K - THE UNIVERSAL PARAMETER FOR EFFICIENCY CONTROL

From Manufacturing Production to Information Systems

[Version 1.0 - September 2025 - Foundational Document for 3P3 System]

EXECUTIVE SUMMARY

The K coefficient is a revolutionary metric born in KOOL TOOL production that measures total systemic efficiency. The formula **K = Extracycles / (Performance × Presenteeism)** synthesizes into a single number the vitality of any organized system. K=1 represents theoretical perfection, while higher values indicate increasing inefficiencies to eliminate.

PART I: ORIGIN AND FOUNDATIONS OF THE K PARAMETER

1.1 Birth of K in Production

The K coefficient was born from the need to measure with **a single number** the overall efficiency of the KOOL TOOL assembly department. Before K, dozens of separate KPIs were needed that didn't provide a unified view.

1.2 The Three Pillars of Production K

1. EXTRACYCLES (EXT)

Definition: Everything that SHOULD NOT exist in the ideal process.

In production includes:

- Interruptions due to lack of materials
- Machine breakdowns
- Rework for errors
- Searching for tools/information
- Unplanned color changes
- Unscheduled breaks
- Hours of indirect workers assisting direct workers

Formula:

```
EXT = Extracycle_Hours / Direct_Production_Hours × 100
```

KOOL TOOL Example:

```
Direct production hours: 1100
Hours spent in extracycles: 180
EXT = 180/1100 × 100 = 16.36% = 1.164 (decimal form)
```

IMPORTANT: The theoretical minimum value of EXT is 1.0 (zero extracycles = perfection)

2. PERFORMANCE (R)

Definition: Ratio between assigned time (optimal) and time actually used.

Components:

- **TA (Assigned Time):** Optimal cycle time defined by time and methods analyst
- **TI (Time Used):** Real time minus extracycles

Formula:

```
R = (TA × Pieces_Produced) / (TI - Extracycles)
```

Value scale:

- R = 100% (1.00) = Theoretical optimal performance
- R = 97% (0.97) = Excellent
- R = 85% (0.85) = Good
- R = 75% (0.75) = Minimum survival
- R < 75% = Critical

Example:

```
Optimal cycle time for cover: 1 minute
Pieces produced: 400
Time used (net): 420 minutes
R = (1 × 400) / 420 = 0.952 (95.2%)
```

3. PRESENTEEISM (P)

Definition: Actual presence versus planned presence.

Formula:

```
P = (100 - Absenteeism%) / 100
```

Where Absenteeism:

```
A = (Absent_Workers / Total_Workers) × 100
```

Example:

```
Total workers: 330
Absent workers: 20
A = 20/330 × 100 = 6.1%
P = (100 - 6.1) / 100 = 0.939
```

1.3 THE COMPLETE K FORMULA

```
K = EXTRACYCLES / (PERFORMANCE × PRESENTEEISM)
K = EXT / (R × P)
```

Complete KOOL TOOL calculation example:

```
EXT = 1.164 (16.4% extracycles)
R = 0.97 (97% performance)
P = 0.939 (93.9% presenteeism)

K = 1.164 / (0.97 × 0.939) = 1.278
```

1.4 K Interpretation

K Value	Meaning	Required Action
K = 1.0	Theoretical perfection	Impossible ideal target
1.0 < K ≤ 1.2	Excellent system	Maintain and refine
1.2 < K ≤ 1.5	Good system	Room for improvement
1.5 < K ≤ 2.0	Inefficient system	Corrective interventions needed
2.0 < K ≤ 3.0	Critical system	Urgent reorganization
K > 3.0	Operational emergency	Immediate drastic intervention

PART II: ADVANCED APPLICATIONS OF PRODUCTION K

2.1 Standard Base and Production Capacity

Revolutionary concept: Create standard products with same cycle time (TC) to calculate capacity.

Required Force Formula:

```
FN = (Nr_Standard_Base × TC_Base) / (480 × K)
```

Example:

```
Daily standard base: 2500 pieces
Base cycle time: 50 seconds
Shift minutes: 480
Current K: 1.278

FN = (2500 × 50/60) / (480 × 1.278) = 333 workers needed
```

2.2 K as Performance Predictor

K allows us to:

- **Predict** real production capacity
- **Identify** hidden bottlenecks
- **Quantify** improvement impact
- **Simulate** what-if scenarios

Simulation example:

```
If we reduce extracycles by 20%:
New_EXT = 1.164 × 0.8 = 0.931
New_K = 0.931 / (0.97 × 0.939) = 1.022
Efficiency improvement: 20%
Workers saved: 333 - 290 = 43
```

PART III: K EVOLUTION IN INFORMATION SYSTEMS

3.1 The Challenge: From Physical to Digital

Problems to solve:

1. How to measure "extracycles" in digital processes?
2. What does "performance" mean for a phone call?
3. How to calculate "presenteeism" for remote work?

