

# Informática Industrial

## Guidelines for Practical Implementation of the Factory Automation System

Mário J. Sousa, Luís Almeida, Pedro M. Santos  
2019/2020 - 2º semester

# Outline

1. Technology Risk Assessment
2. Development Process
3. Team Management
4. Version Control Systems
5. Object-Oriented Programming
6. Next steps

# 1. Technology Risk Assessment

- For each technological component in your system:
  1. Identify potential technologies to use;
  2. Select a subset (one or more); build **Minimum Working Example (MWE)** for each
- Rank options regarding:
  - Ease of development
  - Suitability to the requirements of your system architecture
  - Ease of integration with other technological components (i.e., are there libraries to connect to the other components; are these solutions new or well-established/tested)
  - Ease of maintenance when system is in operation (**Product Lifecycle Management**)
  - Other criteria you might find relevant
- Keep this effort constrained in time (apr. 1 week, if that much)

## 2. Development Process

*Hint: Target a modular and bottom-up development approach*

1. Identify the minimum set of implementation to have a working system (the core of your system)
  1. Assume one single type of order
  2. Set up one operating cell in the shopfloor
  3. Set up connection MES - PLC (through OPC-UA)
  4. Set up connection to database to obtain shopfloor status
  5. Set up very basic path planning (or even a static path)
2. Then build on top of that & add non-core functionalities
  1. Interpretation of ERP XMLs
  2. Permanent/event-driven update of database with PLC status
  3. Etc...

Takeaway: build a minimally working system, then improve it modularly

# 3. Team Management

- **Task planning & effort allocation: who does what?**
  - Divide work in tasks; identify precedences and opportunities for parallel work
  - Assign tasks to developers and assign reasonable timeframe (*suggestion: double your initial estimate*)
- **Project Management Tools**
  - Tools: MS Project, Redmine,...
  - **Task & Issues definition:** Redmine defines not only “Tasks”, but also “Features” and “Bugs”
  - **Reporting & Accountability:**
    - “Finished this task that was assigned to me”
    - “Found a bug, who’s responsible for it or in charge of addressing it?”
  - **Specification & User Manual:** how is the system actually built, and how to use it?
    - Through Wiki, documents...
- **Work Methodology & Time management**
  - **Work flow:** Sprints? Weekly deliveries? Ad hoc one-to-one meetings, or weekly meetings?
  - **Leadership/monitoring:** someone in charge of checking everything is going as planned? If indecision, who makes the final call?
  - **Gantt chart:** I suggest you do one to keep track of your evolution

# Snapshot from Redmine

<https://redmine.fe.up.pt/>

**Redmine**

Visão geral Download Actividade Planificação **Tarefas** Notícias Wiki Forums Repositório

**Tarefas**

▼ Filtros

Estado aberto ▼ Adicionar filtro

► Opções

Aplicar  Limpar

#	Tipo	Estado	Assunto	Alterado	Categoria
<input type="checkbox"/> 33156	Defect	New	allow mobilebrowser zoom	2020-03-16 11:40	UI
<input type="checkbox"/> 33153	Feature	New	UI feature to quickly change issue status	2020-03-16 13:11	UI
<input type="checkbox"/> 33151	Feature	New	Provide status for issue children via REST API	2020-03-15 15:09	REST API
<input type="checkbox"/> 33148	Defect	New	application stuck if query with filter "Issue" and with large amount of IssueIDs	2020-03-13 17:44	Issues filter
<input type="checkbox"/> 33140	Defect	Confirmed	Gantt bar is not displayed if the due date is the first day of the month	2020-03-16 07:07	Gantt
<input type="checkbox"/> 33139	Defect	New	Redmine.pm not working with socks	2020-03-13 04:51	SCM extra
<input type="checkbox"/> 33138	Defect	New	Apache crashes with Redmine.pm and RedmineCacheCredsMax active	2020-03-13 10:32	SCM extra
<input type="checkbox"/> 33129	Feature	New	Restore "Latest Projects" to Home page	2020-03-11 15:20	UI
<input type="checkbox"/> 33127	Defect	New	Assignee icon is misaligned when print stylesheet is applied	2020-03-11 10:43	Issues
<input type="checkbox"/> 33126	Feature	New	Add UserCustomField value to csv of users list	2020-03-15 06:21	Accounts / authentication
<input type="checkbox"/> 33121	Defect	New	IssueQuery not usable from plugins	2020-03-10 15:00	Plugin API
<input type="checkbox"/> 33118	Patch	New	Detect plain diffs in e-mail submitted issues and map to attachments	2020-03-10 11:21	
<input type="checkbox"/> 33117	Patch	New	Redirect to users_path instead of edit_user_path in order to avoid confusion	2020-03-10 22:11	Accounts / authentication

