DC4Cities
An environmentally sustainable data centre for Smart Cities

Project № 609304

D3.3
Description of the final technical requirements and architecture design

Responsible: HP
Contributors: Freemind, CN, UNI PASSAU, ENEA, UMA, INRIA
Document Reference: D3.3 - Description of the final technical requirements and architecture design
Level: PU
Version: 1.0
Date: 20.03.2015
## CONTRIBUTORS TABLE

<table>
<thead>
<tr>
<th>DOCUMENT SECTION</th>
<th>AUTHOR(S)</th>
<th>REVIEWER(S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I Introduction</td>
<td>ROBERTO CHIAPPINI (HP)</td>
<td>DAVIDE DE CHIARA (ENEA) GIOVANNI GIULIANI (HP)</td>
</tr>
<tr>
<td>II Actors and Use Cases</td>
<td>TORBEN MÖLLER (UMA)</td>
<td>DAVIDE DE CHIARA (ENEA) GIOVANNI GIULIANI (HP)</td>
</tr>
<tr>
<td>III.1. High level view</td>
<td>ROBERTO CHIAPPINI (HP)</td>
<td>FABIEN HERMENIER (INRIA) GIOVANNI GIULIANI (HP)</td>
</tr>
<tr>
<td>III.2. D4C Control System</td>
<td>ROBERTO CHIAPPINI (HP)</td>
<td>FABIEN HERMENIER (INRIA) GIOVANNI GIULIANI (HP)</td>
</tr>
<tr>
<td>III.3. Northbound Subsystem High level architecture</td>
<td>AMMAR ALYOUSEF (UNI PASSAU) FLORIAN NIEDERMEIER (UNI PASSAU)</td>
<td>FABIEN HERMENIER (INRIA) GIOVANNI GIULIANI (HP)</td>
</tr>
<tr>
<td>III.4 Southbound Subsystem High level architecture</td>
<td>CORENTIN DUPONT (CN)</td>
<td>FABIEN HERMENIER (INRIA) GIOVANNI GIULIANI (HP)</td>
</tr>
<tr>
<td>III.5. Federation</td>
<td>GIOVANNI GIULIANI (HP) ROBERTO CHIAPPINI (HP)</td>
<td>FABIEN HERMENIER (INRIA) GIOVANNI GIULIANI (HP)</td>
</tr>
<tr>
<td>IV.1. D4C Control System Single Site Architecture Specification</td>
<td>ANDREA MILANI (HP)</td>
<td>CORENTIN DUPONT (CN) ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.1.1. DC Process Controller Module</td>
<td>ANDREA MILANI (HP)</td>
<td>CORENTIN DUPONT (CN) ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.1.2. DC Power Planner Module</td>
<td>AMMAR ALYOUSEF (UNI PASSAU) FLORIAN NIEDERMEIER (UNI PASSAU)</td>
<td>CORENTIN DUPONT (CN) ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.1.3. DC Power Plan Splitter Module</td>
<td>ANDREA MILANI (HP)</td>
<td>CORENTIN DUPONT (CN) ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.1.4. EASC Handler Module</td>
<td>ANDREA MILANI (HP)</td>
<td>CORENTIN DUPONT (CN) ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.1.5. DC Power Plan Option Consolidator</td>
<td>FABIEN HERMENIER (INRIA)</td>
<td>CORENTIN DUPONT (CN) ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.1.6. Power Plan Escalation Manager Module</td>
<td>DAVIDE DE CHIARA (ENEA) GIOVANNI PONTI (ENEA)</td>
<td>CORENTIN DUPONT (CN) ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.1.7. Configuration Controller</td>
<td>THOMAS SCHULZE (UMA)</td>
<td>MEHDI SHEIKHALISHAHI (CN) ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.1.8. Historical Database</td>
<td>FREDERIC WAUTERS (FM)</td>
<td>MEHDI SHEIKHALISHAHI (CN) ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>DOCUMENT SECTION</td>
<td>AUTHOR(S)</td>
<td>REVIEWER(S)</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>----------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>IV.1.9. Connectors</td>
<td>FREDERIC WAUTERS (FM)</td>
<td>MEHDI SHEIKHALISHAHI (CN)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.1.10. Service Gateway</td>
<td>FREDERIC WAUTERS (FM)</td>
<td>MEHDI SHEIKHALISHAHI (CN)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.1.11. Energis</td>
<td>FREDERIC WAUTERS (FM)</td>
<td>MEHDI SHEIKHALISHAHI (CN)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.1.12. Graph Viewer</td>
<td>FREDERIC WAUTERS (FM)</td>
<td>MEHDI SHEIKHALISHAHI (CN)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.1.13. Dashboard</td>
<td>FREDERIC WAUTERS (FM)</td>
<td>MEHDI SHEIKHALISHAHI (CN)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2. D4C Control System Federation Architecture Specification</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.1. DC Process Controller Module</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.2. DC Power Planner Module</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.3. DC Power Plan Splitter Module</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.4. EASC Handler Module</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.5. DC Power Plan Option Consolidator Module</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.6. Power Plan Escalation Manager Module</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.7. Configuration Controller</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.8. Historical Database</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.9. Connectors</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.10. Service Gateway</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.11. Energis</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.12. Graph Viewer</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>IV.2.13. Dashboard</td>
<td>GIOVANNI GIULIANI (HP)</td>
<td>FREDERIC WAUTERS (FM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>DOCUMENT SECTION</td>
<td>AUTHOR(S)</td>
<td>REVIEWER(S)</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>------------------------------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>V.1. Renewable Energy Forecasting Interface</td>
<td>AMMAR ALOYousef (UNI PASSAU)</td>
<td>THOMAS SCHULZE (UMA)</td>
</tr>
<tr>
<td></td>
<td>FLORIAN NIEDERMEIER (UNI PASSAU)</td>
<td>TORBEN MÖLLER (UMA)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>V.2. Energy Adaptive Software Control Interface</td>
<td>MEHDI SHEIKHALISHAHI (CN)</td>
<td>THOMAS SCHULZE (UMA)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TORBEN MÖLLER (UMA)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>VI Development &amp; testing</td>
<td>FABIEN HERMENIER (INRIA)</td>
<td>FLORIAN NIEDERMEIER (UNI PASSAU)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HERMANN DE MEER (UNI PASSAU)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>VII Appendix A: Low Level Use Cases</td>
<td>TORBEN MÖLLER (UMA)</td>
<td>AMMAR ALOYousef (UNI PASSAU)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HERMANN DE MEER (UNI PASSAU)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
<tr>
<td>VIII Appendix B: Metrics Catalogue</td>
<td>FREDERIC WAUTERS (FM)</td>
<td>AMMAR ALOYousef (UNI PASSAU)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HERMANN DE MEER (UNI PASSAU)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROBERTO CHIAPPINI (HP)</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS

I. Introduction ........................................................................................................................................ 9
   I.1. Acronyms ...................................................................................................................................... 9

II. Actors and Use Cases ........................................................................................................................ 12
   II.1. Actors ......................................................................................................................................... 12
   II.2. Use Cases ................................................................................................................................... 13

III. High level Architecture .................................................................................................................... 14
   III.1. High level view .......................................................................................................................... 14
   III.2. D4C Control System .................................................................................................................. 15
      III.2.1. Power Planning Loop .......................................................................................................... 16
      III.2.2. EASC Control Loop ........................................................................................................... 17
      III.2.3. EASC Monitoring Loop ...................................................................................................... 20
      III.2.4. D4C Control System complementary modules .................................................................. 20
   III.3. Northbound Subsystem High level architecture ........................................................................ 20
   III.4. Southbound Subsystem High level architecture ....................................................................... 21
   III.5. Federation .................................................................................................................................. 22

IV. Functional Specifications .................................................................................................................... 27
   IV.1. D4C Control System Single Site Architecture Specification ................................................... 27
      IV.1.1. DC Process Controller Module ............................................................................................. 28
      IV.1.2. DC Power Planner Module .................................................................................................. 30
      IV.1.3. DC Power Plan Splitter Module ............................................................................................ 31
      IV.1.4. EASC Handler Module .......................................................................................................... 33
      IV.1.5. DC Power Plan Option Consolidator Module ....................................................................... 35
      IV.1.6. Power Plan Escalation Manager Module ............................................................................... 37
      IV.1.6.a. Evaluation .......................................................................................................................... 37
      IV.1.7. Configuration Controller ....................................................................................................... 39
      IV.1.8. Historical Database ............................................................................................................... 40
      IV.1.8.a. Metric values ...................................................................................................................... 40
      IV.1.9. Connectors ............................................................................................................................. 41
      IV.1.10. Service Gateway .................................................................................................................. 42
      IV.1.11. Energis ................................................................................................................................. 42
      IV.1.11.a. Estimate Formula Generator ............................................................................................ 42
      IV.1.11.b. Estimate Formula Executor ............................................................................................... 44
      IV.1.11.c. Prediction/Forecasting purpose ........................................................................................ 45
      IV.1.11.d. What-if/Estimation purpose ............................................................................................... 46
      IV.1.12. Graph Viewer ...................................................................................................................... 46
      IV.1.13. Dashboard ............................................................................................................................. 47
IV.2. D4C Control System Federation Architecture Specification ......................51
   IV.2.1. DC Process Controller Module .............................................................51
   IV.2.2. DC Power Planner Module ................................................................51
   IV.2.3. DC Power Plan Splitter Module ...........................................................51
   IV.2.4. EASC Handler Module .......................................................................52
   IV.2.5. DC Power Plan Option Consolidator Module ......................................53
   IV.2.6. Power Plan Escalation Manager Module .............................................54
   IV.2.7. Configuration Controller ....................................................................54
   IV.2.8. Historical Database ............................................................................54
   IV.2.9. Connectors ..........................................................................................54
   IV.2.10. Service Gateway ...............................................................................54
   IV.2.11. Energis ..............................................................................................55
   IV.2.12. Graph Viewer ....................................................................................55
   IV.2.13. Dashboard .........................................................................................55
V.  Interface Specifications ..................................................................................56
   V.1. Renewable Energy Forecasting Interface ................................................56
      V.1.1. Forecast API .....................................................................................56
      V.1.2. DC Controller Power Plan API ..........................................................57
   V.2. Energy Adaptive Software Control Interface .........................................59
      V.2.1. EASC OptionPlan API ....................................................................59
      V.2.2. EASC ActivityPlan API ..................................................................61
VI. Development & testing ...............................................................................62
   VI.1. Development ..........................................................................................62
   VI.2. Testing ....................................................................................................62
      VI.2.1. Issue management in the codebase ...................................................63
      VI.2.2. Testing with a simulator ..................................................................63
   VI.3. Demo Support ........................................................................................65
VII. Appendix A: Low Level Use Cases .........................................................66
    VII.1. Configuration ......................................................................................66
    VII.2. Ideal Power Plan ................................................................................68
    VII.3. Power Budgets ....................................................................................70
    VII.4. Escalation ............................................................................................71
VIII. Appendix B: Metrics Catalogue ...............................................................73
TABLE OF FIGURES

