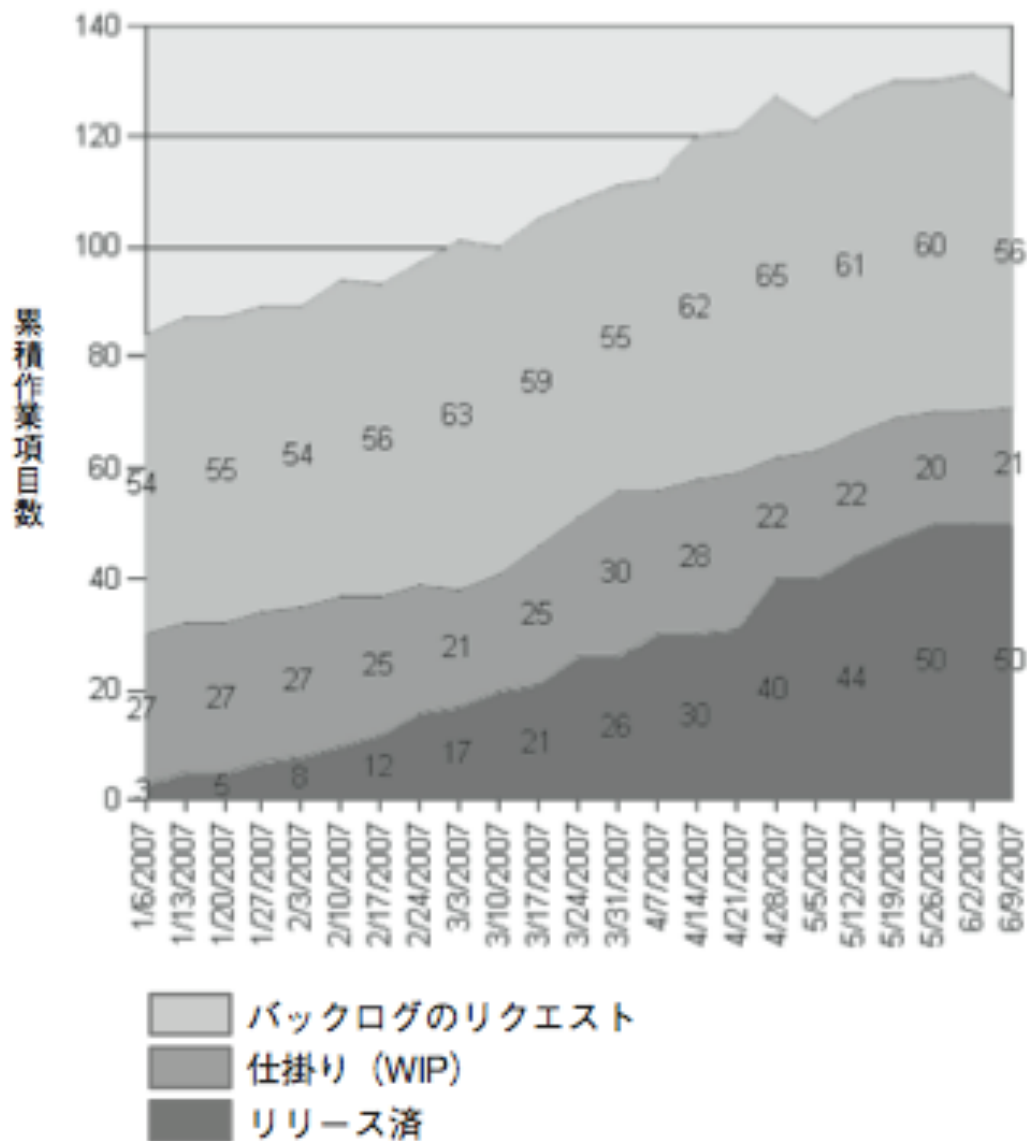
オレンジ色の作業カード

メトリクスと マネジメントレポート

- 仕掛かりの追跡
- リードタイム
- スループット
- 初期品質

仕掛りの追跡

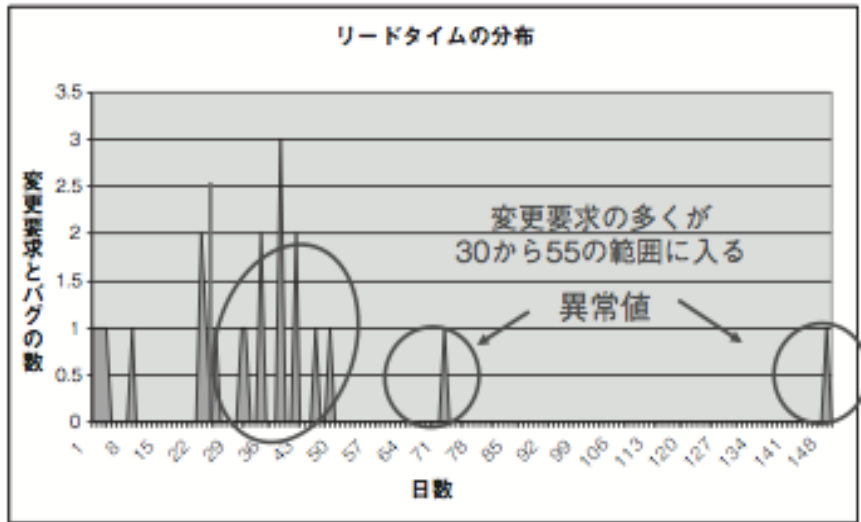
カンバンシステムでの累積フローダイアグラムの例



