

Gamification

booster l'engagement dans la dette technique et les tests

Xavier Blanc – IUF – Univ. Bordeaux - ProMyze

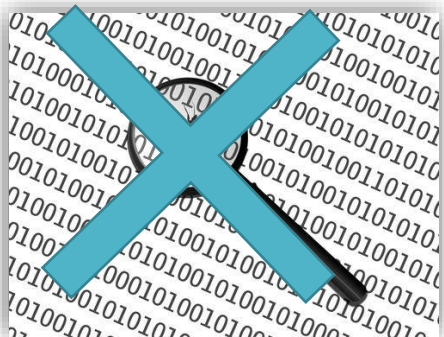
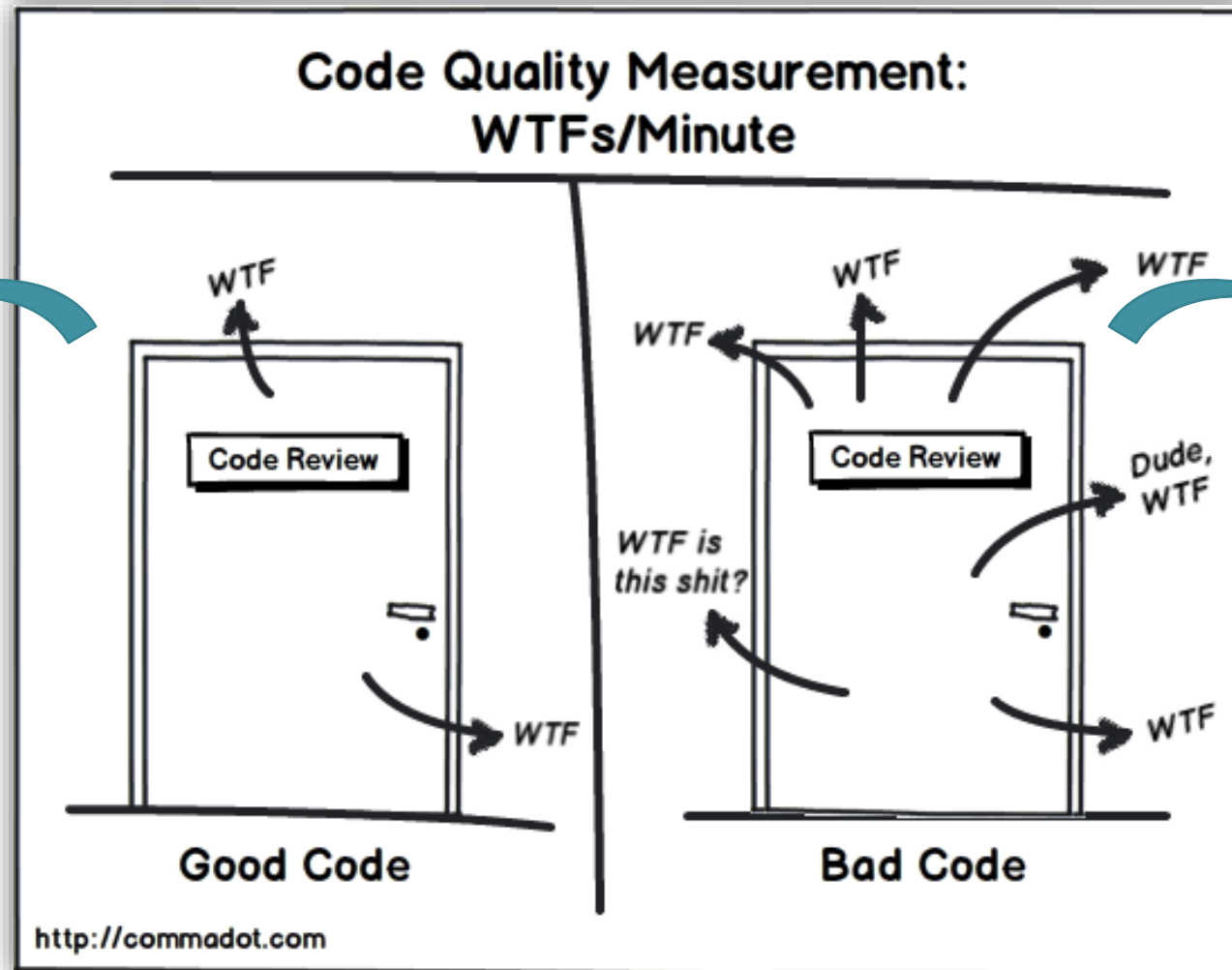


Pas (ou peu) de bug

Hypothèse

D'où vient l'introduction des bugs?

Bad Code => Plus de bug



Coder Bien

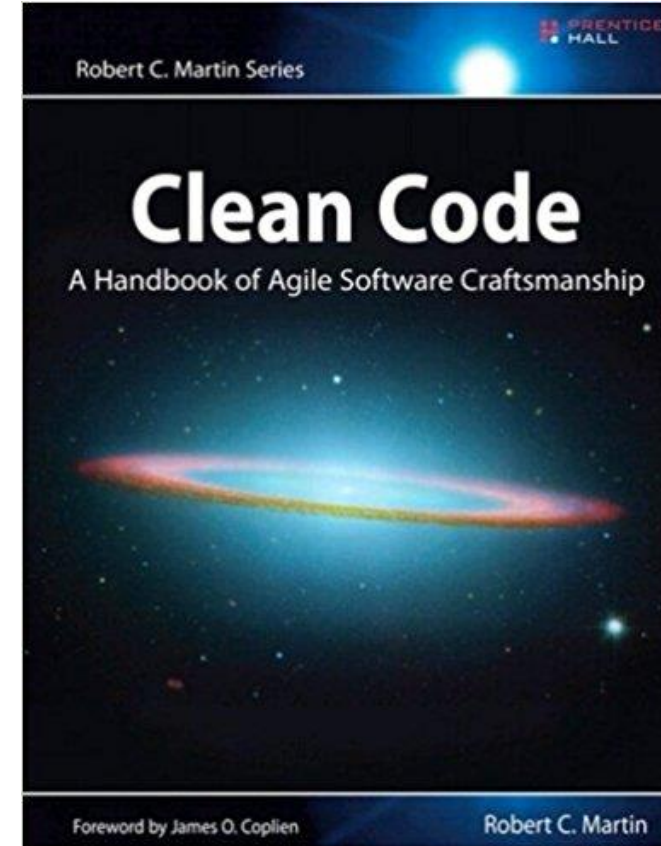
Bon ou mauvais code ?