Figure 1: DC4Cities Actors ................................................................. 12
Figure 2 High level architecture ......................................................... 14
Figure 3: D4C Control System Architecture ........................................ 15
Figure 4: D4C Control System control loops ....................................... 16
Figure 5: Power Planning Loop .......................................................... 16
Figure 6: EASC Control Loop – defining DC power plan quota ............ 17
Figure 7: EASC Control Loop – defining global data centre plan .......... 18
Figure 8: EASC Control Loop – exception management ...................... 19
Figure 9: EASC Control Loop – consolidated power plan communication . 19
Figure 10: Northbound Subsystem High level architecture .................. 21
Figure 11: Southbound Subsystem High level architecture .................... 21
Figure 12: Smart City Assisted / On Demand Federation ....................... 23
Figure 13: Service relocation ............................................................. 24
Figure 14: DC4Cities Continuous Federation ....................................... 24
Figure 15: Continuous federation with 2 levels consolidator ................ 25
Figure 16: D4C Control System Component diagram ........................... 27
Figure 17: DC Process Controller Module ........................................... 28
Figure 18: DC Power Planner Module ............................................... 30
Figure 19: DC Power Plan Splitter Module ......................................... 32
Figure 20: Per-EASC vs group splitting ............................................. 32
Figure 21: Option Plan Collections ................................................... 33
Figure 22: Activity plan execution ..................................................... 34
Figure 23: EASC Handler Module internal architecture ......................... 35
Figure 24: DC Power Plan Option Consolidator Module ....................... 36
Figure 25: set of EASC Activity Plans of 24 hours and their works ......... 38
Figure 26: Evaluation output based on the concept of traffic light ......... 38
Figure 27: High-level analysis based on time intervals (short, medium and long-term) 39
Figure 28: Configuration Controller Architecture ............................. 40
Figure 29: Model identification – input ............................................. 43
Figure 30: Model identification – Output ............................................ 43
Figure 31: Model Usage ................................................................. 44
Figure 32: Energy Power Dashboard – site selection ............................ 47
Figure 33: Energy Power Dashboard – past period .............................. 48
Figure 34: Energy Power Dashboard – future period ........................... 50
Figure 35: Power Splitter Modules ................................................... 52
Figure 36: Consolidating different service categories .......................... 53
Figure 37: Federated Option Plan Consolidator .................................. 53
Figure 38: Development methodology ............................................. 62
# Table of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Actors</td>
<td>13</td>
</tr>
<tr>
<td>Table 2</td>
<td>Use Cases</td>
<td>13</td>
</tr>
<tr>
<td>Table 3</td>
<td>Sample power demands for 2 EASCs having each 2 options</td>
<td>36</td>
</tr>
<tr>
<td>Table 4</td>
<td>Overall costs for the data centre provider depending on the selected options</td>
<td>37</td>
</tr>
<tr>
<td>Table 5</td>
<td>Forecast API - URL parameters</td>
<td>56</td>
</tr>
<tr>
<td>Table 6</td>
<td>Forecast API - Input parameters</td>
<td>57</td>
</tr>
<tr>
<td>Table 7</td>
<td>Forecast API - Output parameters</td>
<td>57</td>
</tr>
<tr>
<td>Table 8</td>
<td>DC Controller Power Plan API - URL parameters</td>
<td>58</td>
</tr>
<tr>
<td>Table 9</td>
<td>DC Controller Power Plan API - Output parameters</td>
<td>58</td>
</tr>
<tr>
<td>Table 10</td>
<td>EASC OptionPlan API - Input parameters</td>
<td>59</td>
</tr>
<tr>
<td>Table 11</td>
<td>EASC OptionPlan API - Output parameters</td>
<td>60</td>
</tr>
<tr>
<td>Table 12</td>
<td>EASC Activity Plan API – Activity Plan type</td>
<td>61</td>
</tr>
<tr>
<td>Table 13</td>
<td>Solving duration for the optimizer when replaying trial experiments</td>
<td>64</td>
</tr>
<tr>
<td>Table 14</td>
<td>Solving duration for the optimizer when replaying trial experiments</td>
<td>65</td>
</tr>
<tr>
<td>Table 15</td>
<td>Solving duration for the optimizer against the number of EASCs</td>
<td>65</td>
</tr>
<tr>
<td>Table 16</td>
<td>Metric Catalog</td>
<td>74</td>
</tr>
</tbody>
</table>
I. INTRODUCTION

This deliverable is the output of the analysis work done in DC4Cities WP3 (Architecture and Integration of DC4Cities System, Metrics Support and Benchmarks).

The deliverable starts by describing in the first chapter the actors interacting with the DC4Cities system.

After that, in the second chapter the DC4Cities Reference Architecture is presented with references to the needs of the different actors and to the power/energy objectives set to the data centres by the Smart City.

In chapter IV the functionalities of the different modules of the DC4Cities Control System are presented together with the technical directions which has been used for the development of these modules.

In chapter V the main interfaces, which have been used for the system-to-system communication, are presented.

Finally we provide more details on the development and testing tools which have been used in the DC4Cities project in chapter VI.

I.1. Acronyms

In this table are listed acronyms used in this deliverable.

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>AS</td>
<td>Application Server</td>
</tr>
<tr>
<td>biz perf</td>
<td>Business Performance</td>
</tr>
<tr>
<td>CI</td>
<td>Continuous Integration</td>
</tr>
<tr>
<td>CO2</td>
<td>carbon dioxide</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
</tr>
<tr>
<td>CRUD</td>
<td>Create, read, update and delete</td>
</tr>
<tr>
<td>CTRL</td>
<td>DC4Cities Control System</td>
</tr>
<tr>
<td>CVRMSE</td>
<td>Coefficient of Variation of the Root Mean Square Error</td>
</tr>
<tr>
<td>D4C</td>
<td>DC4Cities</td>
</tr>
<tr>
<td>D4C ADM</td>
<td>DC4Cities system administrator</td>
</tr>
<tr>
<td>DB</td>
<td>Data Base</td>
</tr>
<tr>
<td>DC</td>
<td>Data Centre</td>
</tr>
<tr>
<td>DC4EA</td>
<td>DC4Cities Energy Administrator</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>DC4ES</td>
<td>DC4Cities Energy Subsystem</td>
</tr>
<tr>
<td>DCA</td>
<td>DC Adapt</td>
</tr>
<tr>
<td>DCBM</td>
<td>Data Centre Business Manager</td>
</tr>
<tr>
<td>DCIM</td>
<td>Data Centre Infrastructure Manager</td>
</tr>
<tr>
<td>EASC</td>
<td>Energy Adaptive Software Controller</td>
</tr>
<tr>
<td>EASW</td>
<td>Energy Adaptive Software</td>
</tr>
<tr>
<td>EMA</td>
<td>Energy Management Authority</td>
</tr>
<tr>
<td>EMA-SC</td>
<td>Smart City Energy Management Authority</td>
</tr>
<tr>
<td>EMIS</td>
<td>Energy Management Information System</td>
</tr>
<tr>
<td>ERDS</td>
<td>Energy-Related Data Supplier</td>
</tr>
<tr>
<td>GreenSDA</td>
<td>Green Service Delivery Agreement</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HTML</td>
<td>Hypertext Markup Language</td>
</tr>
<tr>
<td>IAAS</td>
<td>Infrastructure as a Service</td>
</tr>
<tr>
<td>ICT</td>
<td>Information and Communications Technology</td>
</tr>
<tr>
<td>IPP</td>
<td>Ideal Power Plan</td>
</tr>
<tr>
<td>ISO</td>
<td>International Organization for Standardization</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>KPI</td>
<td>Key Performance Indicator</td>
</tr>
<tr>
<td>kW</td>
<td>Kilowatt</td>
</tr>
<tr>
<td>kWh</td>
<td>kilowatt hour</td>
</tr>
<tr>
<td>MBE</td>
<td>Mean Bias Error</td>
</tr>
<tr>
<td>POJO</td>
<td>Plain Old Java Object</td>
</tr>
<tr>
<td>PUE</td>
<td>Power Usage Efficiency</td>
</tr>
<tr>
<td>PV</td>
<td>Photovoltaic</td>
</tr>
<tr>
<td>R²</td>
<td>coefficient of determination</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>RE</td>
<td>Renewable Energy</td>
</tr>
<tr>
<td>Ren</td>
<td>Renewable</td>
</tr>
<tr>
<td>Ren Pct</td>
<td>Renewable Percentage</td>
</tr>
<tr>
<td>REST</td>
<td>Representational State Transfer</td>
</tr>
<tr>
<td>SCM</td>
<td>Source Code Manager</td>
</tr>
<tr>
<td>SLA</td>
<td>Service Level Agreement</td>
</tr>
<tr>
<td>SMA</td>
<td>Service Management Authority</td>
</tr>
<tr>
<td>SMA-SC</td>
<td>Smart City Service Management Authority</td>
</tr>
<tr>
<td>SW</td>
<td>Software</td>
</tr>
<tr>
<td>UC</td>
<td>Use Case</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>UTC</td>
<td>Coordinated Universal Time</td>
</tr>
<tr>
<td>VM</td>
<td>Virtual Machine</td>
</tr>
<tr>
<td>W</td>
<td>Watt</td>
</tr>
<tr>
<td>WAR</td>
<td>Web application Archive</td>
</tr>
<tr>
<td>WM</td>
<td>Working Mode</td>
</tr>
<tr>
<td>WP</td>
<td>Work Package</td>
</tr>
<tr>
<td>WS</td>
<td>Web Server</td>
</tr>
</tbody>
</table>
II. ACTORS AND USE CASES

II.1. Actors

The DC4Cities Control System offers services to a set of actors and it does that through several interactions. It is therefore important to have the actors part of the DC4Cities ecosystem defined in a clear way and to understand their roles and their expectations. It is also necessary to show how they interact with the system. The following use case diagram (see Figure 1) shows the actors interacting with the DC4Cities Control System, which is composed of three subsystems. The Energy Related Data Supplier and the Energy Management Authority use the DC4Cities Energy Subsystem to monitor and predict the availability of energy, evaluate DC power budgets and set objectives. The Energy Management Authority, Data Centre Business Manager and DC4Cities Energy Admin use the DC4Cities Control System to set and check the configuration and gather information about escalations. The DC4Cities system computes the ideal data centre power plan and compute and distributes the power budgets, without external interaction. The Data Centre Technical Manager interacts with the Energy Aware Software Controller to determine the working mode of the applications.