3.2 Production → Information System Mapping

Production	Information System	Metric
Assembled pieces	Completed processes	Entities/day
Machine interruptions	System downtime	Minutes offline
Material searches	Information searches	Clicks/queries
Rework	Data corrections	Updates after save
Tool changeover	Context switching	App/tab changes
Unscheduled breaks	Idle time	Inactivity >5min

3.3 K for Digital Processes - Redefinition

DIGITAL EXTRACYCLES

Digital_EXT = All actions that don't add value:
- Searching for missing information
- Navigation between disconnected systems
- Data re-entry (duplications)
- Waiting for approvals
- Error corrections
- Non-productive meetings

Practical measurement:

EXT = (Non_Value_Time / Total_Time) + 1

Where 1 represents the theoretical minimum (no extracycles)

DIGITAL PERFORMANCE

Digital_R = Expected_Output / Actual_Output

Where:
- Expected_Output = Nr processes × standard time
- Actual_Output = Time really used

PHO (phone calls) example:

Expected calls: 20 × 5 min = 100 minutes
Actual time: 120 minutes
R = 100/120 = 0.833 (83.3%)

DIGITAL PRESENTEEISM

```
Digital_P = Actual_Availability / Planned_Availability

Includes:
- Physical/remote presence
- System availability
- Focus (not excessive multitasking)
```

3.4 Complete Digital K Formula

```
Digital_K = Digital_EXT / (Digital_R × Digital_P)
```

Complete PHO calculation example:

```
Phone call PHO25001:
- Standard time: 5 minutes
- Actual time: 8 minutes
- Interruptions: 2 (customer info search, slow system)
- Corrections: 1 (name typo)

EXT = (3 extra minutes / 8 minutes) + 1 = 1.375
R = 5/8 = 0.625
P = 1.0 (operator present)

K = 1.375 / (0.625 × 1.0) = 2.2

Interpretation: Inefficient process, requires optimization
```

PART IV: K IN THE 3P3 SYSTEM

4.1 K as Ontological Attribute

In the 3P3 paradigm, K becomes an **intelligent attribute** of every entity:

```
javascript
```

```
PHO_ENTITY = {
  "DNA_code": "PHO25001",
  "k_coefficient": {
    "value": 1.35,
    "components": {
      "extracycles": 1.15,
      "performance": 0.92,
      "presenteeism": 0.95
    },
    "trend": "improving",
    "target": 1.2,
    "suggestions": ["reduce_search_time", "improve_data_quality"]
  }
}
```

4.2 K in the 10 Ontological Domains

K integrates into attribute domains:

1. **IDENTITY:** K identifies process efficiency
2. **TEMPORAL:** K evolves over time (historical)
3. **AUTHORIZATION:** Only managers see detailed K
4. **COMMUNICATION:** K communicates to dashboards/APIs
5. **TRIGGER:** K > 2 activates automatic alerts
6. **DOCUMENTAL:** K reports generated automatically
7. **MATERIAL:** K calculated real-time (performance)
8. **PERFORMANCE:** K itself is THE metric
9. **SECURITY:** Anomalous K signals problems
10. **EVOLUTION:** ML optimizes K over time

4.3 Real-Time K Dashboard

```
╔═══════════════════════════════════════════════════════════════════════════╗
║ K COEFFICIENT DASHBOARD - KOOL TOOL SYSTEM                                ║
╠═══════════════════════════════════════════════════════════════════════════╣
║                                                                           ║
║ PHYSICAL PRODUCTION          │ DIGITAL SYSTEM                             ║
║ ──────────────────           │ ──────────────                             ║
║ Assembly: K=1.28 ✓           │ PHO Calls: K=1.35 ✓                        ║
║ Gluing: K=1.45 ⚠             │ RCH Requests: K=1.52 ⚠                     ║
║ Packaging: K=1.12 ✓          │ BOM Creation: K=2.1 ⚠                      ║
║                              │                                            ║
```

```
╠ WEEKLY TREND          | SUGGESTED OPTIMIZATIONS  ╣
╠ ──────────────        | ──────────────────────   ╣
╠ ▼ -8% improvement     | • Reduce context switching ╣
╠ Q4 Target: K<1.3      | • Improve data quality   ╣
╠                       | • Automate approvals     ╣
╚═══════════════════════════════════════════════════╝
```