Pour la plupart,

Un bon code est **lisible**

Un mauvais code est *difficile à lire*, à comprendre

La qualité d'une application est décuplée quand le code est lisible et compréhensible



Nommage des variables

Un nom = une intention

Le nom doit porter l'intention et rien d'autre

Prononçables

Les noms vont être communiqués lorsqu'on parlera du code

Cherchable

On va chercher les noms (ctrl+f),

```
46 public class ConnectionFactory implements Cloneable {
47
48     /** Default user name */
49     public static final String DEFAULT_USER = "guest";
50     /** Default password */
51     public static final String DEFAULT_PASS = "guest";
52     /** Default virtual host */
53     public static final String DEFAULT_VHOST = "/";
54     /** Default maximum channel number;
55      * zero for unlimited */
56     public static final int    DEFAULT_CHANNEL_MAX = 0;
57     /** Default maximum frame size;
58      * zero means no limit */
59     public static final int    DEFAULT_FRAME_MAX = 0;
60     /** Default heart-beat interval;
61      * 60 seconds */
62     public static final int    DEFAULT_HEARTBEAT = 60;
63     /** The default host */
```

```
92
93     private static final Version clientVersion =
94         new Version(AMQP.PROTOCOL.MAJOR, AMQP.PROTOCOL.MINOR);
95
96     /** The special channel 0 (<i>not</i> managed by the <code><b>_channelManager</b></code>) */
97     private final AMQChannel _channel0;
98
99     protected ConsumerWorkService _workService = null;
100
101     /** Frame source/sink */
102     private final FrameHandler _frameHandler;
103
104     /** Flag controlling the main driver loop's termination */
105     private volatile boolean _running = false;
106
107     /** Handler for (uncaught) exceptions that crop up in the {@link MainLoop}. */
108     private final ExceptionHandler _exceptionHandler;
109
110     /** Object used for blocking main application thread when doing all the necessary
```

```
    else {
        if (frame.channel == 0) { // the special channel
            _channel0.handleFrame(frame);
        } else {
```



Commenter ou Coder

Un commentaire n'améliore pas un mauvais code

On écrit souvent un commentaire sur un module mal codé.

Cela ne changera pas le problème.

Il faut recoder le module et le rendre plus clair.

Le code s'explique lui-même

Mettre des commentaires dans le code ne sert à rien, le code doit se lire et être intelligible

```

    * garbage collected when the connection object is no longer referenced.
    */
    public void start()
        throws IOException, TimeoutException {
        initializeConsumerWorkService();
        initializeHeartbeatSender();
        this._running = true;

        // Make sure that the first thing we do is to send the header,
        // which should cause any socket errors to show up for us, rather
        // than risking them pop out in the MainLoop
        AMQChannel.SimpleBlockingRpcContinuation connStartBlocker =
            new AMQChannel.SimpleBlockingRpcContinuation();
        // We enqueue an RPC continuation here without sending an RPC
        // request, since the protocol specifies that after sending
        // the version negotiation header, the client (connection
        // initiator) is to wait for a connection.start method to
        // arrive.
        _channel0.enqueueRpc(connStartBlocker);
        try {
            // The following two lines are akin to AMQChannel's
            // transmit() method for this pseudo-RPC.
            _frameHandler.setTimeout(handshakeTimeout);
            _frameHandler.sendHeader();
        } catch (IOException ioe) {
            _frameHandler.close();
            throw ioe;
        }

        this._frameHandler.initialize(this);
    }

```



```
402  
403 // We can now respond to errors having finished tailoring the connection  
404 this._inConnectionNegotiation = false;  
405 }  
406
```




```
/**  
 * Protected API - set the heartbeat timeout. Should only be called  
 * during tuning.  
 */  
public void setHeartbeat(int heartbeat) {  
    try {  
        _heartbeatSender.setHeartbeat(heartbeat);  
        _heartbeat = heartbeat;  
  
        // Divide by four to make the maximum unwanted delay in  
        // sending a timeout be less than a quarter of the  
        // timeout setting.  
        _frameHandler.setTimeout(heartbeat * 1000 / 4);  
    } catch (SocketException se) {  
        // should do more here?  
    }  
}
```

```

223  /**
224   * Convenience method for setting the fields in an AMQP URI: host,
225   * port, username, password and virtual host. If any part of the
226   * URI is omitted, the ConnectionFactory's corresponding variable
227   * is left unchanged.
228   * @param uri is the AMQP URI containing the data
229   */
230  public void setUri(Uri uri)
231      throws URISyntaxException, NoSuchAlgorithmException, KeyManagementException
232  {
233      if ("amqp".equals(uri.getScheme().toLowerCase())) {
234          // nothing special to do
235      } else if ("amqps".equals(uri.getScheme().toLowerCase())) {
236          setPort(DEFAULT_AMQP_OVER_SSL_PORT);
237          // SSL context factory not set yet, we use the default one
238          if (this.sslContextFactory == null) {
239              useSslProtocol();
240          }
241      } else {
242          throw new IllegalArgumentException("Wrong scheme in AMQP URI: " +
243              uri.getScheme());
244      }
245
246      String host = uri.getHost();
247      if (host != null) {
248          setHost(host);




```








 Rechercher 








Clean Code


Xavier Blanc - IUF - Université de Bordeaux - ProMyze

 **institut
universitaire
de France**

   0:17 / 1:18:16    

Clean Code - Java User Group - Bordeaux

453 vues  17  0  PARTAGER  

 **xavier blanc**
Ajoutée le 19 déc. 2017

MODIFIER LA VIDÉO

Validation de l'hypothèse

La qualité du code est-elle liée au nombre de bug ?