The following table describes the actors interacting with the DC4Cities Control System. An actor can be a human or an external system. For more detail about the actors and information about the remaining actors, we refer to WP2 deliverable D2.3 (to be published).

<table>
<thead>
<tr>
<th>Actor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Energy Management Authority (EMA)</td>
<td>The (Smart City) Energy Management Authority issues power/energy objectives for each controlled DC and monitors</td>
</tr>
</tbody>
</table>
II.2. Use Cases

The use case diagram in Figure 1 shows the high level use cases of the DC4Cities system. High level use cases describe the interactions on a more abstract level with only few details. The high level use cases are explained in detail in D2.3 (to be published). The following table shows summarizations of the use cases 2, 3, 4 and 8, which are the most relevant use cases for this work package. These use cases are concretized in this deliverable and are described in a very detailed way, called low level use cases. You can find the low level use cases of WP3 in the chapter VII.

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UC2: Set/Check Configuration</td>
<td>The data centre is configured to become part of the DC4Cities system</td>
</tr>
<tr>
<td>UC3: Compute Ideal Data Centre Power Plan</td>
<td>For a configured timeframe the ideal power plan is computed, based on the energy forecast.</td>
</tr>
<tr>
<td>UC4: Compute and Distribute Power Budgets</td>
<td>The data centre computes and distributes power budgets for all EASCs.</td>
</tr>
<tr>
<td>UC8: Escalation</td>
<td>An objective cannot be accomplished, therefore a responsible person get informed about this problem in order to take appropriate action.</td>
</tr>
</tbody>
</table>
III. High level Architecture

In this chapter we describe the high level architecture of DC4Cities; the first section will focus on the external components interfacing with the DC4Cities (D4C) control system that will then be described briefly in section III.2. and in detail in the chapter IV.

III.1. High level view

DC4Cities goal is to tune the data centre software execution load in such a way, that the data centre power consumption matches the renewable energy source availability in compliance to energy/power goals, set by an Energy Management Authority – for instance in the context of a Smart City.

DC4Cities therefore needs to know the energy availability and source mix forecasts to determine its reference consumption levels (called max/ideal power) to meet the energy/power goals. This information is retrieved and continuously updated from the Energy/Power Forecast Data providers. Until now there is no reference standard for the format and the technology to retrieve this information, DC4Cities uses “Connectors” (see Figure 2) to decouple the peculiarities of the different data providers with respect to the unified “Renewable Energy Forecasting Interface” (also called the “Northbound Interface”).

DC4Cities can also collect both environmental data (such as detailed weather info) and locally produced power data. These historical data allow a “Local Forecaster” to provide short to medium term forecasts on the availability of the renewable energy. For example, in the case of a photovoltaic panels directly providing power to the data centre, the “Local Forecaster” is able to compute the relevant power production forecasts.

Figure 2 High level architecture
In the Figure 2, the blue arrows represent the data flow from the external data sources feeding information to the D4C Control System\(^1\).

DC4Cities approach might become a standardization candidate of the Renewable Energy Forecasting Interface, so that data providers might in the future be able to directly connect to the control system, as represented by the yellow line (top centre of the diagram).

The D4C Control System (described in section III.2.), first computes the max/ideal power targets for the next time window (approx. 24 hours) for a data centre. It then gathers information about the software running inside the data centre and provides directions on its expected energy behaviour through the “Energy Adaptive Software Control Interface” (also known as Southbound Interface).

Currently, software applications and systems are neither aware nor adaptive to energy/power constraints, therefore DC4Cities has developed a set of sample controllers for some common classes of typical data centre software applications; the templates of these controllers can then be expanded to make other software systems become energy adaptive.

DC4Cities might become a standardization candidate, so that software developers might natively integrate this interface into their applications that would then become natively energy adaptive and, therefore, connectable to a DC4Cities Control System, as represented by the yellow arrow (bottom centre) in the diagram (see Figure 2).

### III.2. D4C Control System

Figure 3 highlights all the modules that are included into D4C Control System.

---

\(^1\) The actual control flow might be different based on potential standardization effort of data access methods.

\(^2\) In the picture, numbers represent steps performed during D4C loops. The square frame around the number highlights the fact that the step involves one of the external interfaces (either Northbound or Southbound).
A detailed description of each module of DC4Cities architecture is provided in the next chapters.

As shown in Figure 4, the D4C Control System periodically executes three control loops that are described in the next sub-sections:

- The Power Planning Loop;
- The EASC Control Loop;
- The EASC Monitoring Loop.

III.2.1. Power Planning Loop

The Power Planning Loop activities are described in Figure 5.

During this step, the D4CProcess Controller module (shown at the centre of Figure 3) retrieves through Northbound Interface (data flow labelled as 1 in Figure 3) the following information:
- Energy Availability Forecast (from the Grid) that provide the amount of power that will be available, the sources used and related percentage of power obtained from those renewables;
- Amount of green power provided by Renewable Energy Providers (or by PV panels installed in the roof of the building) or Weather Forecast in order to calculate the related forecast.

### III.2.2. EASC Control Loop

The D4C Process Controller module coordinates all activities in the EASC Control Loop. The time window base consists of a number of time slots, e.g. 24 hours divided into 15 minutes slots, and a loop iteration is performed at the beginning of each time slot.

The first part of the loop is described in Figure 6.

![Figure 6: EASC Control Loop – defining DC power plan quota](image)

During this step, the D4CProcess Controller module, through Max/Ideal Data Centre power planner module (data flow labelled as 2 in Figure 3), use the information previously retrieved (described in section III.2.1.) and the data centre power consumption plan (computed in the previous cycle), in order determine the overall data centre power consumption plan – i.e. the maximum (or ideal) power consumption the data centre can afford, still satisfying the energy/power objectives (like percent of renewable energy, or power capping for certain sources, etc) set by the EMA-SC.

The overall data centre power plan is then split into quotas (data flow labelled as 3 in Figure 3) for each software application/system running in the data centre: the criteria used to perform this split are based on a set of Power Split Policies (data flow labelled as 4 in Figure 3).

This set of policies can be selected and configured for the data centre by the DC business manager (DCBM) and can implement different strategies/algorithms.
The different power plan quotas will be passed to the Energy Adaptive Software Controllers (EASC) through the Southbound Interface (data flow labelled as 5 in Figure 3).

The second part of the loop is described in Figure 7.

![Figure 7: EASC Control Loop – defining global data centre plan](image)

During this step each EASC responds to the input received by D4C Process Controller module this request (data flow labelled as 5 in Figure 3) with a set of alternative power plan options that try to match as close as possible the power plan quota. Multiple options should be returned with at least one meeting the baseline request, but also other variations even if slightly above the request to provide higher degrees of freedom at the global consolidation step.

The total set of all power plan options by all EASCs are analysed and consolidated into a single data centre plan (data flow labelled as 6 in Figure 3), by computing the best combination of the plan options. The optimization capitalizes then on the flexibilities expressed in the power plan options and to the data centre energy contracts (e.g. GreenSDAs⁴) to refine the power consumption plan for the next time window.

The global data centre plan is built through the selection of one specific plan option for each controlled software application/system, among the ones proposed by EASC.

The output of this step, together with power/energy goal metrics and possible constraint violations are sent to the Escalation Manager (data flow labelled as 7 in Figure 3). It analyses retrieved data and provide short, medium and long term status and metrics and alarms that will be displayed inside the dashboard and will also be used, as shown in Figure 8, to involve DCBM and possibly EMA-SC into an escalation process. This process refines data centre service priorities or power/energy goals and retry the problem resolution again (the correct configuration of data centre policies and a realistic tuning of power/energy goals should make this step infrequent).

---

If no exception has to be managed in the very short term (or when they will be solved), as described in Figure 9, the decision of which is the plan option to be enacted is communicated to the EASC through the Southbound Interface (data flow labelled as 8 in Figure 3) and the overall data centre power plan is displayed in the dashboard for the interested actors (ERDS or EMA) to allow higher levels of planning and optimizations (data flow labelled as 9 in Figure 3).
III.2.3. EASC Monitoring Loop

The purpose of the EASC Monitoring Loop is to retrieve regularly from EASCs information related to business performance and power consumption and to provide them in a structured method to the D4C Monitoring Interface to avoid loss of information and to upload consistent data to the historical data base, so that forecasting models can be updated and provide consistent results.

III.2.4. D4C Control System complementary modules

In this section all components of the Control System supporting and enabling the above-mentioned loops are described.

The DC Energy/Power and IT Dashboard provides a GUI interface for DCBM and EMA-SC to monitor and control/configure the high level energy/business view of the data centre control system. This module (shown at the centre right of Figure 3) interacts with the main control modules through a specific internal web service interface, allowing the D4C Dashboard to be implemented as a web application.

The D4C control system relies on a specific persistent configuration that is managed by the DC4Cities system administrator (D4C ADM), and this information is internally distributed to the other modules by the D4C Configuration Manager component.

To enable efficient testing under several different environmental conditions, all D4C modules refer to a D4C Virtual clock for all timings; this for instance allows replaying past scenarios as well as using time compression to run tests faster.

The set of modules in green colour on the left side of Figure 3 are dedicated to the management of historical data. Data are exported from the existing monitoring systems of the data centre IT and DCIM systems into the D4C historical data base through the Energy/Power/Infrastructure monitor components after their consolidation (as described in section III.2.3. ). An historical graph viewer supports the Dashboard presentation with graphs about the past, actual and forecasted data, as well as showing the savings achieved through the different DC4Cities optimizations, and the positioning of D4C performance with respect to the energy/power goals.

Finally, a data analysis and correlation module support the forecasting needs of D4C components by providing a forecast/estimate formula generator that modules can use internally, as well as a forecast/estimate executor, directly computing the results of the formula for a given input data set.

III.3. Northbound Subsystem High level architecture

The Northbound Subsystem contains different modules that are depicted in Figure 10 on a high level.

The Northbound Subsystem helps DC4Cities at collecting both environmental data (such as detailed weather information) and data on locally produced power. These data are used to forecast future renewable energy availability by the Forecast Controller.

The Monitoring Interface offers a unified way to submit data and may be implemented by ERDSs in the future. DC4Cities uses “Connectors” to connect the peculiarities of the different data providers with respect to the unified “Renewable Energy Forecasting Interface”.

A Local Forecaster will be used to provide a renewable energy availability forecast in case the ERDS provide only environmental data.

