2017
5/26

JaSST'17東北
テストエンジニア版
RPG風スキルマップ

@nemorine

自己紹介

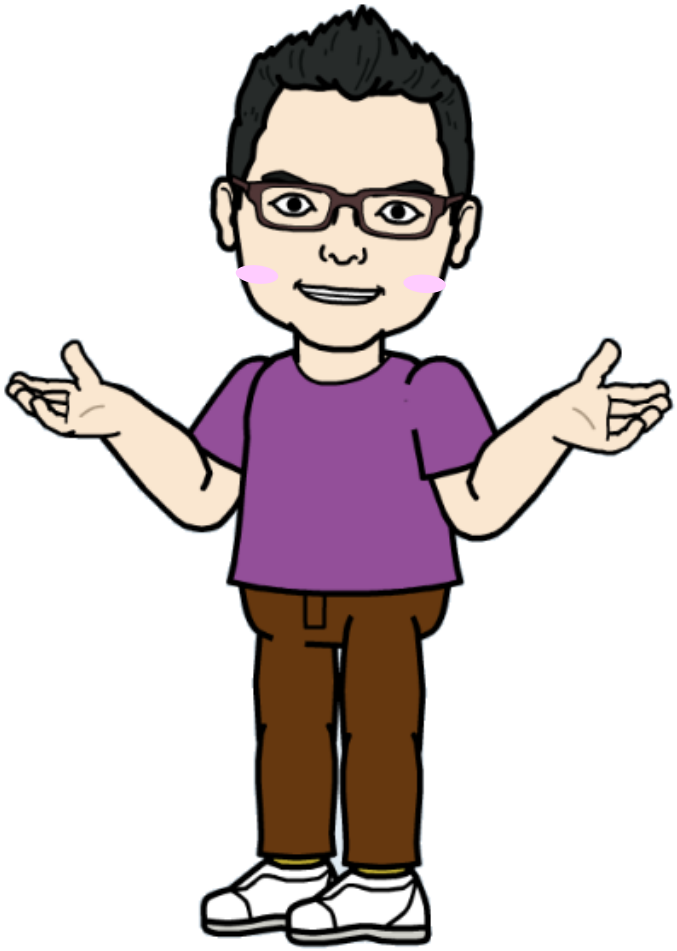
根本紀之
東京エレクトロン 札幌事業所
言語　C#、Java、C++

社外活動

- JaSST北海道実行委員
- JaSST東北実行委員（初代委員長）
- TEF道
- Agile札幌メンバー

資格

- JSTQB FL / AL(Test Analyst)
- Certified Scrum Master



(c) nemorine

範囲が大きい！

「育成」

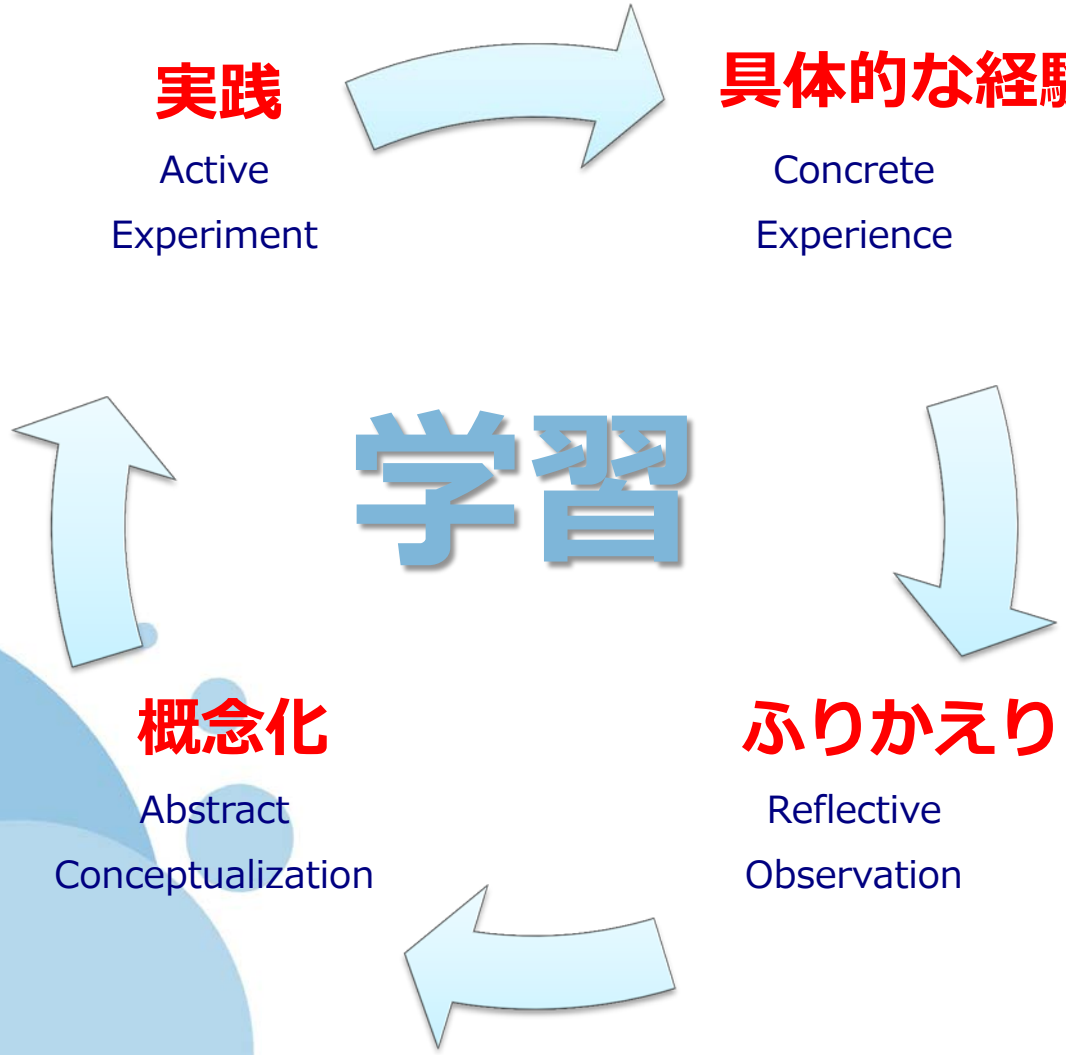