Défaut vs Bug

Un défaut rend le code moins lisible

Le code sera donc plus difficile à maintenir, à optimiser, à améliorer, à corriger, etc.

Un défaut n'est pas forcément la source d'un bug

Un bug dans le code est la cause d'une erreur dans l'application

Les conséquences d'un bug sont donc mesurables

La sévérité du bug est calculée en fonction de l'impact qu'il cause

=> plus il y a de défauts, plus il y a de bugs

Bad Smell : identifier les défauts

Les Bad Smells sont des indices pour identifier des défauts potentiels

Calculables automatiquement, les Bad Smells pointent les éléments du code à surveiller

L'objectif des Bad Smells est d'offrir une estimation quantitative du nombre de défauts

Attention, l'absence de Bad Smell ne veut pas dire qu'il n'y a pas de défaut



Taille

La taille du code est un très bon indicateur des défauts

Le code propre doit être petit, donc un grand code contient probablement des défauts

Les Bad Smell de taille

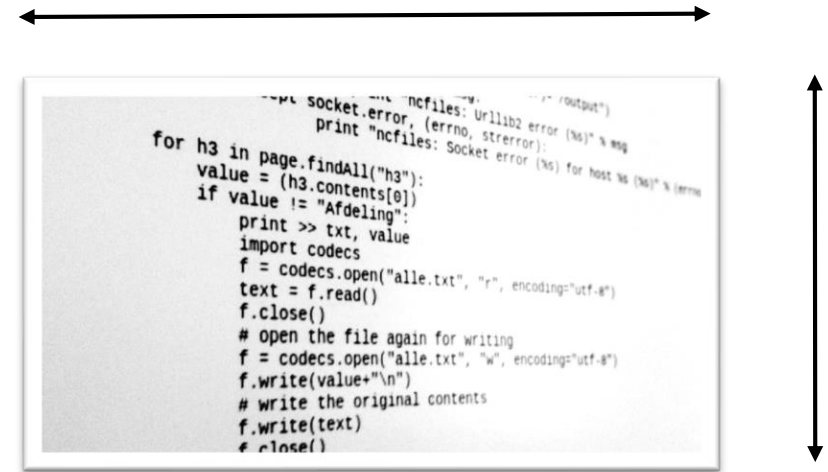
- LOC (Ligne de Code)

- Largeur des lignes

- Nombre de classes,

- Nombre de fonctions,

- etc.