For example, in the case of photovoltaic panels directly providing power to the data centre. By exchanging data with the Forecast Controller via push or pull mechanism the Forecaster builds
the relevant forecast which will be stored in the Historical Database. However, these predictions are requested by the D4C Process Controller via the Forecast API.

III.4. Southbound Subsystem High level architecture

As depicted in Figure 11, the DC4Cities Control System interacts with the modules of the Southbound subsystem using the EASC Interface.

The Southbound subsystem is composed of three generic modules. The EASC Planning module is in charge of providing a few option plans that schedule workload execution. These option plan algorithms are defined within WP5. These option plans will be returned to central system. Central system will consolidate and optimize workload scheduling selecting the best
and the most optimized option plans. The result of this step is an activity plan for each application. EASC Execution module will receive the activity plan from the central system and will enact workload execution based on that.

Finally, EASC Monitoring module is in charge of monitoring application execution to measure energy consumption of each application under its control.

### III.5. Federation

The data centre federation is for DC4Cities an important mechanism to improve renewable energy related metrics at least for two different reasons:

1. Federated data centres with different energy providers, e.g. grids from different providers and/or different local availability of renewable sources, can have a more efficient workload distribution over the time, capitalizing on different renewable energy availability patterns. For instance the highest PV production is in the centre of the day, while wind is often available also at night.

2. A larger IT resource pool lets additional degrees of freedom for an optimized allocation of the workloads to be exploited by the DC4Cities central system workload consolidator. The various option plans generated by the EASCs can be reorganized with fewer constraints, and new optimization deployment strategies can be enabled.

DC4Cities focus is not on the technical (or security and compliance) details on when and how applications and services can be migrated or relocated: DC4Cities considers the capability of the SW to be potentially migrated/relocated as an attribute that will be configured and exposed by the EASC to the controller only by SW services that meet all the needed prerequisites. Many other IT focused projects, and multiple consolidated IT industrial technologies have already been experimented and used in production systems to achieve the goal of moving SW between data centres. In the past FIT4Green\(^4\) and All4Green\(^5\) projects, where several partners of DC4Cities consortium were involved, the cases of single and multiple ownership of the federated data centres has also been examined in relation to energy saving policies, as well as the usage of a broker service to identify “on demand” the best data centre partner, both from compliance and from energy perspective, among as set of independent data centres.

Rather DC4Cities focuses on decision policies that will rely on migration technologies to achieve better renewable energy related behaviours: i.e. how to deploy SW services inside a set of federated data centres, so that the total energy metrics of the data centre federation wide will improve.

DC4Cities will consider two logical models for the “organizational” aspects of the federation of data centres:

- The **Smart City Assisted / On demand federation**: in this setting each data centre is autonomously controlling its resources and is seeking to reach its power/energy goals specified by the Energy Management Authority of the Smart City (EMA-SC). As described in Figure 12, only if the data centre, through its energy planning and management tool foresees a potential issue in respecting the power/energy goals in the short/medium term (some hours in advance), then the data centre (business and energy) manager will setup an escalation to EMA-SC asking for support in managing a critical situation. This request will have a specific time range, and will contain the proposal for the relocation of one or more SW service to another compatible data centre, so that the critical situation can be mitigated. The Smart City EMA, who has and overall visibility of the “associated” data centres can facilitate finding a suitable partner data centre to host the services. Once the destination DC is selected, services, as

---

\(^4\) [http://www.fit4green.eu/](http://www.fit4green.eu/)

\(^5\) [https://www.all4green-project.eu/](https://www.all4green-project.eu/)
shown in Figure 13, will be migrated through ad hoc manual processes performed by the data centres operators, under the supervision of the respective managers.

In the depicted case DC4Cities is used in data centre 1 (on the left of Figure 12), where at the end of the usual planning loop (max/ideal power plan, splitting policies, EASC option plan proposal and consolidation) the DC4Cities escalation manager module notices a potential energy issue (represented by the red triangle bottom left) and notifies the DC manager, providing a list of potential migration candidates. The DC manager, looking at the DC4Cities Dashboard (Service View) and simulation tools identify the best SW candidate for migration that would fix the energy issue. This info will be reported to the EMA-SC with an assistance request.

The target DC may or may not have DC4Cities installed, in fact its DC manager can use any tool as decision support system for the evaluation of the impact of running an additional application in the data centre (obviously DC4Cities features would facilitate this case as well, but it’s not a must in this type of “loose” federation). If the EMA-SC proposal is accepted by the second DC (Figure 13), then operators will take care of the technical aspects of the migration, while DC business managers will finalize the financial aspects of the transaction.

- **The Continuous federation:** in this setting all the federated data centres need to have DC4Cities installed in federated mode, and accept to have a single optimizer taking decisions on the actual deployment targets of the SW services that can potentially be executed on multiple data centres. Thus there will be a centralize federation wide option plan consolidator that will identify the optimal deployment strategy for the full federation, based on the energy metrics of all data centres.
Figure 14 represents a high level view of the interactions of the various DC4Cities modules inside a federated configuration.
Each data centre has a different power sourcing, and therefore will compute its own max/ideal power plan (top left and top right) that might have quite different profiles, based on the different types/amounts of renewable sources from each provider.

The various power plans will be considered separately by the power splitters as separate pools, where to potentially allocate the various SW services/application running in the different data centres. Each SW service will be configured inside the EASC based on its federation capabilities; the following options are possible:

A. SW can be deployed only in a specific data centre;
B. SW can be potentially be deployed on a list of data centres, but only in one data centre at the time;
C. SW can be potentially be deployed on a list of data centres, also in parallel on more than one data centre (and in different working modes).

The option plans produced by the EASC will respect the federation configuration of each single SW service, therefore these plans can be – respectively – always for a single data centre, for a sequence of alterative data centres (in different time slots), or include multiple simultaneous deployments on different data centres, e.g. application with inter DC load balancing.

All option plans will be consolidated by a single federation wide optimizer, which will then send to the various EASCs in the federated data centres the execution directives for the optimal plan.

In this scenario all DC managers agree to share all resources and have them managed by a single common shared optimizer. Each DC manager still has the visibility of all decisions taken through the DC4Cities dashboard.
In case DC managers would like to keep the control over their not shared SW services (the ones of case A. in the above categorization), a two level decision process might be considered using a slight variation of the previous process, as presented in Figure 15.

The basic difference is that the common process described above will manage only the SW services that can potentially be deployed in the federation (type B. or C.) as represented in the central part of the diagram (blue background). Other SW services that can be deployed only in one data centre are excluded from the common shared optimization. The results of the first level consolidation, i.e. the selected option plans for SW services of type B. and C. are then consolidated at a second level that is data centre specific (brown background aside): each data centre will therefore perform the optimization of its own specific service, merging the results of the first level optimization of the shared service.

This variant of the process on one side keeps definitely more control in the hands of the single data centre managers, on the other side generate a suboptimal overall solution, since the split of resources might exclude in advance some possible solutions of the problem.

From the conceptual point of view, it’s possible to imagine also the combination of the two major federation models. A Continuous Federation that uses an On Demand assistance to manage its federation-wide exceptions for Smart City escalations in case of very complex federation-wide energy issues.

It is worth to notice that EASCs role here is to control federated applications. A federated application is an application that either spans several data centres, or that is able to migrate from one data centre to another. There is only one EASC Planning module to oversee on a given application, even if this application is spanning several DCs.

In federated mode, we make the assumption that the application is already enabled for federation activities: it is assumed that it can balance its workload across data centres and/or migrate part of its activities through them. It is also assumed that the monitoring and enacting of the working modes of the application can be performed by the EASC modules: depending on where the application is physically running and the security restrictions of the inter-data centre communications for IT monitoring and automation tools, EASC execution and monitoring module might require distributed agents in the various data centres.
IV. FUNCTIONAL SPECIFICATIONS

In this chapter we describe the functionalities offered by each component and the technology/product which will be used to implement this component.

IV.1. D4C Control System Single Site Architecture Specification

In this section we discuss how the D4C Control System works in a single site scenario, when no federation with remote data centres is required.

The following component diagram (see Figure 16) shows the detailed architecture, highlighting:

- the internal components
- the relationships between internal and external components
- the main interfaces exposed

From right to left, the Web Application Archive (WAR) components are implemented as Web Applications, based on the Java Enterprise Edition specification; the Historical Database is implemented using the open source products Kairos DB and Cassandra DB; Energis is a Freemind product that is used as the Energy Management Information System (EMIS).

The Front End is composed of the various graphical user interface (GUI) modules used to interact with human actors, for example showing graphs about past and planned energy usage in the data centre.
The Back End includes the main business logic modules that participate in the energy optimization loop process and interacts with external systems via the Northbound and Southbound interfaces in order to collect data and schedule optimized activity plans.

The Service Gateway is an abstraction for the Energy Management Information System (EMIS) needed to implement the whole system. It is implemented as a wrapper of the functionality provided by the Energis product.

Finally, Connectors are provided as libraries to facilitate access to the Historical Database and Service Gateway from components in the Northbound and Southbound layers.

Intra application module communication is implemented using standard Java interfaces, depicted in the diagram above as dotted arrows, while external communications happen via RESTful APIs using JSON to represent data. Communication flow is depicted using solid line arrows, while the most relevant API endpoints are represented as blue circles.

Most of the components are implemented as Java POJO classes, leveraging de-facto standard open source frameworks such as Spring and Jackson for building the application infrastructure and providing common functionalities such as JSON parsing. The user interface in the Front End is implemented using HTML 5 and modern Javascript frameworks such as AngularJS and jQuery.

Each functional aspect identified in the High Level Architecture is covered and implemented by one of the internal components, as will be described in the next sections.

### IV.1.1. DC Process Controller Module

The D4C Control System tries to achieve the data centre energy objectives through the use of an automatic optimization process (described in section III.2.) that runs periodically and tries to match the availability of renewable power with workload requirements of IT services. The optimization produces activity plans that are sent to the Southbound layer for execution. Running the optimization process and interacting with the Northbound and Southbound layers is the main task of the DC Process Controller module.

Specifically, the DC Process Controller module implements the following functional roles:

- Acts as a workflow manager that triggers the optimization loops periodically, retains the system state between iterations and uploads relevant metrics to the Historical Database
- Coordinates the interaction among the components involved in the optimization
- Implements a Data Centre REST API that allows to get information about the system configuration and current state and manage workflow execution
- Implements a virtual clock that allows to manipulate the time seen by the system for testing purposes

Figure 17 shows the internal functional architecture of the module.

The Workflow Manager acts as the main workflow controller. It is initialized with the Process Configuration, provided by the Configuration Controller module, which contains settings such as the frequency of optimization loops and the endpoints of ERDS and EASC instances. Based on configuration settings and the Virtual Clock, the Workflow Manager is responsible for starting the execution of the Power Planning, Control and Monitoring loops at the correct real or virtual time.