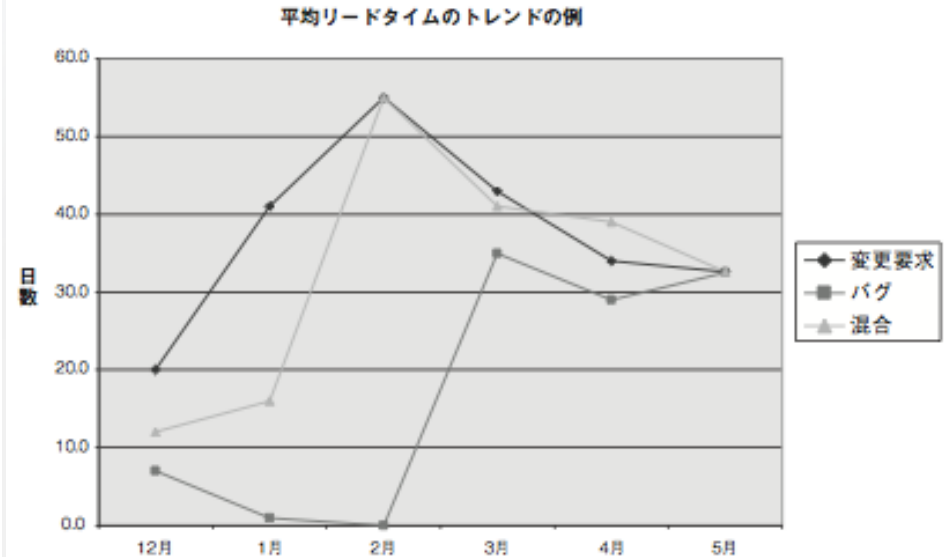
出典：「カンバン ソフトウェア開発の変革」
〈リックテレコム〉より

リードタイム

リードタイムのスペクトル分析の例



平均リードタイムの傾向の例



リードタイムと納期パフォーマンスを示すレポートの例

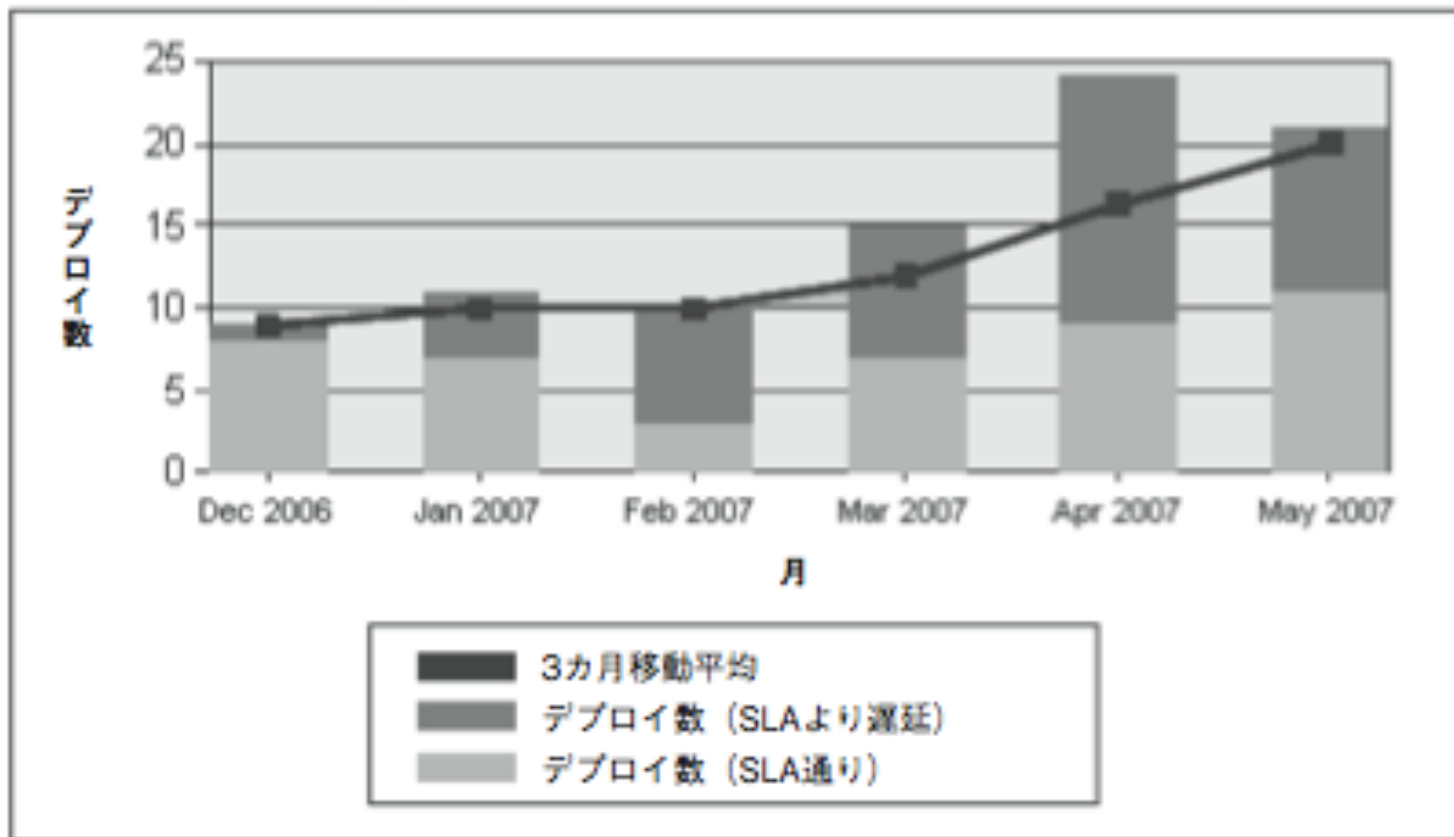
間隔	リードタイム (平均日数)			納期パフォーマンス (%)	
	目標	2007年5月	2006年12月から 2007年5月	2007年5月	2006年12月から 2007年5月
着手からリリースまでのリードタイム (変更要求とバグフィックス)	30	32.5	31.1	52	50
着手からリリースまでのリードタイム (変更要求のみ)	30	32.6	40.4	50	30
着手からリリースまでのリードタイム (バグのみ)	30	32.5	19.6	55	75