学習との関係は？

(c) nemorine

「学習」を
手助けするのが
「育成」

(c) nemorine

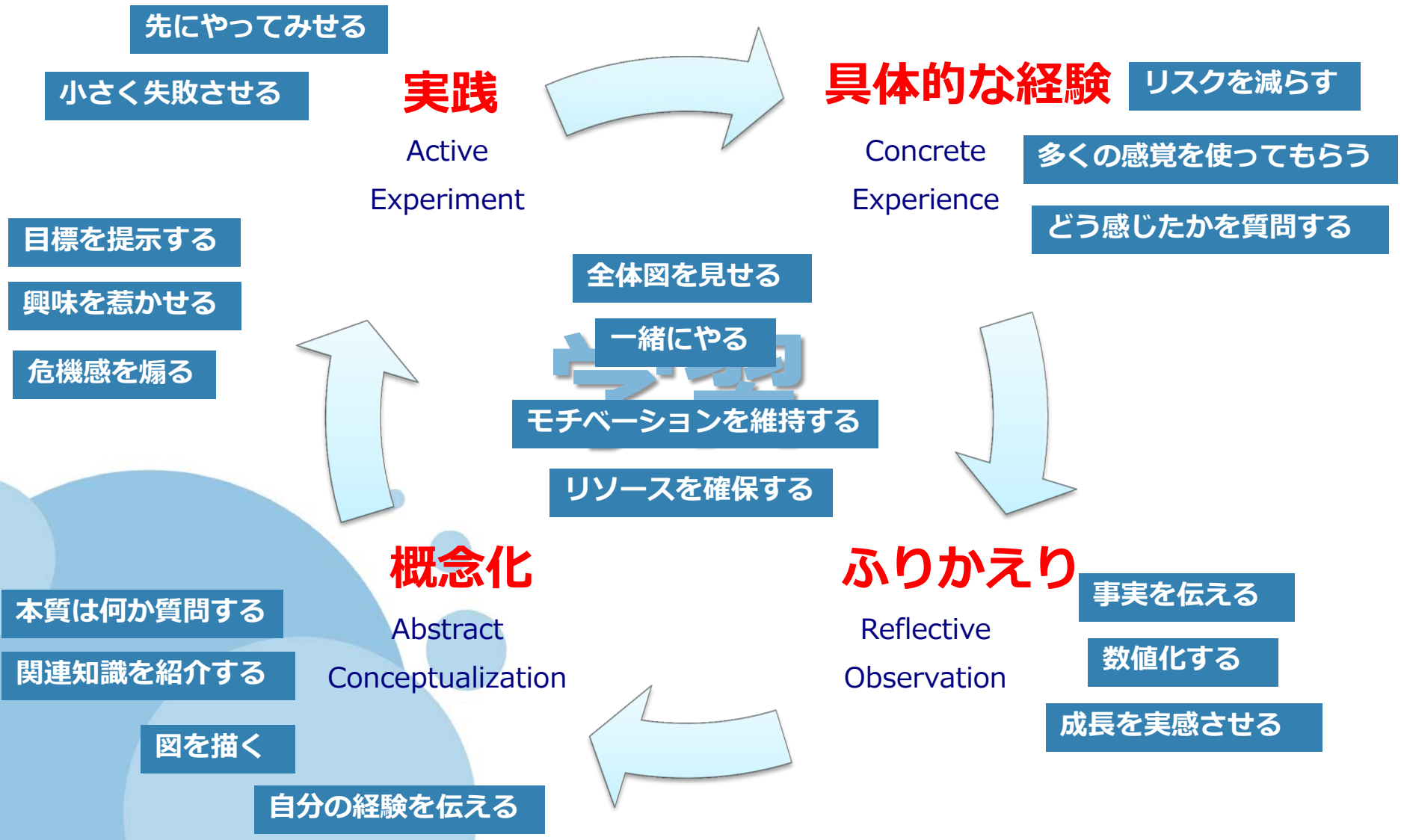
育成のイメージ@nemorine



デービッド・コルブの「経験学習モデル」

(c) nemorine

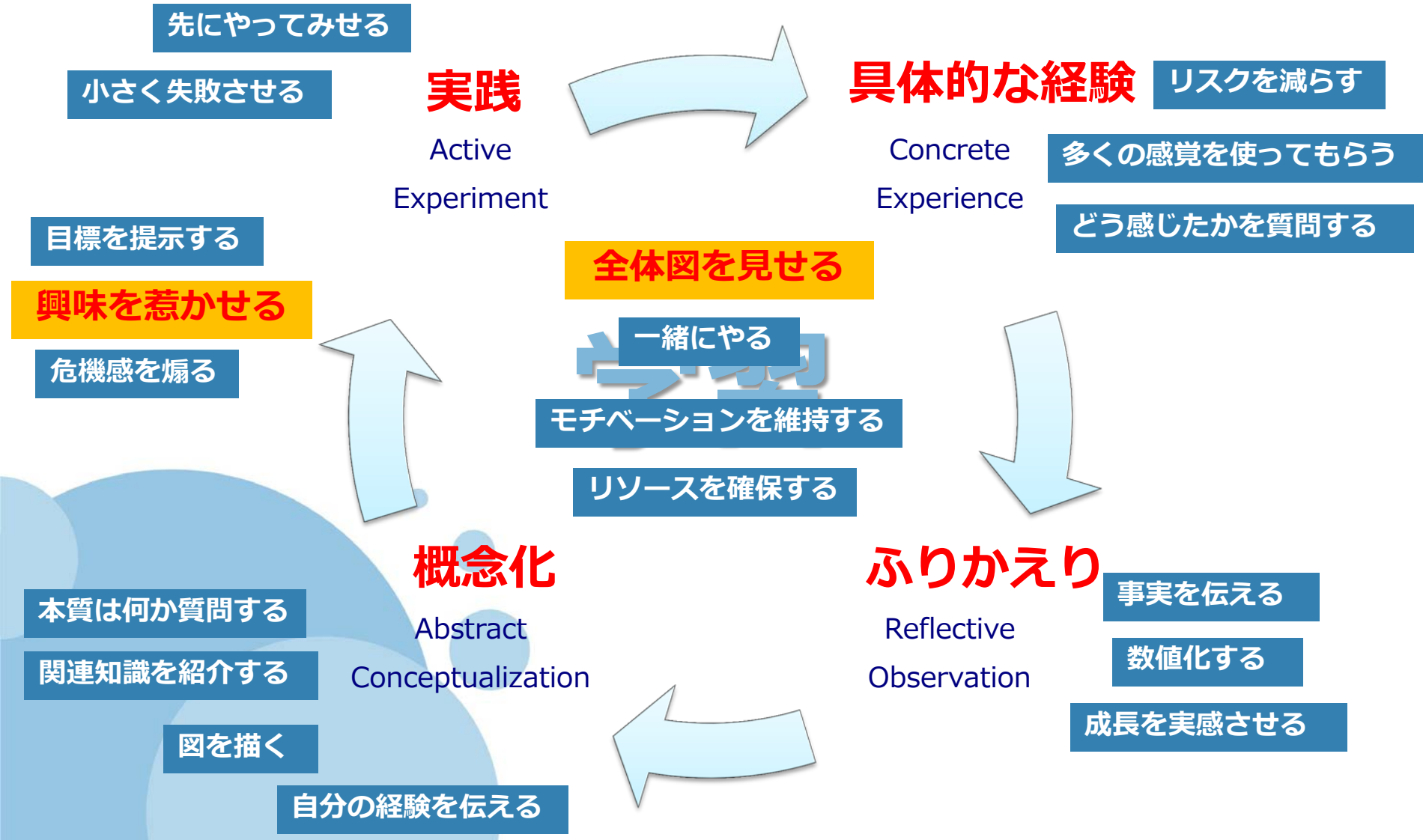
育成のイメージ@nemorine



デービッド・コルブの「経験学習モデル」

(c) nemorine

育成のイメージ@nemorine



デービッド・コルブの「経験学習モデル」

(c) nemorine