PART V: TECHNICAL IMPLEMENTATION FOR PROGRAMMERS

5.1 Measurement Architecture

javascript

```
class KCoefficientCalculator {
  constructor() {
    this.measurements = [];
    this.baseline = null;
  }

  // Capture metrics for process
  captureMetrics(process) {
    const metrics = {
      timestamp: Date.now(),
      process_type: process.type,

      // Extracycles
      searches: process.searchCount,
      corrections: process.correctionCount,
      waiting_time: process.waitingMs,
      context_switches: process.appSwitches,

      // Performance
      standard_time: process.standardTimeMs,
      actual_time: process.actualTimeMs,

      // Presenteeism
      scheduled_time: process.scheduledTimeMs,
      available_time: process.availableTimeMs,
      focus_score: process.focusScore // 0-1
    };

    return this.calculateK(metrics);
  }

  calculateK(metrics) {
    // Extracycles (minimum 1.0)
    const nonValueTime = metrics.waiting_time +
                         (metrics.searches * 30000) +
                         (metrics.corrections * 60000);
    const EXT = Math.max(1.0, 1 + (nonValueTime / metrics.actual_time));

    // Performance (0-1)
    const R = Math.min(1.0, metrics.standard_time / metrics.actual_time);

    // Presenteeism (0-1)
    const P = (metrics.available_time / metrics.scheduled_time) *
              metrics.focus_score;

    // Final K
```

```
    const K = EXT / (R * P);

    return {
      k_value: K.toFixed(3),
      extracycles: EXT.toFixed(3),
      performance: R.toFixed(3),
      presenteeism: P.toFixed(3),
      status: this.getKStatus(K),
      suggestions: this.generateSuggestions(metrics, K)
    };
  }

  getKStatus(k) {
    if (k <= 1.2) return 'EXCELLENT';
    if (k <= 1.5) return 'GOOD';
    if (k <= 2.0) return 'WARNING';
    if (k <= 3.0) return 'CRITICAL';
    return 'EMERGENCY';
  }

  generateSuggestions(metrics, k) {
    const suggestions = [];

    if (metrics.searches > 3) {
      suggestions.push('Improve data accessibility');
    }
    if (metrics.corrections > 1) {
      suggestions.push('Enhance validation rules');
    }
    if (metrics.context_switches > 5) {
      suggestions.push('Reduce app fragmentation');
    }
    if (metrics.waiting_time > 60000) {
      suggestions.push('Automate approval workflow');
    }

    return suggestions;
  }
}
```

5.2 Integration in CMP-ETY-LOG

IN CMP (Template):

```
json
```

```
{
  "template_id": "PHO_TEMPLATE_V1",
  "k_configuration": {
    "standard_time_seconds": 300,
    "extracycle_weights": {
      "search": 30,
      "correction": 60,
      "wait": 1
    },
    "k_thresholds": {
      "excellent": 1.2,
      "warning": 1.5,
      "critical": 2.0
    }
  }
}
```

IN ETY (Runtime):

json

```
{
  "entity_id": "PHO25001",
  "k_tracking": {
    "start_time": "2025-09-17T09:15:00Z",
    "end_time": "2025-09-17T09:23:00Z",
    "interruptions": 2,
    "corrections": 1,
    "k_calculated": 1.35
  }
}
```

IN LOG (History):

json

```
{
  "timestamp": "2025-09-17T09:23:00Z",
  "event": "K_CALCULATED",
  "entity": "PHO25001",
  "k_value": 1.35,
  "components": {
    "ext": 1.15,
    "r": 0.92,
    "p": 0.95
  },
  "alert": false
}
```