New issue

Tracker \* Bug ▼

Subject \*

Description **B I U S C H1 H2 H3**

Status \* New ▼

Priority \* Normal ▼

Assignee

Parent task

Start date 16/03/2020

Due date dd/mm/aaaa

Estimated time  Hours

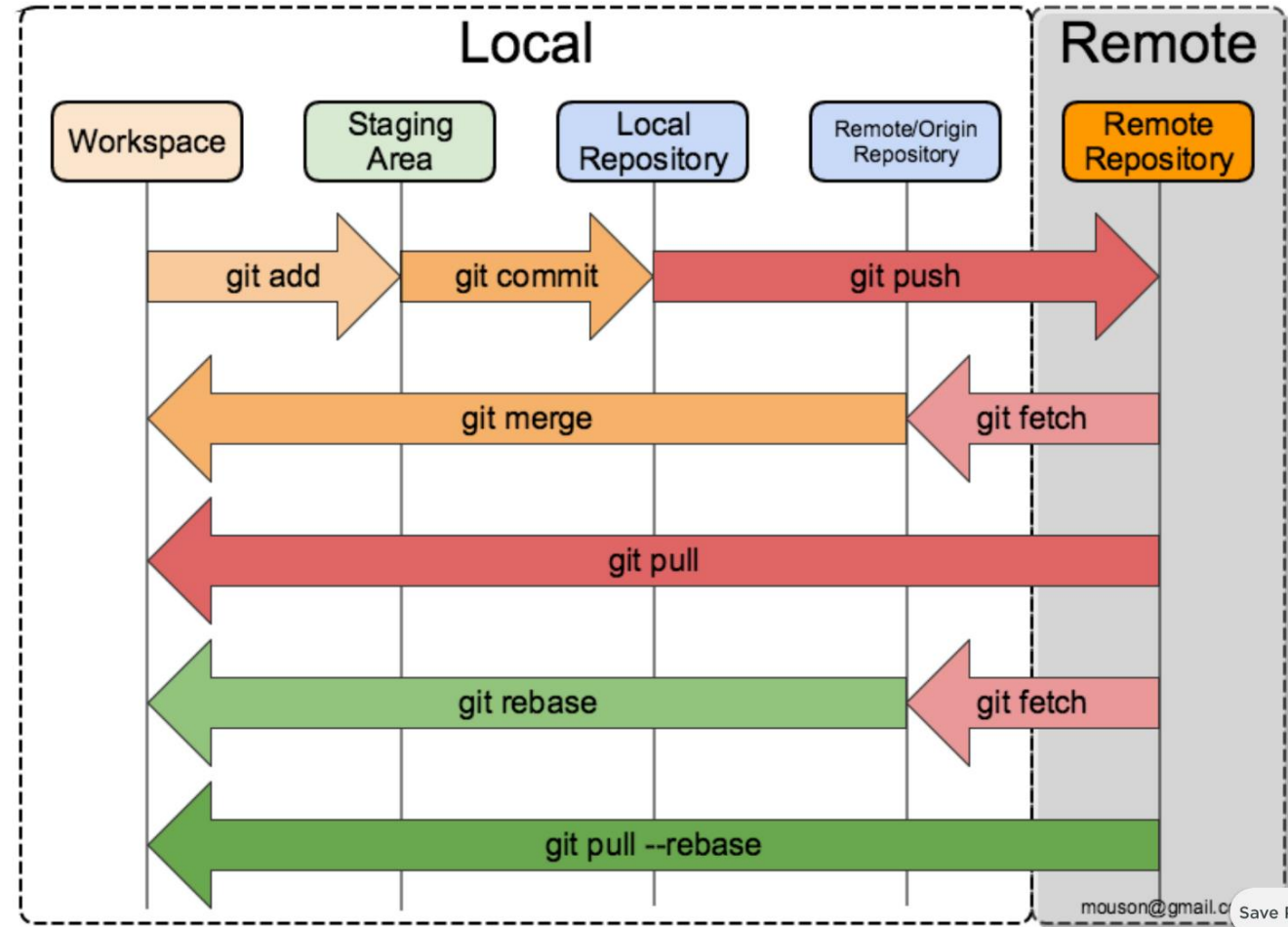
% Done 0% ▼

Files  Nenhum ficheiro selecionado (Maximum size: 15 MB)