RPG風スキルマップとは？

- 元々は楽天の川口さんが教えてくれたもの。
- チームで楽しく使うことを考えたスキルマップ。
- 巨大な敵を倒すためには色々な職種の人が必要だよね！というのがすぐに分かる。
- オリジナルネタは日経SYSTEM（2015年11月号～）に掲載されている。

  ※現在はWebでも閲覧可能

  記事一覧：http://itpro.nikkeibp.co.jp/atcl/column/16/102500242/

(c) nemorine

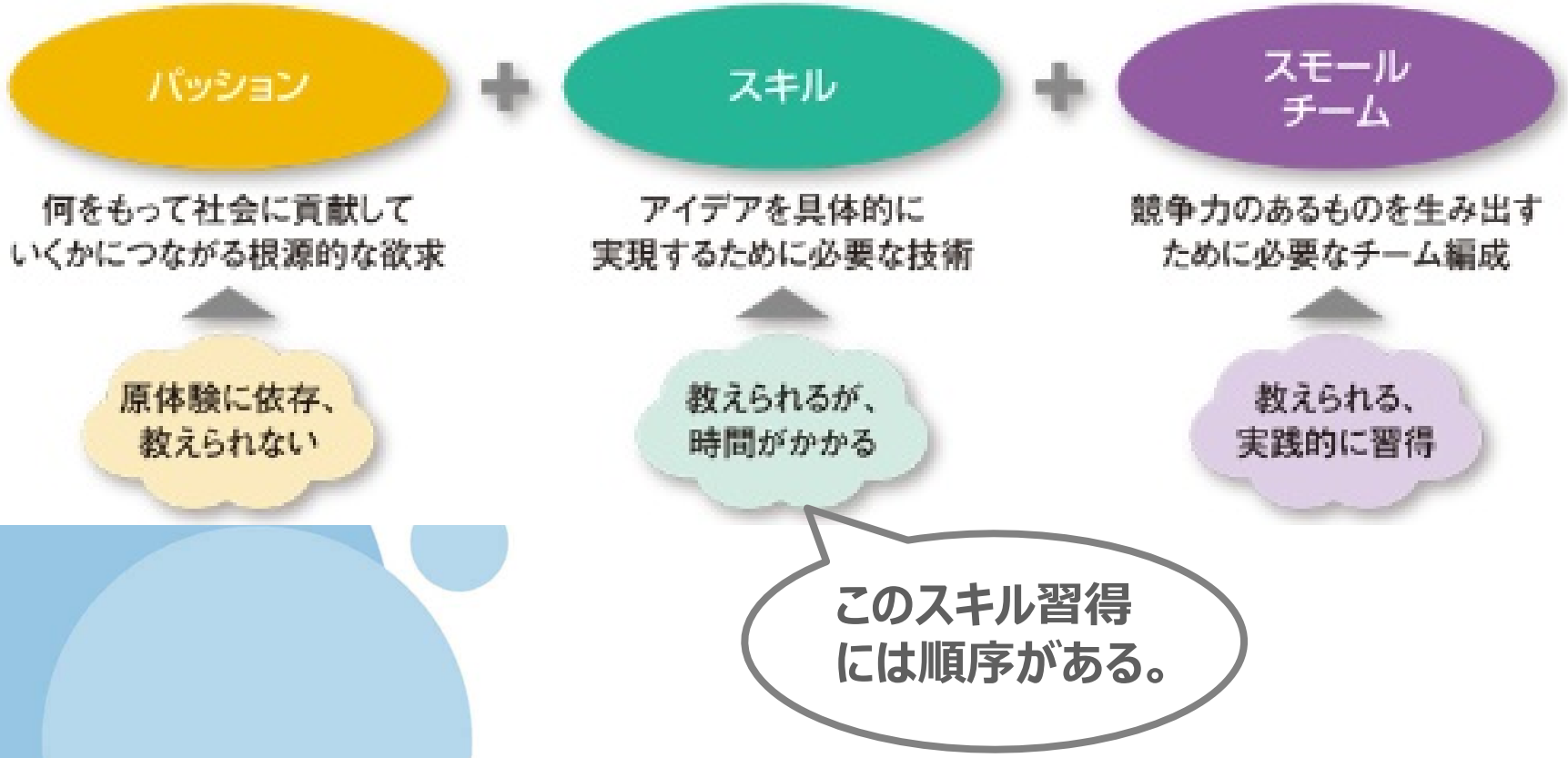
JaSST'17北海道
の基調講演者
2017/09/08(金)