Each loop is delegated to a dedicated component in charge of coordinating the interaction between the internal components that implement the actual business logic.

The Power Planning Loop is implemented by the Power Planning Process component, which executes the following steps:

1. Get energy forecasts by invoking the Northbound Forecast API
2. Compute the ideal power plan by invoking the DC Power Planner Module
3. Upload the new ideal power plan to the Historical Database
4. Return the new ideal power plan to the Workflow Manager

The EASC Control Loop is implemented by the Optimization Process component, which executes the following steps:

1. Split the ideal power plan into quotas for each EASC, by invoking the DC Power Plan Splitter Module
2. Send power quotas and collect option plans from EASC controllers, by invoking the EASC Handler Module
3. Select an activity plan for each EASC based on the collected option plans, by invoking the DC Power Plan Option Consolidator Module
4. Send the selected activity plans to EASC instances for execution, by invoking the EASC Handler Module
5. Calculate the consolidated power plan by summing up the estimated energy consumption for the selected activity plans
6. Assess the quality of activity plans by invoking the Escalation Manager Module which will detect any critical conditions such as unmet SLAs or unsatisfied energy objectives
7. Upload the new consolidated power plan to the Historical Database
8. Return the new consolidated power plan and the analysis done by the Escalation Manager to the Workflow Manager

The EASC Monitoring Loop is implemented by the Monitoring Process component, which executes the following steps:

1. Query EASCs for updated metrics, by invoking the EASC Handler Module;
2. Apply required normalization rules to the metrics;
3. Upload the normalized metrics to the Historical Database.

Note that as a general guideline, in order to simplify implementation, all business logic components involved in the optimization loops are stateless. The Power Planning, Optimization and Monitoring Process components retain the state of the current loop while they are running, but their life is limited to the duration of the loop iteration, since a new instance is created by the Workflow Manager for each iteration and terminated afterwards. Therefore
the only stateful component in the Process Controller is the Workflow Manager, which retains the system state for the life of the application. This is the reason why loops return the result of their execution to the Workflow Manager, which can update the Process State. Note that the Process State is considered volatile and stored in memory only, so it is lost if the application is stopped. Any data that must to be stored durably, such as power plans and execution metrics, is persisted to the Historical Database.

As mentioned above, data uploaded to the Historical Database by the Process Controller includes metrics about actual performance and energy consumption, collected by querying the EASC instances in the Monitoring loop. Using the Process Controller as a collector, instead of letting every single EASC upload its own metrics to the database, allows applying the same normalization rules to all metrics. For example some metrics need to be uploaded as average values over a time slot, with their timestamp aligned to the start of the time slot and using a certain known identifier. Uploading all metrics in a consistent way thanks to a centralized control point allows simplifying subsequent use by the Dashboard or Energis, without requiring each EASC to implement normalization rules separately with the risk of discrepancies.

Finally the DC Process Controller Module exposes an administration API, called Data Centre API, which allows interacting with the system from external applications such as a dashboard. Supported actions include reading system information, such as configuration settings or the current consolidated power plan, and controlling the optimization process, for example starting a new optimization loop manually. The API is defined in JSON/REST format.

IV.1.2. DC Power Planner Module

The DC Power Planner module is responsible for building the so called “DC max/Ideal Power Plan” (IPP), i.e. giving boundaries on power consumption for all time slots in the considered future time frame. At the same time, it will make sure to satisfy the correct usage of renewable energy for the next defined timeframe (e.g. the next 24 hours). This module gets information from the Northbound Subsystem via the Renewable Energy Forecasting Interface about the predictions on the availability and properties of future energy of the power connection of the data centre (Grid, Micro Grid, local plants). Additionally, the current Data Centre Power plan forecast will be sent via this interface. This data will be used to calculate a power plan which satisfies the energy/power objectives configuration settings, which define the EMA expressed goal on how to use the renewable energy/power in the defined timeframe.

![Figure 18: DC Power Planner Module](image-url)
The Figure 18 describes the functionality of the DC Power Planner.

The three graphs on the left represent the available power forecasted from now (the time the optimization process iteration has started) to 24 hours into the future, divided in discrete time slots (e.g. of 15 minutes). For each slot the forecaster provides power values in terms of source mix: total power, percentage of renewable power and CO2 emissions.

To calculate the DC Ideal Power Plan three things are needed:

1. The power forecasts obtained by Energy-Related Data Supplier Forecaster, which may be of two different types:
   - Data coming from the grid. The total power can also be “ideally infinite”, while the percentage of renewable refers the percentage of value drained from the grid by the data centre, as a composition of any kind of renewable source mix (from wind, PV, etc.) that composes that power. It’s up to the forecaster to collect that relevant information from external systems.
   - Data coming from the local renewable energy plant (e.g. local photovoltaic plant). The total power refers to the max power produced by the plant, while the percentage of renewable is equal to 100%. It’s up to the forecaster to base its forecasts on any relevant data provider useful to determine that values (weather forecast system, etc.)
2. DC Energy/Power Budget objectives defined by EMA.
3. The current DC Power plan to solve any objective that refers to percentage of consumed power and also to any objective that expresses energy goals.

The data centre needs a minimum power to ensure compliance with the SLA of IT services, therefore in each time frame the IPP should provide the necessary amount of power. This power is used by the data centre’s hardware infrastructure and facility infrastructure to run the IT services installed inside the DC itself without any effect at the end user side or at least with accepted drop in performance by the end user.

**IV.1.3. DC Power Plan Splitter Module**

The DC Power Plan Splitter module is responsible for calculating the power budgets dedicated to each Energy Adaptive Software Controller (EASC). This module takes as input the ideal power plan for the optimization interval (e.g. the next 24 hours) and splits it into smaller quotas, each one dedicated to a single EASC in the system.

Considering that many different splitting strategies may be adopted and different choices may impact the behaviour of the services managed by the EASCs, the splitting process must be driven by proper policies defined as part of the contractual agreements established between the actors involved in the system.

As an example, the simplest policy might be that the power budget is equally distributed between all the EASCs in the system. However, it would probably be far from being the optimal one, since the power requirements of the EASCs may differ. For instance, one controller may receive a budget higher than needed; another controller may receive a budget which is too low.

In order to take the different requirements for each EASC into account, the Power Splitter implements a weight-based strategy with two levels of splitting. At configuration time, EASCs are assigned to groups and each group is assigned a weight; furthermore, each EASC has its own weight within the group. Using this information, the Power Splitter performs a first division of the ideal power among groups proportionally to group weights, and then does a further division among the EASCs within each group, again using the weights of the EASCs. However, while at the group level the splitting is done on the instant power available for each time slot,
at the EASC level the division refers to percentage of the total energy that can be used by the EASC in the whole optimization interval.

**Figure 19: DC Power Plan Splitter Module**

In the example shown in Figure 19, the EASCs are divided in two groups. The first group contains only one EASC, called EASC A, while the second group contains two EASCs, called B and C. Considering an ideal power of 1000 W, first the power is split into the two groups based on their weight. Group 1 has weights 1 and gets 333 W, while group 2 weighs 2 and gets 667 W. Then considering the second level of splitting, EASC A is the only element in group 1, so it gets 100% of the energy; on the other hand group 2 contains two elements, so the 667 W must be divided among them. Based on their weights, EASC B gets 60% of the energy and EASC C gets 40% of the energy.

**Figure 20: Per-EASC vs group splitting**

In order to understand the advantages of the new version of the splitting algorithm, it is important to stress the difference between assigning power and energy to the EASCs. While
the power quota defines the maximum instant power that an EASC can consume for each time slot in the optimization interval, the energy quota defines how much work the EASC can do during the whole time interval, with the freedom to reach higher power usage peaks as long as this reduces the time needed to complete work. In other words, instead of just slicing the ideal power horizontally and assigning a reserved slice to each EASC, we assign a shared, higher power quota to a group of EASCs and we give each EASC an indication on how large is the area of the power curve that it can use, with the freedom to move it in time. This allows achieving a better degree of optimization, as shown in Figure 20.

The upper part of Figure 20 shows the situation when each EASC gets a dedicated power quota, while the lower part shows group splitting. In the first case, the ideal power is divided between all EASCs and thus the resulting per-EASC quota is lower, while in the second case the ideal power is divided between a smaller number of groups and so the per-group quota is higher. Of course this quota must be shared, so each EASC must produce some option plans taking into account the percentage of energy it has been allocated. However, since the peak power in the quota is higher, the EASC may be able to use better performing working modes than if it had a dedicated but smaller power quota. Furthermore, since using better working modes allows completing work in less time, the EASC can produce shorter plans positioned in different parts of the day, such as BX1 in the morning and BX2 in the evening, instead of longer low-performing plans that span most of the day, like B1 and B2. Being able to choose among short plans positioned at different times of the day allows the consolidator more flexibility in finding combinations that maximize renewable energy usage. For example if our group included B and C, the consolidator might choose to let B execute a plan that works at maximum power in the morning and let C run at maximum power in the afternoon, thus maximizing renewable energy usage during the sunny hours of the day.

IV.1.4. EASC Handler Module

This module is responsible for the communication between the D4C Control System and EASC instances. Its purpose is abstracting the details of the communication protocol from the Process Controller and providing high level functions for:

- Collecting option plans based on the power quota assigned to each EASC
- Sending to each EASC the activity execution plan selected by the controller
- Querying EASCs to get metrics about activity execution

The option plan collection is the step of the EASC Control Loop in which each EASC is asked to provide one or more option plans based on the assigned power quota.

Each EASC controls a specific IT Service and is able to change the way the IT Service works and consumes power. The idea is that an IT Service can run in different configurations called working modes. A working mode specifies the number and the type of resources needed to support the IT Service SLAs and also the actions to be taken to switch to that configuration.
For instance, for a multi-tier architecture, a working mode may specify the number of web servers, the number of application servers and the number of database instances and it should be possible to switch from one configuration (2 WS, 2 AS, 2 DB) to another one (4 WS, 3 AS, 2 DB). As another example, working modes for a batch processing application may be associated to different sets of machines, with some configurations leveraging fast powerful CPUs and others using slower CPUs that consume less energy.

It is sufficiently reasonable to assume it is always possible to identify different architectural configurations of an IT Service that imply different ways to support the load and the implied power consumption. From the Control System point of view, it is not important to know the internal details of each working mode, since it is up to the EASC to hide and manage those details. The Control System must only know that there are different choices on how an IT Service can work and that each choice has an associated power consumption.

An option plan expresses this concept for a time period. In fact it contains information regarding the possible working modes, with the corresponding estimated power consumption, that the controlled IT Service can use to run at a specific time slot. In this way for each time slot the most suitable working mode can be selected for the specified power quota.