Complexité

La complexité peut être mesurée en comptant le nombre de chemins dans le code

Plus un code est complexe moins il est lisible

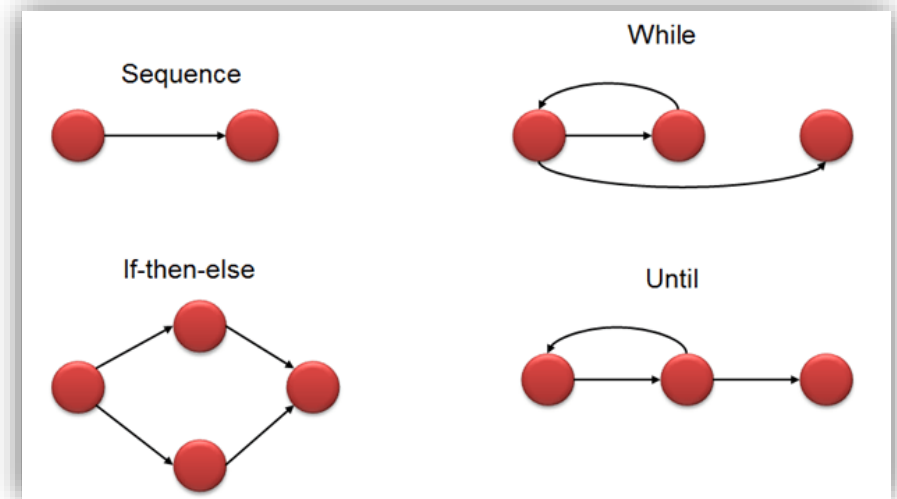
Les Bad Smell de complexité

- McCabe complexity

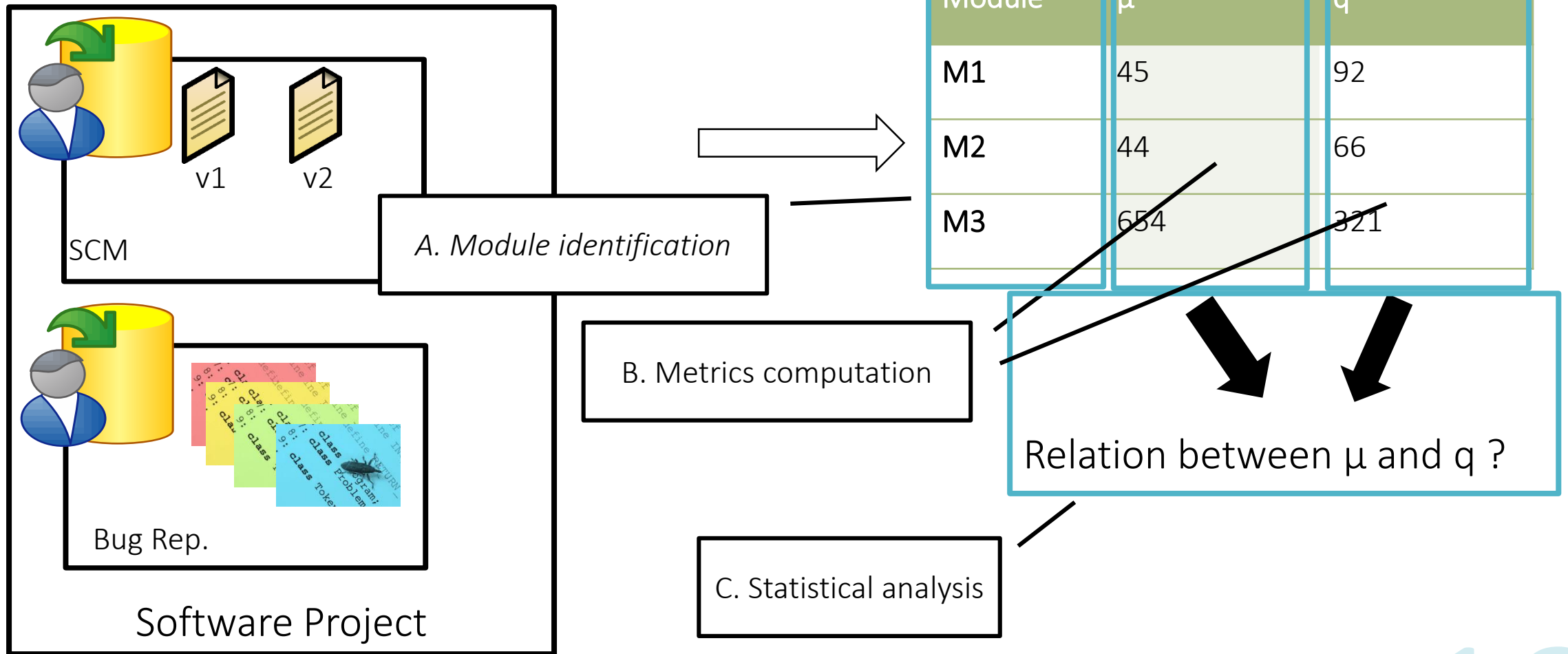
- Nombre de paramètres

- Nombre de switch case

- Nombre de return



Validation des linters

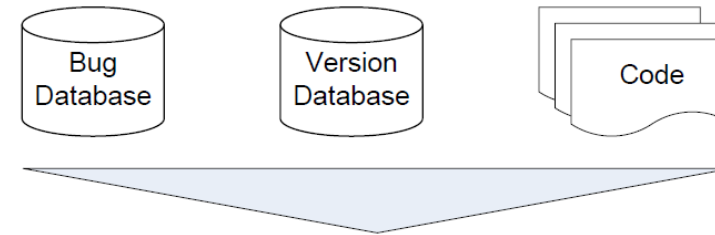


Nagappan Ball Zeller 06

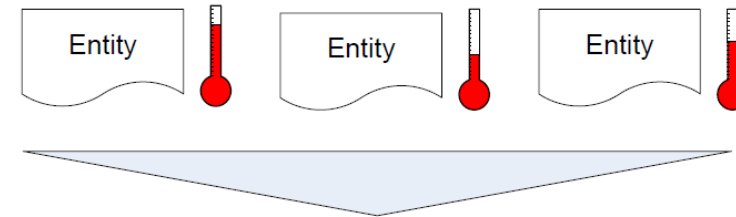
Corrélation Métriques / Bug pour identifier les composants critiques

L'objectif étant de focaliser la maintenance sur ces composants

1. Collect input data



2. Map post-release failures to defects in entities



3. Predict failure probability for new entities

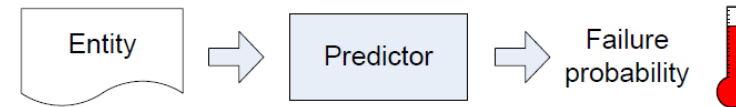


Figure 1. After mapping historical failures to entities, we can use their complexity metrics to predict failures of new entities.

Corrélation

Metric	Description		Correlation with post-release defects of M				
			A	B	C	D	E
Module metrics — correlation with metric in a module M							
<i>Classes</i>	# Classes in M		0.531	0.612	0.713	0.066	0.438
<i>Function</i>	# Functions in M		0.131	0.699	0.761	0.104	0.531
<i>GlobalVariables</i>	# global variables in M		0.023	0.664	0.695	0.108	0.460
Per-function metrics — correlation with maximum and sum of metric across all functions $f()$ in a module M							
<i>Lines</i>	# executable lines in $f()$	Max	-0.236	0.514	0.585	0.496	0.509
		Total	0.131	0.709	0.797	0.187	0.506
<i>Parameters</i>	# parameters in $f()$	Max	-0.344	0.372	0.547	0.015	0.346
		Total	0.116	0.689	0.790	0.152	0.478
<i>Arcs</i>	# arcs in $f()$'s control flow graph	Max	-0.209	0.376	0.587	0.527	0.444
		Total	0.127	0.679	0.803	0.158	0.484
<i>Blocks</i>	# basic blocks in $f()$'s control flow graph	Max	-0.245	0.347	0.585	0.546	0.462
		Total	0.128	0.707	0.787	0.158	0.472
<i>ReadCoupling</i>	# global variables read in $f()$	Max	-0.005	0.582	0.633	0.362	0.229
		Total	-0.172	0.676	0.756	0.277	0.445
<i>WriteCoupling</i>	# global variables written in $f()$	Max	0.043	0.618	0.392	0.011	0.450
		Total	-0.128	0.629	0.629	0.230	0.406
<i>AddrTakenCoupling</i>	# global variables whose address is taken in $f()$	Max	0.237	0.491	0.412	0.016	0.263
		Total	0.182	0.593	0.667	0.175	0.145
<i>ProcCoupling</i>	# functions that access a global variable written in $f()$	Max	-0.063	0.614	0.496	0.024	0.357
		Total	0.043	0.562	0.579	0.000	0.443
<i>FanIn</i>	# functions calling $f()$	Max	0.034	0.578	0.846	0.037	0.530
		Total	0.066	0.676	0.814	0.074	0.537
<i>FanOut</i>	# functions called by $f()$	Max	0.107	0.360	0.612	0.245	0.465

Dépend des métriques mais aussi du logiciel

Combiner les métriques

Table 5. Regression models and their explanative power

Project	Number of principal components	% cumulative variance explained	R^2	Adjusted R^2	F - test
A	9	95.33	0.741	0.612	5.731, $p < 0.001$
B	6	96.13	0.779	0.684	8.215, $p < 0.001$
C	7	95.34	0.579	0.416	3.541, $p < 0.005$
D	7	96.44	0.684	0.440	2.794, $p < 0.077$
E	5	96.33	0.919	0.882	24.823, $p < 0.0005$

Mais ...