川口さんの記事紹介

(c) nemorine

川口さんの記事ダイジェスト①

- パッション、スキル、スモールチームがこれからの企業活動に必要である。



http://itpro.nikkeibp.co.jp/atcl/column/16/102500242/102500001/

(c) nemorine

川口さんの記事ダイジェスト②

- 個人のスキル＋チームのスキルの地図を作る。

	戦士	武闘家	僧侶	魔法使い	盗賊	旅芸人
特性	ソフトウェアはコードを書かなければ動かない！開発言語に精通し、素早く的確な実装を行う。テストを書きながら、スピーディかつ高品質なコードを生み出し、要件を満たしていくのが戦士の修業技だ！	もっと早く、もっと丁寧に。アプリケーションの本質をつかみ、各種の設計手法を適切に使い、保守の高いソフトウェア開発を実現するのがソフトウェアのプロ、武闘家の十八番だ！	コードは健全に管理できているか？メンバーの知識は共有されー人休んでも仕事がよどみない状態になっているか？開発チームの健全性を維持するのが僧侶の役目だ！	あらゆる操作シーケンスを手作業で試すことなど不可能だ。魔法使いは不可能を可能にし、最低限の工数でほぼ全体をカバーするテストを設計し、省力化しながら確実に実行する！	すべての奥義を自分ですることはできないし、非効率だ。コミュニティや書籍、達人たちとの相互作用を用いて効率的に知識を吸収し、試していくのが最強のスキルセットだ。スピーディに成長していく！	要件定義、利用者、マネージャ、コミュニティ、チームメンバー…。あらゆる人々と効率的で健全で前向きなコミュニケーションをとるスキルは里遠ござせない。プロセスをゲームに！目標をタスクに！
LV1	使用言語の初級研修を受けるか、独学によって本番向けのコードを生み出せるようになっている。開発用サーバにアクセスできどこにコミットすればリリースできるかを知っている。	クラスやオブジェクトを、明確に意識してコードを書くことができる。	自分の書いたコードを、ほかの人にコードレビューをしてもらうことができる。	要件に従い、確認項目を洗い出すことができる。正常系、少なくとも一つのハッピーパスを書き出してテストすることができる。	社内の自分の専門分野のコミュニティや、詳しい人を知っている。難しい問題が出たときにはその人に質問してみることができる。	成果を出したら発表する。なにかいいことを見つけたら発表する。チームや部門内に迅速に共有するための資料を作ることができる。
LV7	xUNITなどでユニットテストを書くことができる。各メソッドの正常系、異常系を列挙してテストを実装する。どのように実行すればよいかを理解している。	単一責任の原則を意識し、相互会話照会を考えて設計できる。	コードレビューをファシリテートし、前向きな示唆を与えることができる。コードレビューは門番ではなく、前向きな改善を生み出すためのアドバイスが重要と理解している。	ハッピーパス以外のパスを網羅的に洗い出して、最低限の組み合わせですべての国王や操作を一通りテストすることができる	自分にあったトレーニングは自分にしか発見できない。定期的に情報を収し、次に習得とするスキルを絞られるトレーニングを提案できる。	ConfluenceとJIRAは私に任せろ。ドキュメントとコーディングの設計をして、円滑で効率的なコミュニケーションを作り出すことができる。
LV14	ユニットテストのカバレッジは十分だ。かつ、スローテストはなく、ごく短い時間にテストスイートを実行することができる。	業務要件とマッチした利用価値の高いクラスやオブジェクトの設計を意識している	ソースコード管理はGitだ。コードの共同所有を推進し、チームの誰もが自分の生み出したコードを理解した状態に持っていける。	步機能要件や外部システムとの接続など、要件にかかわらないテスト項目の洗い出しとテストの実施を計画できる。	世界の実践者コミュニティは専門情報の宝庫だ。いくつかの勉強会に参加してみて、自分にあった、安心で学び習慣会やコミュニティを見つけている！	開発用のサーバ環境はスピードを支配する。必要な開発ツールがなければ生産性は上がらない。適切なタイミングを見切って先手を打って環境を整備できる
LV21	ユニットテストが充いていないテストもある。各テストの重要性は変化する。必要なテストを常に実行するセットにおさめ、常に実行しないテストスイートも同意する。	業務のヒアリングから言葉の抽出し、モデリングを行い、クラス設計や動作シーケンスなど必要な設計と表現/伝達/文書化を行うことができる	Jenkinsなどのツールを使って、日々のコードの健全性は常にチェックできるようにしている。毎日、チェックするのだ。	テストデータを用意できる。ハッピーパス、全量データ、負荷テスト用、特殊ケースなど必要なテストデータをスクリプトなどを駆使して作り出せる	必要なコミュニティがなければ自分で作れば良い。自ら自律的なコミュニティを立ち上げ、社内または社外の人が集う環境を整備した！勉強会の開き方を熟知した。	インタビューはソフトウェアの成否を決める重要な情報を得る手段だ。利用者やステークホルダーへの質問を設計し、インタビューを行い、まとめることができる。インタビュー結果の集合から要件を抽出することができる
LV28	テストファーストだ。先にテストを書いてからメソッドを実装する。ペアプログラミングでナビゲーターとドライバーが適切にコミュニケーションしながら進む。	DDDやデザインパターンを利用して、最後に有効な設計を適用できる引き出しを持っている。	リリース作業に手作業が多いると危険が多い。テスト作業を自動化し、リカバリープランを用意しておく。本番機と開発機の差異も整理されになく、スムーズで高頻度なリリースを行えるように整備できる。	探索的テストの名人だ。他の人がいくらやっても発見できないバグを、いつも見つけてしまう。	技術を提供する相手との協調/協力関係は作れているか？意識しているか？あなたの活動を上司に報告して、理解を得ているか？	コーチングはチームやエンジニアに同僚に働いてもらうための必須のスキルだ。よい開発作業になって、気持ちよく仕事を進められるように手助けするぞ。