5.3 SQL Queries for K Analysis

sql

```sql
-- Average K by process type
SELECT
  process_type,
  AVG(k_value) as avg_k,
  MIN(k_value) as best_k,
  MAX(k_value) as worst_k,
  COUNT(*) as total_processes
FROM process_metrics
WHERE date >= DATE_SUB(NOW(), INTERVAL 30 DAY)
GROUP BY process_type
ORDER BY avg_k ASC;

-- Weekly K trend
SELECT
  WEEK(date) as week_num,
  AVG(k_value) as weekly_k,
  AVG(extracycles) as avg_ext,
  AVG(performance) as avg_perf,
  AVG(presenteeism) as avg_pres
FROM process_metrics
GROUP BY WEEK(date)
ORDER BY week_num DESC
LIMIT 12;

-- Critical processes (K > 2)
SELECT
  entity_id,
  process_type,
  k_value,
  JSON_EXTRACT(suggestions, '$') as improvement_areas,
  created_at
FROM process_metrics
WHERE k_value > 2.0
  AND date = CURDATE()
ORDER BY k_value DESC;
```

PART VI: PRACTICAL USE CASES

6.1 PHO Case - Phone Call Management

CURRENT SCENARIO:

- 20 calls/day
- Average time: 8 minutes (standard: 5)

- 3-4 interruptions per call
- Fragmented system (5 different apps)

CURRENT K CALCULATION:

```
EXT = 1.4 (40% time in extracycles)
R = 5/8 = 0.625
P = 0.9 (10% time in meetings)
K = 1.4 / (0.625 × 0.9) = 2.49
```

POST-3P3 IMPLEMENTATION:

```
EXT = 1.1 (10% residual extracycles)
R = 5/5.5 = 0.91
P = 0.95
K = 1.1 / (0.91 × 0.95) = 1.27
```

IMPROVEMENT: 49% efficiency recovered

6.2 BOM Case - Bill of Materials Creation

CURRENT SCENARIO:

- 5 BOMs/day
- Average time: 45 minutes (standard: 20)
- Continuous searches for codes/prices
- Frequent errors requiring rework

K CALCULATION:

```
EXT = 1.8 (80% extracycles)
R = 20/45 = 0.44
P = 0.85
K = 1.8 / (0.44 × 0.85) = 4.81 (CRITICAL!)
```

POST-3P3 TARGET:

```
Target_K = 1.5 (reducible with automation)
```

PART VII: FUTURE EVOLUTION OF K

7.1 Predictive K with Machine Learning

python

```python
# ML for K prediction
from sklearn.ensemble import RandomForestRegressor

class KPredictor:
    def __init__(self):
        self.model = RandomForestRegressor()

    def train(self, historical_data):
        features = ['hour_of_day', 'day_of_week',
                    'process_type', 'user_experience',
                    'system_load', 'pending_tasks']
        X = historical_data[features]
        y = historical_data['k_value']
        self.model.fit(X, y)

    def predict_k(self, context):
        return self.model.predict([context])[0]

    def suggest_optimal_time(self, process_type):
        # Find moment with minimum predicted K
        predictions = []
        for hour in range(8, 18):
            context = [hour, datetime.now().weekday(),
                       process_type, ...]
            k_pred = self.predict_k(context)
            predictions.append((hour, k_pred))
        return min(predictions, key=lambda x: x[1])
```

7.2 K as Smart Contract

solidity

```
contract KCoefficient {
    mapping(address => uint) public kValues;
    uint constant K_THRESHOLD = 1500; // K = 1.5

    function updateK(address entity, uint k) public {
        kValues[entity] = k;

        if (k > K_THRESHOLD) {
            emit AlertHighK(entity, k);
            triggerOptimization(entity);
        }
    }

    function rewardLowK(address entity) public {
        if (kValues[entity] < 1200) { // K < 1.2
            // Reward efficiency
            transferBonus(entity, calculateBonus(kValues[entity]));
        }
    }
}
```

CONCLUSIONS

The K parameter represents a **revolution in efficiency measurement**. Born in production, it naturally evolves into digital while maintaining its essence: **one number that tells everything**.

Advantages of K:

1. **Simplicity:** One single number instead of dozens of KPIs
2. **Universality:** Applicable to any process
3. **Objectivity:** Mathematical formula, not opinion-based
4. **Actionability:** Clearly indicates where to intervene
5. **Evolvability:** Adapts to new contexts

K in the 3P3 future:

- Every entity will have its K
- ML will optimize K in real-time
- Smart contracts will reward low K
- Unified production-digital dashboards
- K will become THE universal metric

"K shows us how good we are at producing value while eliminating waste"

KOOL TOOL SRL - Romania

Where efficiency becomes measurable