Watchers  Ana Aguiar  Pedro Salgueiro Santos  Luis Miguel Sousa

# 4. Version Control Systems

- Git is a widely-used version control system
- Other well-known system is SVN (centralized as opposed to decentralized, as Git is)
- Essential to keep track of the evolution of a large software project (no GDrive or Dropbox here 😊)
- Some features
  - History of project evolution is recorded; developers can **ROLL-BACK** to previous version when bug is found
  - Tools for merging disparate versions are available
  - All *Commits* must be commented



Taken from: Jianping Zeng, <https://medium.com/@zjpjack/reverting-modified-in-4-stages-in-git-f3997f526902>

# 5. Object-Oriented Programming

conveyor.h / .cpp

cell.h / .cpp

factory.cpp (main)

```

1  /*
2  * conveyor.h
3  *
4  * Created on: 16/03/2020
5  * Author: pedro
6  */
7
8  #ifndef CONVEYOR_H_
9  #define CONVEYOR_H_
10
11 class conveyor {
12 public:
13     conveyor();
14     virtual ~conveyor();
15     bool moveMinus();
16     bool movePlus();
17     bool stop();
18
19 private:
20     // -1: moving minus;
21     // 0: stopped;
22     // 1: moving plus
23     int motor=0;
24 };
25 #endif /* CONVEYOR_H_ */
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
  
```

```

1  /*
2  * conveyor.cpp
3  *
4  * Created on: 16/03/2020
5  * Author: pedro
6  */
7
8  #include "conveyor.h"
9
10 conveyor::conveyor() {
11     // TODO Auto-generated constructor stub
12     motor = 0;
13 }
14
15 conveyor::~conveyor() {
16     // TODO Auto-generated destructor stub
17     motor = 0;
18 }
19
20 bool conveyor::movePlus() {
21     motor = 1;
22     return true;
23 }
24
25 bool conveyor::moveMinus() {
26     motor = -1;
27     return true;
28 }
29
30 bool conveyor::stop() {
31     motor = 0;
32     return true;
33 }
34
  
```

```

1  /*
2  * cell.h
3  *
4  * Created on: 16/03/2020
5  * Author: pedro
6  */
7
8  #include "conveyor.h"
9
10 #ifndef CELL_H_
11 #define CELL_H_
12
13 class cell {
14 public:
15     cell();
16     virtual ~cell();
17     bool movePlus();
18     bool moveMinus();
19     bool stop();
20
21 private:
22     // Declaração dos conveyors
23     conveyor conv[3];
24 };
25 #endif /* CELL_H_ */
26
  
```

```

1  /*
2  * cell.cpp
3  *
4  * Created on: 16/03/2020
5  * Author: pedro
6  */
7
8  #include "cell.h"
9
10 cell::cell() {
11     // TODO Auto-generated constructor stub
12
13     // Inicialização dos conveyors
14     for (int i=0; i<3; i++)
15         conv[i] = conveyor();
16 }
17
18 cell::~cell() {
19     // TODO Auto-generated destructor stub
20 }
21
22 bool cell::movePlus() {
23     for (int i=0; i<3; i++)
24         conv[i].movePlus();
25
26     return true;
27 }
28
29 bool cell::moveMinus() {
30     for (int i=0; i<3; i++)
31         conv[i].moveMinus();
32
33     return true;
34 }
35
36 bool cell::stop() {
37     for (int i=0; i<3; i++)
38         conv[i].stop();
39
40     return true;
41 }
42
  
```

```

1  //====
2  // Name      : factory2.cpp
3  // Author    : Pedro Santos
4  // Version   :
5  // Copyright : Your copyright notice
6  // Description: Hello World in C++, Ansi-style
7  //=====
8
9  #include <iostream>
10 using namespace std;
11
12 #include "conveyor.h"
13 #include "cell.h"
14
15 int main() {
16     cout << "!!!Hello World!!!" << endl;
17
18     cell cell1 = cell();
19     cell1.movePlus();
20
21     return 0;
22 }
23
  
```

Object instantiation



## 6. Next Steps

- Define tasks
  - Assign developers
  - Draw timeline
- 
1. Set up project management & VCS tools
  2. Create MWE of potential technologies
  3. Start development