After the Option Consolidator module (see section IV.1.5.) has selected the activity plans for all EASCs, the Process Controller invokes the function of the EASC Handler that sends each EASC its own plan for execution.

In this way the EASC is informed of the chosen plan and it can enact any required working mode changes on the relevant IT Service.

Since the number of EASC instances inside a data centre may vary from a few units to thousands, the Control System should be scalable enough to handle them. While the set of EASCs managed in the Control System prototype is defined statically in the Process Configuration, a production-ready version may support adding or removing EASCs at runtime, for example for maintenance reasons.

Finally, in order to support the EASC Monitoring Loop in the Process Controller, the EASC Handler allows querying EASCs for collecting execution metrics, such as the actual power consumption and business performance.

Figure 23 describes the internal functional architecture of the EASC Handler module, needed to support the above features.

The EASC Handler component exposes an interface to the Process Controller that hides the details of interacting with the EASCs and provides Java methods to collect option plans, send activity plans and collect metrics. Internally, the EASC Handler relies on an EASC Registry, populated via the Process Configuration, which contains the list of EASCs registered in the system with their API endpoints.
The communication with the EASCs is done using a JSON/REST API. In order to speed up operations on multiple EASCs the Handler may execute parallel invocations in separate threads. Results are returned to the Process Controller only when responses have been received from all EASCs.

IV.1.5. DC Power Plan Option Consolidator Module

After sending Power Quotas and retrieving back a list of selected power plan options from each EASC, the D4C Control System is responsible to consolidate the power plan options collected for each time slot. Figure 24 illustrates the typical power plan option consolidation process.

For phase 1, the option plan consolidator focused on reducing the energy footprint by selecting the plans that minimized the overall consumption of non-renewable energies. The consolidator did not consider the possible performance penalty an EASC can encounter when it agreed on reducing its energy footprint. Indeed, despite the notion of green-point has been defined for that purpose, the semantic of this property was too abstract for being integrated wisely into the system. Such an approach is not appropriate in reality. In a same manner the consolidator had to fit the smart city authority expectations in terms of energy governance. This may tend to a situation where a solution does not simply exist. Finally, we ignored the energy market price.

For phase 2, we decided to move the option plan consolidator behaviour from strict energy-efficiency oriented model to an economical model. With regards to:

- The energy availability and its associated cost,
- The economical penalty for the EASCs that have to run in degraded mode,
- The possible benefits for the consolidator that can sell exceeded green power to some EASCs,
- The penalty for violating the smart-city authority constraints.

The option plan consolidator will select a plan for each EASC that tends to maximize the data centre revenue.
This approach can be considered as a negotiation-based system that aims at finding a sweet spot for every actor. In this case, the economic model is the pivot metric that allows integrating more concerns than during phase 1. The practical benefit we expect from this new approach is:

1. To obtain the most profitable trade-off that will satisfy all the actors (each EASC, the datacentre provider, the smart-city authority) according to their stated tolerance.

2. An efficient reporting for the escalation manager in case of critical situation. Indeed, the computed solution is richer in terms of information as it exhibits the most important constraint to satisfy.

**Example:**

The following example shows the options sent by two EASC, containing their energy consumption in kW for a period of 4 time slots. Each EASC proposes two different options to apply:

<table>
<thead>
<tr>
<th>Time Slot</th>
<th>EASC 1</th>
<th>EASC 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Option 1a</td>
<td>Option 1b</td>
</tr>
<tr>
<td>T1</td>
<td>160W</td>
<td>100W</td>
</tr>
<tr>
<td>T2</td>
<td>120W</td>
<td>100W</td>
</tr>
<tr>
<td>T3</td>
<td>120W</td>
<td>100W</td>
</tr>
<tr>
<td>T4</td>
<td>200W</td>
<td>100W</td>
</tr>
</tbody>
</table>

**Table 3:** Sample power demands for 2 EASCs having each 2 options
Table 4 summarizes the different costs when option 1b leads to a penalty of 2€, option 2b leads to a penalty of 10€ (other options have no penalties), and the energy price equals 0.03€ per Watt:

<table>
<thead>
<tr>
<th>Watts</th>
<th>Penalty</th>
<th>Overall Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a, 2a</td>
<td>1100W</td>
<td>0€</td>
</tr>
<tr>
<td>1a, 2b</td>
<td>900W</td>
<td>10€</td>
</tr>
<tr>
<td>1b, 2a</td>
<td>900W</td>
<td>2€</td>
</tr>
<tr>
<td>1b, 2b</td>
<td>700W</td>
<td>12€</td>
</tr>
</tbody>
</table>

Table 4: Overall costs for the data centre provider depending on the selected options

Using the phase 1 approach that selects the cheapest plans in terms of watts, then options 1b and 2b should be preferred despite the EASCs penalty. If we choose the plans that lead to the least amount of penalties, then we will increase the overall energy consumption and select options 1a and 2a. Using our new economic model, the consolidator would choose options 1b and 2a because the resulting penalties are worthy to pay with regards to the energy price.

IV.1.6. Power Plan Escalation Manager Module

The Power Plan Escalation Manager module has the main objective of evaluating the central system status, with particular focus on the possibility of satisfying power plans in different situations. This component has the main task of evaluating each of the EASC Activity Plan in order to identify possible critical issues in terms of energy requirement and to notify them on the Dashboard.

The Escalation Manager module interacts with two modules which are:

1. The Process Controller (Back End side)
2. The Dashboard (Front End side).

The Back End contains the main logic modules that encompass the datacentre energy control loop process, whereas the Front End is composed of the Graphical User Interface (GUI) modules used to interact with human actors.

A typical workflow starts with the Controller that needs a feasibility analysis, request an evaluation to the Escalation Manager. The Controller sets the proper evaluation parameters on the Escalation Manager and invokes the evaluation method giving a list of plans and a list of energy forecast as input for the Escalation Manager. The results are temporarily stored in the Escalation Manager to allow the Dashboard to take and display them. For this reasons, the Escalation Manager can be considered as a stateless component that is created and used each time it is necessary.

IV.1.6.a. Evaluation

Regarding the evaluation methodology, the Power Plan Escalation Manager component includes a method responsible for such an evaluation and a set of variables that identify the results. This method receives, as an input, a list of current execution plans in the system (i.e., a list of EASC Activity Plan), and a list of objectives related to the expected energy availability for the whole system (i.e., energy forecast).
In Figure 25 we show an example of a set of EASC Activity Plans of 24 hours and their works, with time stamp intervals.

For example, taking into account the timeslot1, the total power required is the sum of the power consumption of work1, work5, work7, and work10, whereas for timeslot5, the total power required is the sum of the power consumption of work3, and work5. The colored segments represent the various works of EASC Activity Plans.

The evaluation output is based on energy forecast and, for each timeslot, the component provides a status representing the level of feasibility of the objectives w.r.t. the forecast. The evaluation is based on the concept of the traffic light that exhibits a state related to the comparison between required and forecasted power. In this respect, for each timeslot, we can have three possible situations (as shown in Figure 26), than are:

- Green – the total power required is quite less than the forecasted one;
- Red – the total power required is more than the forecasted one;
- Yellow – the total power required is less than the forecasted one, but within a warning level;

For example, look at the following figure:
The escalation manager performs the evaluation for each time slot looking at the max forecasted energy available, and store a system status based on traffic lights. In particular, the escalation manager is configurable by setting, before the evaluation request, the warning level in percentage with respect to the max forecasted energy available within which evaluating the state as a warning condition (i.e., yellow traffic-light).

The Power Plan Escalation Manager module can also perform a further analysis, which corresponds to a high-level analysis, which is based on time intervals. In particular, such an analysis aggregates single time slot evaluation to provide a short, medium, and long-term analysis as shown in Figure 27.

![High-level analysis based on time intervals (short, medium and long-term)](image)

The escalation manager has settable variables also for this analysis, that allow configuring time interval length and also timeslot aggregating strategy for time interval evaluation (a possible strategy can be, e.g., taking the most risky alarm to evaluate the time interval).

**IV.1.7. Configuration Controller**

The DC4Cities System is composed of several components which need to be configured. Furthermore, certain components need to communicate with each other based on the same underlying data. For instance, EMA(-SC) sets power/energy objectives which must be made available for the corresponding DC Control system. In order to provide a consistent view of shared knowledge the Configuration Controller is used. It is a global configuration engine that hides the complexity of the underlying data store (e.g. a database or flat files that provide configuration data). Amongst others, it is supposed to handle the following globally shared configuration categories:

- Objectives set by EMA(-SC) for each participating DC
- DC’s Technical Configuration
- DC’s Energy Configuration
- Power Splitter Configuration

Figure 28 shows a simplified model of the Configuration Controller and its integration into the DC4Cities System. The configuration parameters can be added, updated and deleted for certain configuration models through the configuration API. The DC4Cities EMA-SC and DC
dashboards are the main use cases for exploiting this API (e.g. EMA may set override objectives, etc.).

Technically, the controller provides a RESTful CRUD-based API which offers querying and updating capabilities for certain configuration models. Each component depending on some configuration parameters uses the corresponding RESTful client to query them. The document representation used to exchange configuration parameters is in JSON format.

**IV.1.8. Historical Database**

This Historical database component is the component in which all historical, monitored and forecasted metrics used by the D4C Control system are stored.

The Open Source KairosDB/Cassandra has been used as historical data repository. It is a very fast and scalable time series database. It allows storing data that can be labelled with tags. Values can be of type long or float.

Metrics have been classified in the following categories: Power/Energy, Load and KPI metrics. This is explained in more detail in the chapter VIII.

**IV.1.8.a. Metric values**

A metric value is stored as follows in this Historical database

```
timeStamp metricName metricValue tagName1=tagValue ... tagNamen=tagValue
```

in which the different fields have the meaning described below.
**timestamp**

The timestamp of a metric can be in the past, present or future. Timestamps are in UTC time.

**metricName**

The value have to be unique and have to follow a certain naming convention. See chapter VIII with the supported metric names.

**metricValue**

A metric can only have a numerical value.

**tags**

A tag is a keyword that can be stored together with a metric to ease its filtering.

When the metric is stored, the following tags are added

- assetCode: code to identify uniquely the asset;
- assetType: company | site | server | vm | wm | erds;
- reference: actual | forecasted | estimated;
- granularity: none|1 minute |15 minute| 1 hour.

**AssetCode**

This code allows to uniquely identify an asset.

**AssetType**

The assets can be of different types. Currently following types are supported: company, site, server, virtual machine (VM), working mode (WM), ERDS (Energy Related Data Supplier).