出典：「カンバン ソフトウェア開発の変革」
〈リックテレコム〉より

スループット

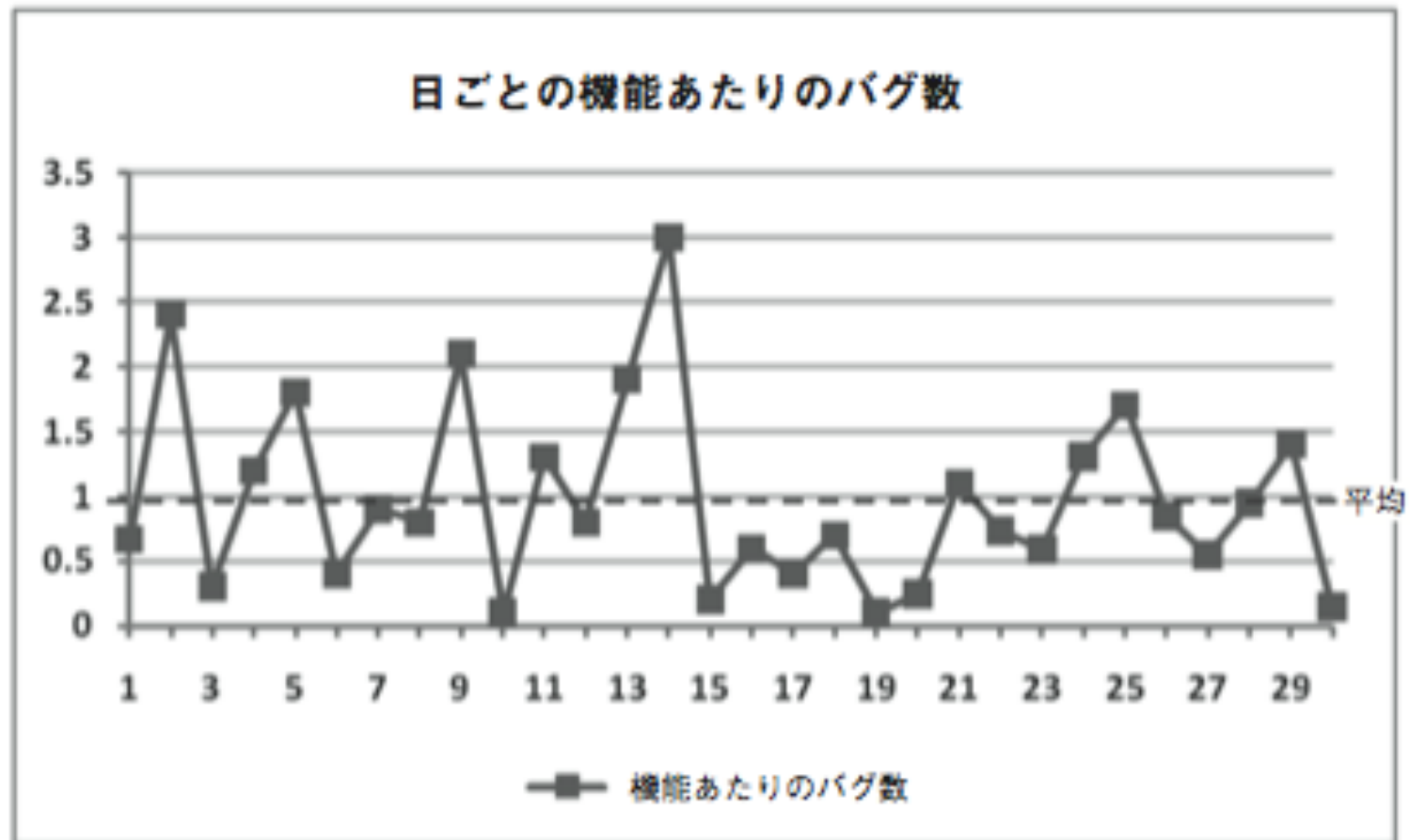
スループットを表す棒グラフの例

スループットと生産数



初期品質

機能あたりの欠陥数を示すグラフ



出典：「カンバン ソフトウェア開発の変革」(リックテレコム)より

3 種類の改善機会

- ボトルネック
- ムダ削減
- ばらつき軽減

改善の5ステップ

- ステップ 1: 制約を識別する。
- ステップ 2: 制約の活用方法を決定する。
- ステップ 3: その他のことはすべて、
ステップ 2 での決定に従属させる。
- ステップ 4: 制約を補強する。
- ステップ 5: 惰性を避け、次の制約を識別して、
ステップ 2 に戻る。

ソフトウェアカンバン手法の概要

長瀬 嘉秀