(c) nemorine

http://itpro.nikkeibp.co.jp/atcl/column/16/102500242/102500002/?P=2

川口さんの記事ダイジェスト③

- 個人の型を知る。

起用貧乏型

職人気質型

バランス型

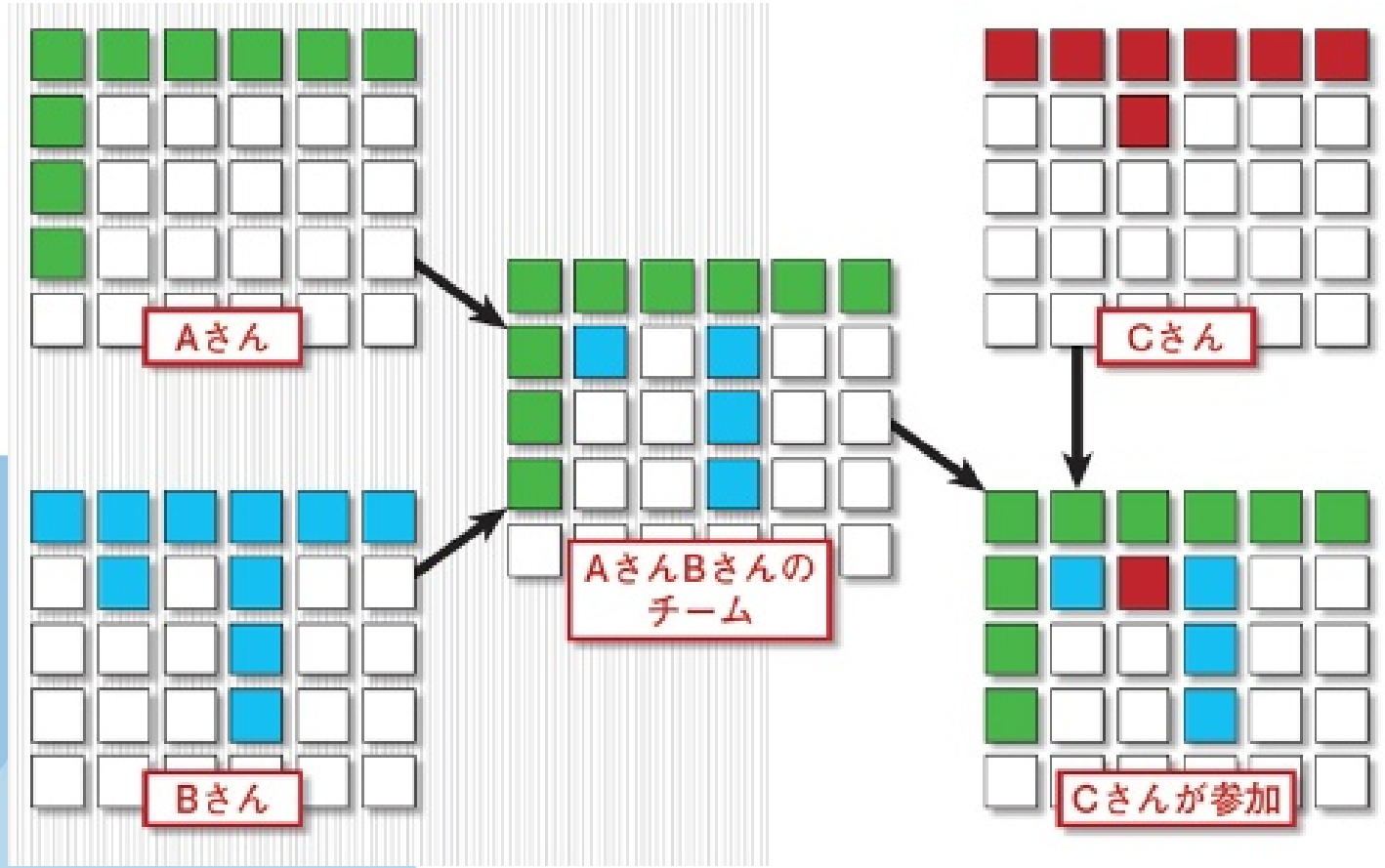
コミュ型

(c) nemorine

http://itpro.nikkeibp.co.jp/atcl/column/16/102500242/110700004/?itp_leaf_index

川口さんの記事ダイジェスト④

- チームの力を知る。



(c) nemorine

http://itpro.nikkeibp.co.jp/atcl/column/16/102500242/110700005/?itp_leaf_index

ということで、、、

テストエンジニア版の
RPG風スキルマップを
作ってみました！！

(c) nemorine

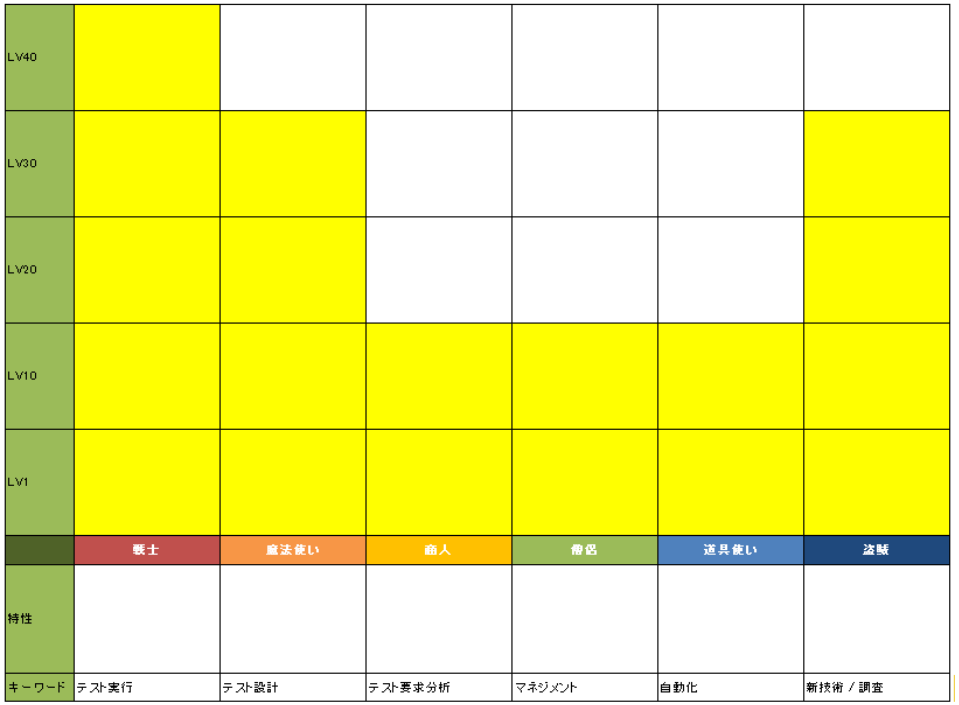
テストエンジニア版RPG風スキルマップ

LV40	CheckingからTestingへ。書かれているテストケース以外のテストを実施し、バグを発見することができる。	スケジュール、納期、網羅度、重み付けを考慮しつつ、目的に適したテストアーキテクチャーを構築することができる。	開発初期から作り込め！ユーザーの真にやりたいこと、ユーザーのドメインを深く理解し、プロダクトの仕様策定から関わることができる。	コーチングはチームやエンジニアに円滑に働いてもらうための必須のスキルだ。よい相談相手になって、モチベーション高く仕事を進められるように手助けすることができ	増えていく自動化プロダクトから不要な自動化は排除し、メンテナンスコストも考えながら自動テストを常に実行することができる。	社内、社外問わず、成果を発表している。発表のチャンスに自ら飛び込むことができる。
LV30	曖昧な点は事前のレビューでつぶし、テスト環境を準備し、怪しい動きはすぐに開発者に質問し、スケジュール通りに着実にテストを行える。	ソフトウェアの構造、変更による影響範囲や機能の組み合わせを考慮し、漏れなく、無駄がないテストを設計することができる。	ユーザビリティラボやインタビューを通じて得たユーザーからのフィードバックを次イテレーションにインプットすることができる。	突発的な出来事に対しても、優先度を決めスケジュール、品質、機能のトレードオフの提案ができる。	ビルドやデプロイなどCIに関わる一連の作業をJenkinsなどを使い自動化することができる。	社外の実践者のコミュニティは専門情報の宝庫だ。JaSSTなど勉強会に参加してみて、自分にあった、役に立つ勉強会やコミュニティを見つけている。