**Reference**

The same metric can be stored with different references. The reference indicates what the metric is used for.

The possibilities are 'actual' for the measured values, 'forecasted' for the predicted values and 'estimated' for the values estimated from other provided values.

**Granularity**

An optional tag. If present, it is expressed as "number timeunit". Timeunit can be "second", "minute", "hour", "day", "week", "month", and "year". Specifying granularity means the metric value refers to the time period after the timestamp. If this tag is missing, then the value is strictly linked to the timestamp. Examples:

- Power at that exact moment in time;
- Power with a granularity means it is an averaged power over that time period.

Irradiation and energy with a granularity means it is the integrated power over that time period (1 minute; 10 minute; 1 hour; 6 hour, 1 day…).

**IV.1.9. Connectors**

Collectors allow to collect and store power/energy, environmental and data centre data as well as data from different sources (ERDS).
Following are examples of the way to store forecasted values for the irradiation metric and actual values for the module_temperature metric.

**Example 1: forecasted values for the irradiation metric**

```java
MetricBuilder builder = MetricBuilder.getInstance();
builder.addMetric("irradiation")
  .addTag("assetType", "site")
  .addTag("assetCode", "hp_milan")
  .addTag("companyCode", "hp")
  .addTag("reference", "forecasted")
  .addDataPoint(timestamp1, value1)
  .addDataPoint(timestamp2, value2)
```

```java
HttpClient client = new HttpClient(<historical_db_address>, <historical_db_port>);
Response response = client.pushMetrics(builder);
client.shutdown();
```

**Example 2: actual values for the module_temperature metric**

```java
MetricBuilder builder = MetricBuilder.getInstance();
builder.addMetric("module_temperature")
  .addTag("assetType", "site")
  .addTag("assetCode", "hp_milan")
  .addTag("companyCode", "hp")
  .addTag("reference", "actual")
  .addDataPoint(timestamp1, value1)
  .addDataPoint(timestamp2, value2)
```

```java
HttpClient client = new HttpClient(<historical_db_address>, <historical_db_port>);
Response response = client.pushMetrics(builder);
client.shutdown();
```

**IV.1.10. Service Gateway**

The Service Gateway is an abstraction for any Energy Management Information System (EMIS) needed to implement the whole system. The Service Gateway provides access to shared energy management services which offer functionalities to the D4C Control System and the Northbound and Southbound Subsystems.

A shared energy management service is the data analysis and correlation service in support of the forecasting needs of D4C components. This is achieved by providing a forecast/estimate formula generator as well as a forecast/estimate executor. The forecast/estimate formula estimator allows identifying formulas by means of correlation modelling using baseline data retrieved from the Historical Database. The forecast/estimate formula executor allows estimating and/or forecasting the electricity consumption and production.

**IV.1.11. Energis**

Energis is an example of an Energy Management Information System providing a forecast/estimate formula generator as well as a forecast/estimate executor.

**IV.1.11.a. Estimate Formula Generator**

Energis allows to generate formulas by analyzing metrics data for a certain historical period. It does so for a certain set of metrics called metrics catalog.
Energis identifies models; the mathematical formula is one element of a model. A model also provides performance criteria to assess the quality of the generated formula. The formula is stored in different formats (Latex, MathML, Java) inside the Energis database. Performance criteria are for example the accuracy, Coefficient of Variation of the Root Mean Square Error (CVRMSE), Mean Bias Error (MBE), R².

In the context of DC4Cities, Energis has been adapted to identify models using the metrics from the DC4Cities metrics catalog. It uses metric values which have been stored in the historical database by the different software components of DC4Cities.

The administrator must manually trigger the identification of a model, using Energis interface. Before launching it, the asset have to be selected as well as the historical period (baseline period), the energy type (Electricity, Gas, Water, Electricity Production), the granularity and the set of metrics to use as variables for the model formula.
Example:

The metrics renewable_power, irradiance and module_temperature have been imported into the historical database for the site of acme_building3.

The administrator can make the selection of the asset, period, energy type and granularity (see Figure 29)

After the identification of the model, the model assessment and the formula are shown to the user (see Figure 30).

**IV.1.11.b. Estimate Formula Executor**

The formulas identified by the Estimate Formula Generator can be executed by the Estimate Formula Executor.

The Estimate Formula Executor is used by the ForecastController (see Deliverable 4.1) and by the EASC (see Deliverable 5.1).

Energis has been designed to execute formulas to quantify energy savings. This happens when a model is being used (see Figure 31)

In the context of DC4Cities the usage of formulas will be extended to be used for:
- Prediction/Forecasting purpose;
- What-if/Estimation purpose.

IV.1.11.c. Prediction/Forecasting purpose

When selecting usage prediction/forecasted, the reference of the metric is changed to become “forecasted”.

If the model identified in the Example described in section IV.1.11.a., is used for prediction purpose, the reference of the metric renewable_power becomes “forecasted” and the input variables become irradiance and ambient_temperature with reference “forecasted”.

It is then possible to query forecasted renewable_power when providing forecasted values for the metrics irradiance and ambient_temperature in the historical database.

A JSON/REST API has been added in the context of DC4Cities to allow other D4C components to query this metric. A java library “queryBuilder” building up these JSON messages has been made available in the DC4Cities source code repository.

A sample JSON/REST request and response querying the forecasted renewable power for the future 24 hours at 15 minutes granularity is provided below:

Request

```json
v1/data/query?apiKey=f47ac10b58cc4372a5670e02b2c3d479
{
   "relativeTimePeriod": {
      "referenceInstant": "2014-03-06T00:12:00.0+01:00",
      "shift": 24,
      "unit": "HOURS"
   },
   "companyCode": "DC4C",
   "assetCode": "acme_building3",
   "metricName": "renewable_power.forecasted",
   "granularity": {
      "value": 15,
      "unit": "MINUTES"
   }
}
```

The response contains the values of the selected metric at the selected granularity for the selected period.

Response

```json
{
   "timeValues": [
      {
         "value": 60,
         "timestamp": "2014-03-06T00:12:15.0+01:00"
      },
      {
         "value": 60,
         "timestamp": "2014-03-06T00:12:30.0+01:00"
      },
      {
         "value": 60,
         "timestamp": "2014-03-06T00:12:45.0+01:00"
      },
      ...
   ]
}
```
IV.1.11.d. What-if/Estimation purpose

When selecting usage prediction/estimated, the reference of the metric is changed to become "estimated".

A model can be identified from historical data to estimate the power of data center resources (see Deliverable 5.1) such as:

- The power consumed by a server from the cpu_usage, memory_usage metrics;
- The power consumed by a virtual machine from the sum_vcpu_usage metric;
- The power consumed by a working mode from the bizperf_items metric.

A JSON/REST API has been added in the context of DC4Cities to allow other D4C components to execute this formula providing different input metric values. A java library "executeBuilder" building up these JSON messages has been made available in the DC4Cities source code repository.

A sample JSON/REST request and response executing the formula to estimate the power for a set of input metrics is provided below:

request

```json
v1/data/execute?apiKey=f47ac10b58cc4372a5670e02b2e3d479
{
  "companyCode": "DC4C",
  "assetCode": "server00",
  "metricName": "power.estimated",
  "inputs"[
    {
      "variableName": "X1", // cpu_usage
      "values": [10,20,50, 90]
    },
    {
      "variableName": "X2", // memory_usage
      "values": [15,25,55, 90]
    }
  ]
}
```

The response contains the values of the selected metric estimated using the variables provided in the request.

response

```json
{
  "values": [60,66,72]
}
```

IV.1.12. Graph Viewer

A graph viewer supports the Dashboard presentation (see section IV.1.13. IV.2.13. ) with graphs about the past, actual and forecasted data, as well as showing the savings achieved through the different DC4Cities optimizations, and the positioning of D4C performance with respect to the energy/power goals.

The Graph Viewer shows per data centre following metrics as a graph at a certain granularity (e.g. 15 min) and for a certain period (e.g. last & next 24 hours):

- actual and forecasted renewable and non-renewable power;
• actual data centre power.

The Graph Viewer has been developed using JavaScript and the HighCharts JavaScript charting library.

**IV.1.13. Dashboard**

The Energy Power Dashboard shows to the user the energy power related metrics about the data centres. The user must only select the site and the time period to visualise these metrics. The user can select the site for which he or she wants to view the impact of the DC4Cities service (see Figure 32).

![Figure 32: Energy Power Dashboard – site selection](image)

Several predefined periods can be selected or the user can make an own custom selection by providing start and end date.

The predefined periods are the following:

- Yesterday: from yesterday 0h until 0h today;
- Last 24h: from last hour until the same hour the day before;
- Tomorrow: from 0h tomorrow until 0h day after tomorrow;
- Next 24h: from current hour until the same hour the day after;
- Last 7 days: from yesterday until the same day of the previous week;
- Last week: from Monday until Sunday of last week.


When the user selects a period in the past, following information is shown (see Figure 33)
There are three charts shown in the Energy Power Dashboard:

- The Production chart shows all the electricity production related metrics for that site;
- The Consumption chart shows all the electricity consumption related metrics for that site;
- The Total chart shows for the total electricity consumption which part is from a renewable source.

Some color conventions are used in the charts which are the following:

- red: forecasted;
- grey: actual;
- orange: planned;
- blue: baseline;
- green: renewable.

The chart **Production** shows following metrics:
- renewable power.actual (W): actual renewable power produced on-site at the data center location (dark green color);
- renewable power.forecasted (W): forecasted renewable power produced on-site at the data center location (dark red color);
- renewable_energy_percentage.actual (%): actual percentage of renewable energy used from the grid (light green color);
- renewable_energy_percentage.forecasted (%): forecasted percentage of renewable energy used from the grid (light red color).

The chart **Consumption** shows following metrics:
- power.actual (W): actual power consumed at data centre level (grey color);
- power.baseline (W): baseline equation before D4C added flexibility in the data center (blue color);
- power.planned (W): plan to which data center tried to adapt itself by means of DC4C (orange color).

The chart **Total** shows following metrics:
- power.actual (W): total power consumption of the site as a stacked bar showing the renewable and non-renewable parts of the total power consumption.

The granularity of the metrics to be shown in the graphs is fixed to hourly values.

**Key Performance Indicators**
Next to the charts three Key Performance Indicators are shown (top left aside the first graph of the page, in bars with dark grey background):
- RenPercent: this KPI gives the percentage of renewable energy consumed by the data centre for the selected time period;
- RenPercentBaseline: this KPI gives the percentage of renewable energy consumed by the data centre for the selected time period in case of baseline energy consumption;
- DCA (DC Adapt): this KPI measures how much the DC energy profile has been shifted from a baseline energy consumption after the implementation of flexibility mechanisms;
- APC (Adaptability Power Curve): this KPI measures how much the DC has adapted its consumption to the power plan.