Table 6. Predictive power of the regression models in random split experiments

Project	Correlation type	Random split 1	Random split 2	Random split 3	Random split 4	Random split 5
A	Pearson	0.480	0.327	0.725	-0.381	0.637
	Spearman	0.238	0.185	0.693	-0.602	0.422
B	Pearson	-0.173	0.410	0.181	0.939	0.227
	Spearman	-0.055	0.054	0.318	0.906	0.218
C	Pearson	0.559	-0.539	-0.190	0.495	-0.060
	Spearman	0.445	-0.165	0.050	0.190	0.082
D	Pearson	0.572	0.845	0.522	0.266	0.419
	Spearman	0.617	0.828	0.494	0.494	0.494
E	Pearson	-0.711	0.976	-0.818	0.418	0.007
	Spearman	-0.759	0.577	-0.883	0.120	0.152

Predictors are accurate only when obtained from the same or similar projects.

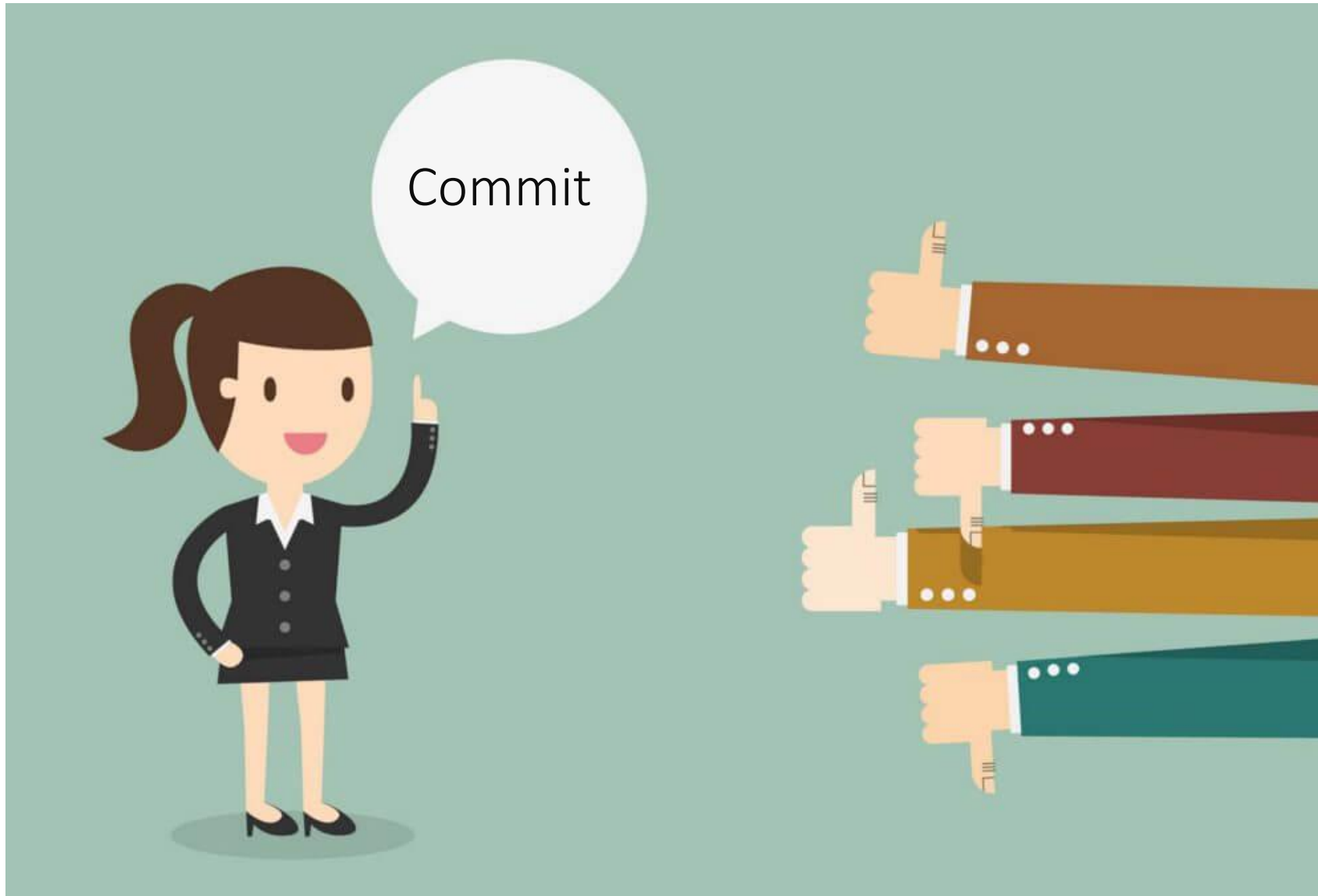
Approche



Constat => Manque d'engagement !