LV20	特定の機能に対して漏れのないテスト手順、期待結果を記述することができる。	網羅的なテストに必要な因子と水準を洗い出すことができる。状態遷移の技法やデシジョンテーブルなどでパターンを洗い出すことができる。	ユーザーの特性、使用状況から最適なユーザビリティ、パフォーマンスなどを導きだすことができる。	他関連部署やマネージャーなどと案件のスケジュール、品質、機能に関して調整ができる。	UIテストの自動化ツールを使いこなし、システムテストを半自動で効率的に実行することができる。	実務で直接使う以外の技術も、将来のために勉強している。 または業務が楽になるように新しい取り組みを取り入れることができる。
LV10	バグを発見した場合は、再現確認を取り、バグの内容、状況等を正確に報告することができる。	正常系のみならず異常系のテストケースを洗い出すことができる。また過去のバグの傾向から出やすいバグを推測することができる。	どんな要求にも背景がある！ペルソナなどを使いユーザーを特定しながら、要求が生まれた理由と優先度を考えることができる。	プロジェクト管理は、私に任せろ。 メンバーの得意分野を理解し、テスト計画をたて、メンバーの進捗管理をすることができる。	品質は積み上げていくもの。 クラスやメソッドに対する単体テストを記述することができる。	社内の自分が関わる案件の専門分野に詳しい人を知っている。難しい問題が出たときにはその人に質問し、有益な情報を集めることができる。
LV1	テスト仕様書に書かれた手順を実行し、実行結果と期待結果からテストのPass/Failを判定することができる。	テスト技法の基本である同値分割・境界値分析を行うことができる。	ミーティング、ドキュメントなどで、ユーザーからの要求をチームで共有することができる。	自分のチームのテストプロセスを図にすることができる。	Excelやプログラミングで簡単な支援ツールを作ることができる。	分からない問題にあたったときに、本を読んだりネットで検索をし、その問題を理解することができる。
	戦士	魔法使い	商人	僧侶	道具使い	盗賊
特性	ソフトウェアはテストしないとリリースすることができない！テストの実装・実行を行う。着実にテストを実施していくのが戦士の得意技だ！	テストは頭脳戦だ！各種の設計手法を魔法のように使い、目的にあったテストアーキテクチャーを構築していくのが魔法使いの仕事だ	顧客が使ってこそ価値が生まれ、利益が生まれる。本当にユーザーが使いやすいソフトウェアを仕様を策定し、テストしていくのが商人の腕の見せ所だっ！	テストプロセスは健全に管理できているか？メンバーの知識は共有されているか？テストチームの健全性を維持するのが僧侶の役目だ！	あらゆる操作シーケンスを手作業で試すことは不可能だ。道具使いは自動化をすることで、テストの支援を行っていく！	書籍、達人たち、コミュニティなどとの相互作用を用いて効率的に知識を吸収し、試していくのが盗賊のスキルセットだ。スピーディーに成長していく！
キーワード	テスト実行	テスト設計	テスト要求分析	マネジメント	自動化	新技術 / 調査