Partenariat Industriel CNRS – ESN



Concept 1 - Retour Personnalisé



Concept 2 - Gamification



Etude de Concepts

Focus Group

- Grande ESN
- Développeurs (env. 30)
- Plusieurs projets
- Plusieurs mois

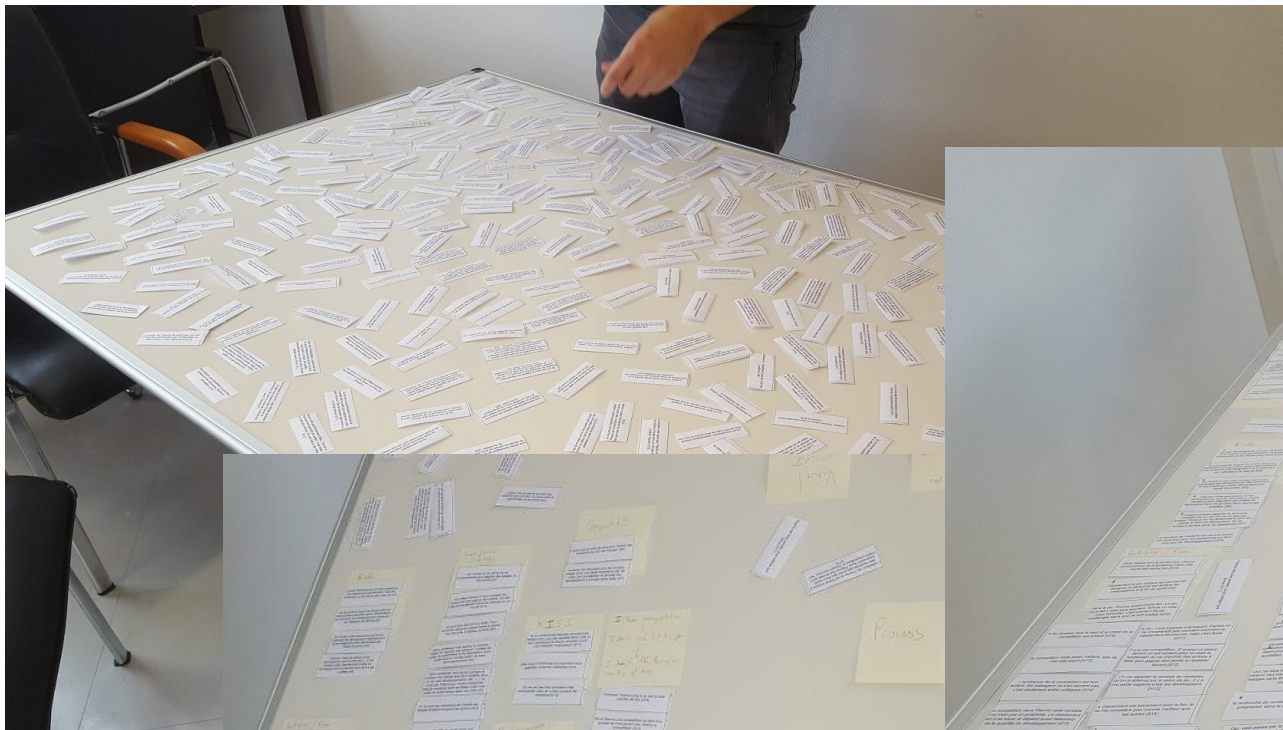
The screenshot displays the Themis web application interface. On the left, a user profile for 'Camille' shows a 'DET' (Debt) level of 10 with a progress bar at 450/660, and a 'COV' (Coverage) level of 9 with a progress bar at 252/550. The main content area shows two commit analysis results. The first commit, 'feat(app) : Files selection modal bug fixed, lazy loading added', shows a DET of +2, COV of +15.5% (57.1 to 72.6), and TFI of 18 new lines covered. The second commit, 'feat(server) : update core system after user feedback', shows a DET of +1, COV of - (0 to 0), and TFI of 23 new lines not covered. On the right, a 'Libra' section shows rules for functions, such as 'Règle: Functions should not be too complex' with a score of 0/15. The bottom of the interface includes a footer with 'Documentation - Web Service API' and 'Themis v1.1.3 - 2017 par ProMyze'.

1. Le Retour Personnalisé par analyse du commit (Dette technique, Couverture de Test, Test First)

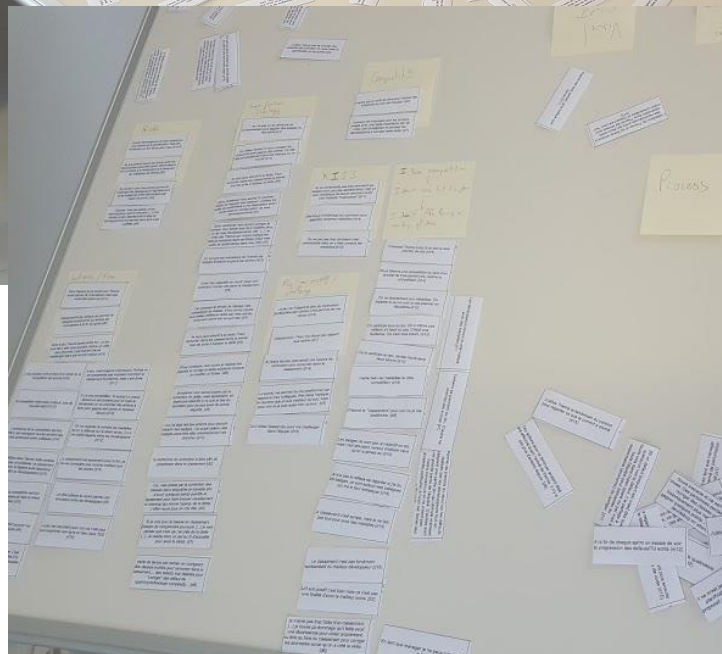
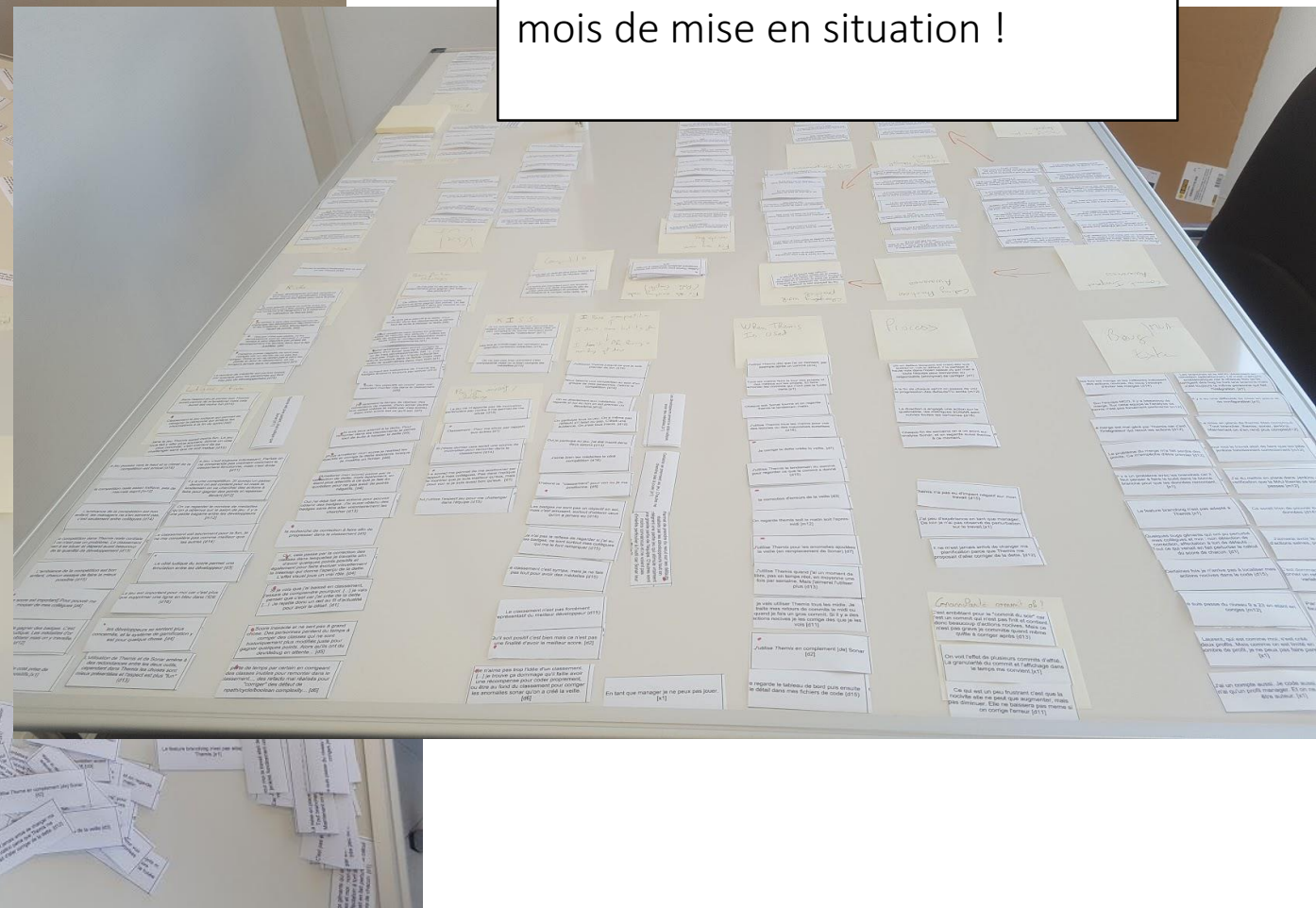
2. Gamification ouverte et positive. Plusieurs mécanismes de gamification (niveau, badge, etc.).