(c) nemorine

5分間くらいで
色をつけてみましょう

☆マークでもOK

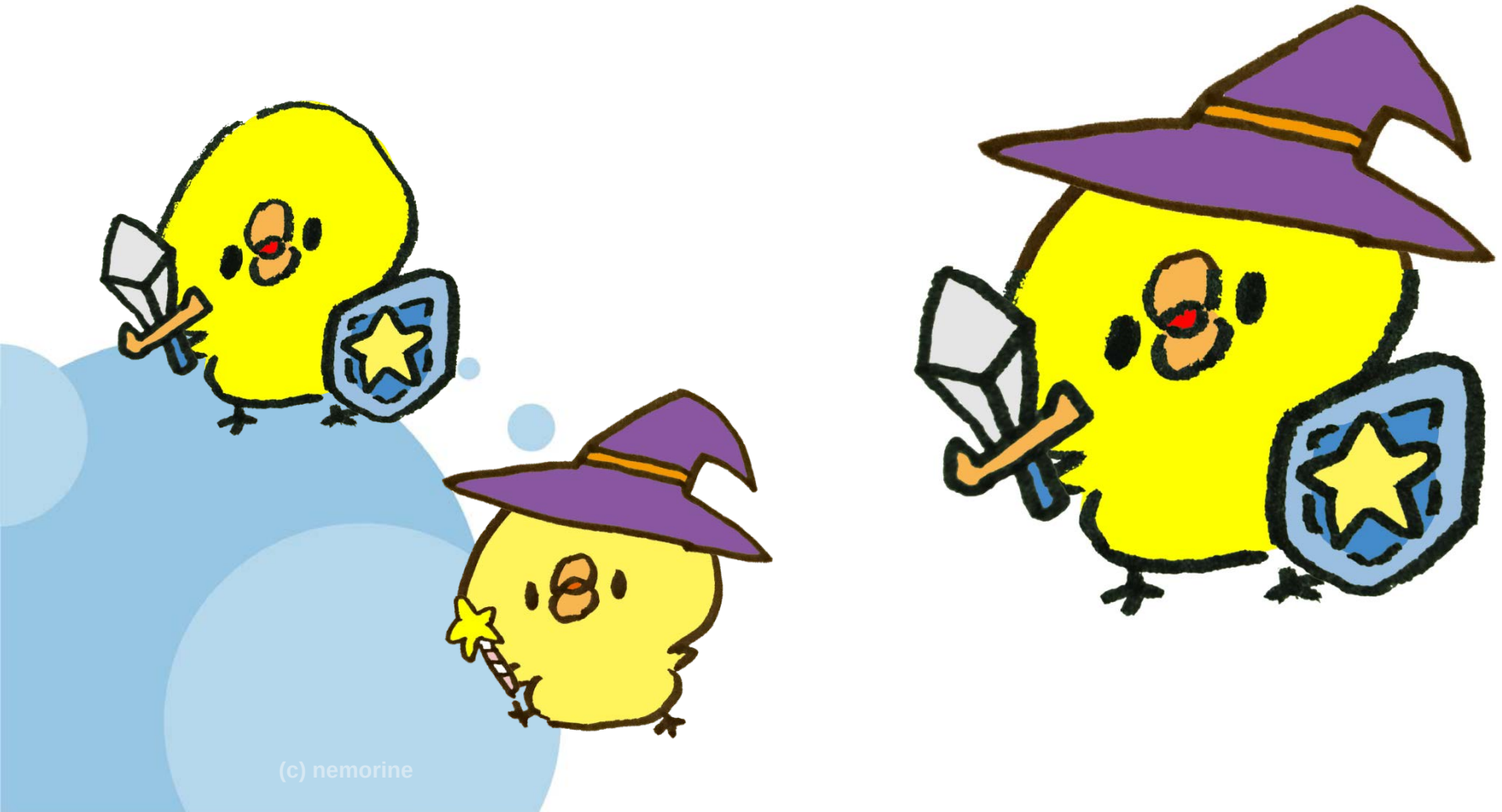


LV40						
LV30						
LV20						
LV10						
LV1						
	戦士	魔法使い	商人	僧侶	道具使い	盗賊
特性						
キーワード	テスト実行	テスト設計	テスト要求分析	マネジメント	自動化	新技術／調査