Un outil qui supporte les 2 concepts

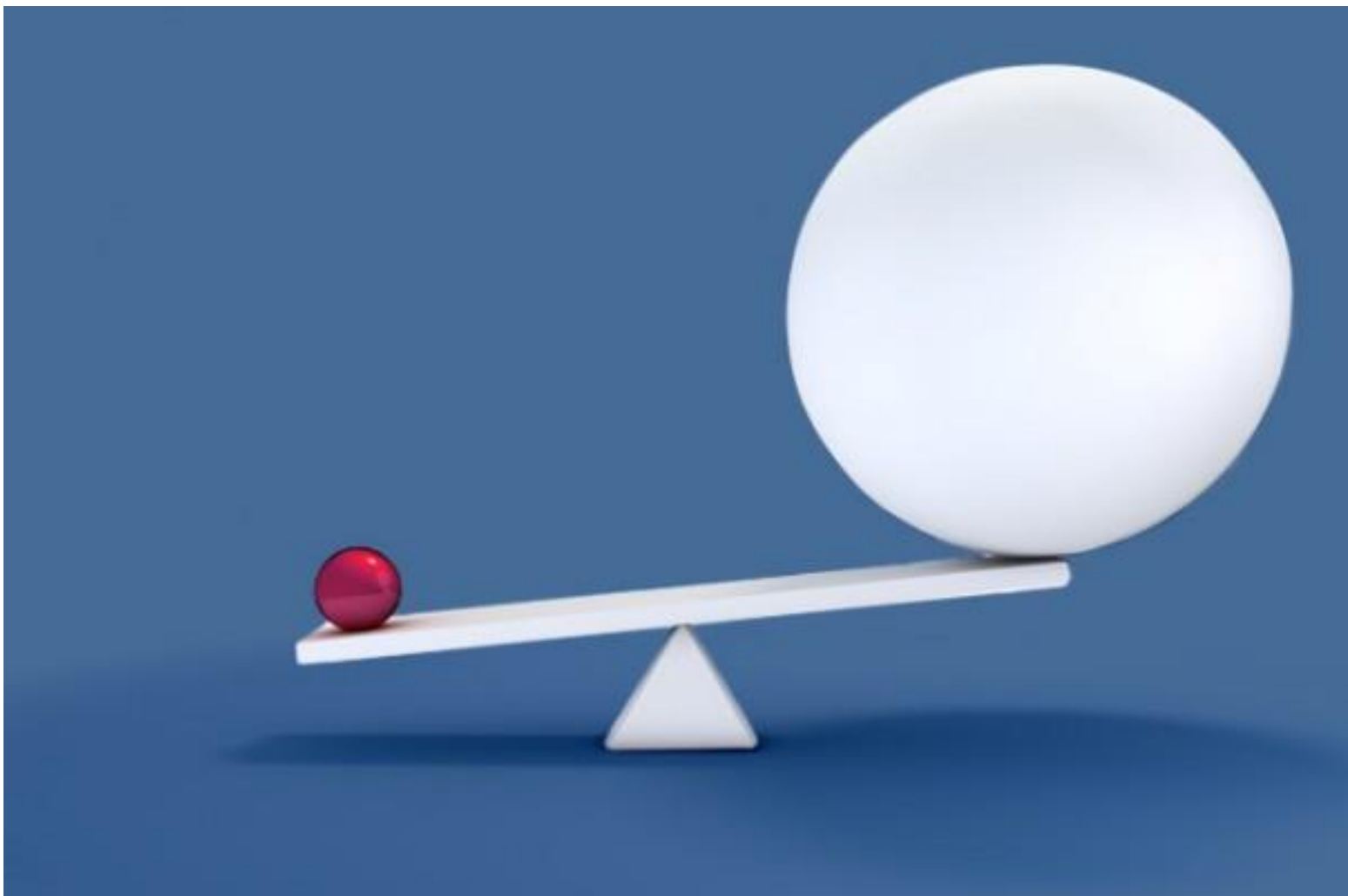
30



Réalisés auprès de développeurs
d'une grande ESN, après plusieurs
mois de mise en situation !



Analyse Qualitative : Interviews



Less is More



Prise de Conscience Vs. Connaissance

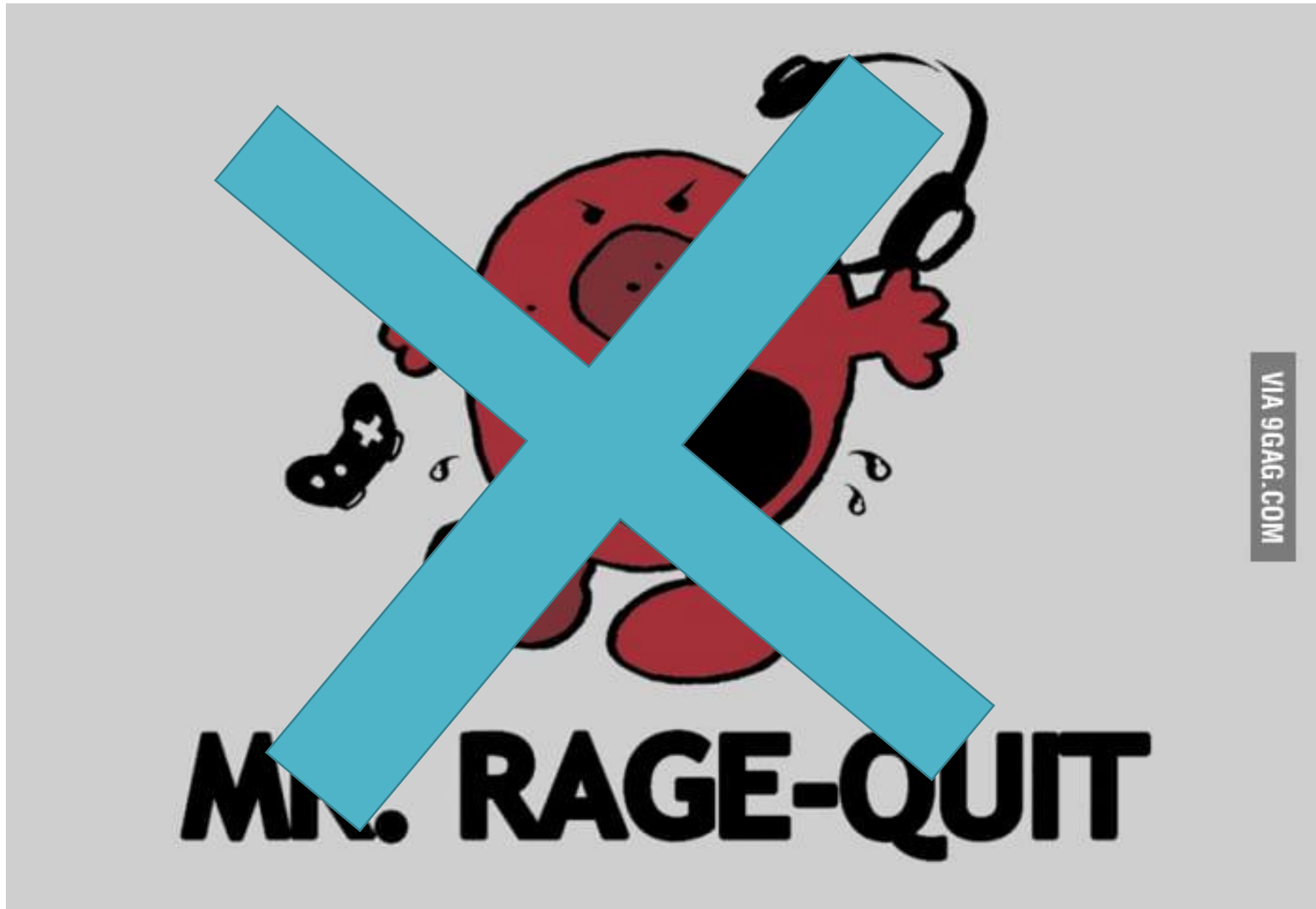




Compétition



Evaluation Personnelle



La Gamification n'est pas un jeu !

Conclusion