(c) nemorine

目指すは最強の魔法戦士！

- 力だけではなく、頭も使って最強の魔法戦士を目指しましょう！



(c) nemorine

なんか定義が
微妙なところがある…

と思った方は
挙手してください！

(c) nemorine

そうです。
みんな
“違う”のです。

(c) nemorine

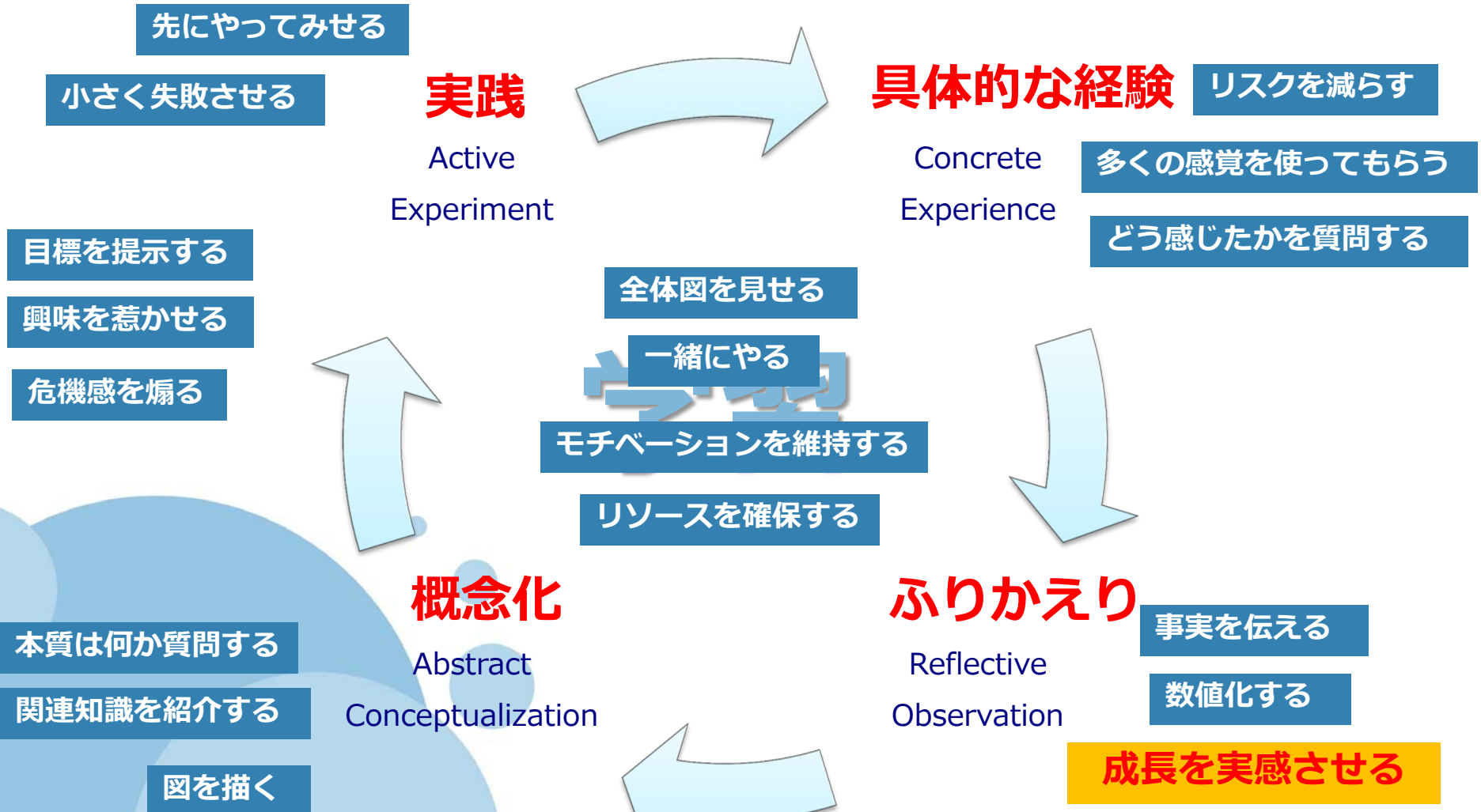
スキルマップを自分達で
再構築してみてください！

チームビルディングにもなりますし、
自分達のチームに必要なスキルを
再認識する機会になります。

スキル定義としては**Test.SSF**が
ありますので詳しくしりたい人は
参考にしてください。

(c) nemori

育成のイメージ@nemorine



(c) nemorine

デービッド・コルブの「経験学習モデル」

成長を実感させる一つに方法に
褒めることがあると思いますが、

どうやって褒めていますか？

(c) nemorine

ゲームから褒め方を考える

- **ゲーマーのタイプの分類に「バートルテスト（Bartle Test）」があります。**

  これは、リチャード・バートル氏が提唱しているゲーマーの分類法です。
  下記の4種類から構成されます。

  ・達成者/アチーバー（Achiever）
  ・探検家/エクスプローラー（Explorer）
  ・社交家/ソーシャライザー（Socializer）
  ・殺人者/キラー（Killer）

➡ タイプによって褒め方を変えてみる

(c) nemorine

達成者/アチーバー（Achiever）

- 何かを達成することに喜びを見出すタイプ
- 数値や基準を軸に伝えた方が良い

- 例えば、あるツールを作った場合…
  - このツールはチーム全員が使っているよ。
  - このツールで作業効率が60%UPしたよ。

(c) nemorine

探検家/エクスプローラー（Explorer）

- 新しいことに挑戦したいタイプ
- 挑戦について言及したり、次の新しいステージを用意しておくのが良い

- 例えば、あるツールを作った場合…
  このツールには新しいライブラリ取り込んでいるね！いいね！
  次回のツールはGo言語で開発してみない？

(c) nemorine

社交家/ソーシャライザー（Socializer）

- 人とのコミュニケーションに重きをおくタイプ
- 人からの感謝などを伝えると良い

- 例えば、あるツールを作った場合…

  チームメンバー全員が楽になったよ。ありがとう。

  森さんがこのツールあって助かった～って言ってたよ。

(c) nemorine

殺人者/キラー（Killer）

- 勝ち負けなど競争に重きをおくタイプ
- 比較してあげると良い。
- ただし他のメンバーが嫌な気持ちにならないように。

- 例えば、あるツールを作った場合…

  今のメンバーの中でこの速さで作れる人は居ないね。

  ウメツより圧倒的にバグ少ないツールだよね。

(c) nemorine

まとめ

- 学習を手助けするのが「育成」であると考える。
- 全体を見える化し、興味を惹かせるためにRPG風スキルマップを使ってみると楽しく、スキルを見える化できる。
- 成長を実感させるためには褒めるという手段があるが、人のタイプによって褒め方を変えると効果が高い。

(c) nemorine

Have a nice testing!

(c) nemorine