The detailed meaning and the way these metrics are computed is explained in Deliverable 7.1. These metrics are aggregated over the selected period.


When the user selects a period in the future, following information is shown (see Figure 34)
The chart Production shows the metrics renewable power.forecasted (W) and renewable_energy_percentage.forecasted (%).

The chart Consumption shows the metrics power.planned (W).

The chart Total shows the metrics power.planned (W) as a stacked bar showing the renewable and non-renewable parts of the total planned power consumption.

Key Performance Indicators

Next to the charts one Key Performance Indicator is shown:

- RenPercent: this KPI gives the percentage of renewable energy forecasted to be consumed by the data centre for the selected future time period.

The meaning and the way these metrics are computed is explained in Deliverable 7.1. This metrics is aggregated over the selected period.

Enhancements

The dashboard is being enhanced during the second year of the project with following functionalities:

- Provide a more detailed view to the DC business manager on services and service related metrics (e.g. contribution of the service to overall DC RenPercent);
• Show escalated alarms and details with possible actions (e.g. check for assisted federation);
• Show both consolidated/planned and ideal power in the charts;
• Show extra aggregated metrics (e.g. CO2 savings).

IV.2. D4C Control System Federation Architecture Specification

The D4C Control system in a federated scenario requires the modification of certain modules and the information flow among some of them. In the reminder of this chapter each module will be analysed separately from the federation perspective. The Smart City Assisted/On demand federation simply requires small changes in EASC configuration and Escalation module, while the Continuous Federation requires deeper changes. In this chapter both these cases will be described; for the Continuous federation the single level case will be considered the reference case.

IV.2.1. DC Process Controller Module

The process controller module will coordinate the execution flow of all the other modules, organizing activities inside three loops, with the same scheme used in the single site case. The main difference is the fact that certain module exist in multiple instances, i.e. one for each data centre. Specifically the loop will be organized as follows:

• Power Planning Loop: the loop will include the same information flow described for the single site case through the Northbound Interface (see section III.2.1.), but this sequence will be executed for each power provider of the federated DCs.
• EASC Control Loop: at federation level each DC Power Splitter module will process its data collected from Max/Ideal power planners (from Power Planning Loop). The power quotas will be distributed to all EASCs and their option plans will be collected and processed all together by the single Option Plan Consolidator (Optimizer) to generate the optimal solution for the whole federation. Multiple Consolidated Plans will be produced, one for each data centre: service specific consolidated plans will be communicated to respective EASC for execution. Each DC Escalation Manager will analyze its own consolidated plans to foresee critical situations.
• EASC Monitoring Loop: the logic loop won’t be affected by the federation; obviously all the EASCs configured inside the DC federation will be monitored.

IV.2.2. DC Power Planner Module

The Power Planner module will not be affected by the federation. The only difference is that there will be a separate Power Planner module instance for each data centre in the federation, each one dealing independently with its own power system (above the Northbound interface).

IV.2.3. DC Power Plan Splitter Module

Inside the Continuous federated scenario there will be one Power Plan Splitter each data centre in the federation. Each Power Plan Splitter will receive max/ideal power plans for its data centres, and will apply power splitting policies similar to the single site case. The produced
plans will be then aggregated by service by the Central Controller, to be distributed to the EASCs.

Each SW service will receive one or more power quotas, computed depending on the configured policies, based on its federation attribute/category described in section III.5. SW services of category A (single site) will receive a single quota, while for category B (one site at the time) and C (parallel sites) a set quotas will be received, one for each data centre that the policy will indicate as appropriate for the deployment (and obviously compatible with the federation attribute of the SW service).

The following picture (Figure 35) presents a graphical view on the input and outputs of the Power Splitter modules (one instance per data centre).

---

**IV.2.4. EASC Handler Module**

EASCs provide federated option plans to the Power Option Plan Consolidator. For each SW service of category A (where only one DC is considered for each service, since this category is “deploy only on one site”) the Option plan will be “regular”, i.e. similar to the single site plan, while for category B services, it will contain a list of regular options plans, but each one might refer to different DC in each time slot.

For SW service of category C – in addition to regular Option plans like for category B – also a list of “linked plans sets” might be received by the EASC. In this case the selection of the optimal solution needs to select one linked set (without breaking sets and intermixing plans of different sets). The following picture (Figure 36) represents a graphical example of the different list of options plans for the three different categories.
IV.2.5. DC Power Plan Option Consolidator Module

The first service example at the top falls into category A, i.e. only on DC1; in fact the option plans only refers to DC1. The last one at the bottom is the symmetric case, i.e. only on DC2.
The second service receives quota for both data centres, but since it’s in category B the option plans can either refer to DC1 (first case) or to DC2 (second case).

The third service also receives quota for both data centres, but in category C it’s possible to deploy on multiple DCs at the same time, so a set of “linked” option plans are provided (including 2 DCs in this example) and the consolidator can select only one set of “linked” plan – no other combinations are possible.

The federated option plan consolidator will take all this inputs and – like in the single site case - together with the power/energy goals, the power plans and general constraints, will select the best combination to achieve optimal satisfaction of power/objectives (e.g. renewable energy source utilization). Figure 37 graphically represents the task of the Consolidator module.

IV.2.6. Power Plan Escalation Manager Module

The Power Plan Escalation Manager module plays a very significant role in the Smart City Assisted/On Demand federation, i.e. it detects when it’s worth to escalate an energy issue and retrieves an initial proposal for SW services candidates to be migrated/relocated.

The input of the Escalation Manager is the same as in the single site case, i.e. the consolidated option plan, where the Consolidator modules stores the outlook of the next 24 hours in terms of SW services working modes, planned power consumption as well as degree of satisfaction of the constraints used during the optimization phase. The only added information in case of federation is the list of active services of category B, and C, (i.e. potentially deployable on multiple data centres) for each time slot.

IV.2.7. Configuration Controller

The Configuration module is affected by the configuration, but only for a few parameters to allow the central controller to correctly address the additional module instances due to the federation deployment.

IV.2.8. Historical Database

The Historical database is basically not affected by the federation, since it has been designed from the beginning to be able to host data about multiple independent data centres. The fact that the DCs are federated doesn’t change this structure.

IV.2.9. Connectors

No changes are expected.

IV.2.10. Service Gateway

No changes are expected.
IV.2.11. Energis

No changes are expected

IV.2.12. Graph Viewer

DC4Cities Graphs will be displayed for each data centre separately, both as data centre totals and by each service individually. By aggregating data on the historical data base from various data centres (totals or for the same service), also summary graphs can be generated with a "federation view".

IV.2.13. Dashboard

The DC4Cities dashboard by default presents per data centre totals view, as well as a per service view. The federation view will include summary of aggregated data, e.g. totals for all federated data centres or sum of service data running on multiple DCs.
V. INTERFACE SPECIFICATIONS

This section describes the external interfaces mentioned in section III.1. specified in JSON/REST format. The detailed version of defined interfaces will be provided in the D3.4 deliverable, together with the specification of the interfaces used within the DC4Cities Control System such as the DataCentre API.

V.1. Renewable Energy Forecasting Interface

The communication between D4C Process Controller and the Northbound Subsystem takes place via the Renewable Energy Forecasting Interface, which introduces two types of API:

V.1.1. Forecast API

This API is used to get the power availability forecast required by the DC4Cities Process Controller

```
POST [ForecasterURL]/v1/erds/(erdsName)/forecast
```

### URL parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Cardinality</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ForecasterURL</td>
<td>1</td>
<td>URL</td>
<td>The URL of the forecasting service</td>
</tr>
<tr>
<td>erdsName</td>
<td>1</td>
<td>String</td>
<td>The name of ERDS</td>
</tr>
</tbody>
</table>

*Table 5: Forecast API - URL parameters*

### Request Body

The input parameters specify from which ERDS the forecast is requested. Furthermore, the time interval and timeslots for which the forecast is required is requested:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Cardinality</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$input</td>
<td>1</td>
<td>Object</td>
<td></td>
</tr>
<tr>
<td>dateFrom</td>
<td>1</td>
<td>ISO-8601</td>
<td>The start date and time of the interval for which the forecast is requested</td>
</tr>
<tr>
<td>dateTo</td>
<td>1</td>
<td>ISO-8601</td>
<td>The end date and time of the interval for which the forecast is requested</td>
</tr>
<tr>
<td>Parameter</td>
<td>Cardinality</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------</td>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>$ output</td>
<td>1</td>
<td>Object</td>
<td></td>
</tr>
<tr>
<td>erdsName</td>
<td>1</td>
<td>String</td>
<td>The name of ERDS</td>
</tr>
<tr>
<td>dateFrom</td>
<td>1</td>
<td>ISO-8601 string</td>
<td>Same as the request</td>
</tr>
<tr>
<td>dateTo</td>
<td>1</td>
<td>ISO-8601 string</td>
<td>Same as the request</td>
</tr>
<tr>
<td>timeSlotDuration</td>
<td>1</td>
<td>String: &quot;&lt;integer&gt; min&quot;</td>
<td>Same as the request</td>
</tr>
<tr>
<td>timeSlotForecasts</td>
<td>1..n</td>
<td>Array</td>
<td>An item for each time slot, with the forecast for that time slot.</td>
</tr>
<tr>
<td>timeSlot</td>
<td>1</td>
<td>Integer</td>
<td>The time slot number, starting from 1.</td>
</tr>
<tr>
<td>Power</td>
<td>1</td>
<td>String: &quot;&lt;integer&gt;W&quot;</td>
<td>The maximum power available in the time slot, in watts.</td>
</tr>
<tr>
<td>renewablePercentage</td>
<td>1</td>
<td>String: &quot;&lt;integer&gt;%&quot;</td>
<td>The percentage of power produced from renewable sources</td>
</tr>
<tr>
<td>carbonEmissions</td>
<td>1</td>
<td>String: &quot;&lt;number&gt;g/(kWh)&quot;</td>
<td>The carbon emissions for producing energy in the time slot, in g/kWh.</td>
</tr>
</tbody>
</table>

Table 6: Forecast API - Input parameters

**Response body**

The output contains the energy forecast for the requested time period. For each timeslot a set of forecasts is provided:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeSlotDuration</td>
<td>The length of the timeslots. The total length of the interval must always be a multiple of the time slot duration.</td>
</tr>
</tbody>
</table>

Table 7: Forecast API - Output parameters

V.1.2. DC Controller Power Plan API

This API can be used by the ERDS or EMA to get the data centre power plan. This allows a higher level of planning and optimization.
GET [DCControllerUr]/v1/datacenters/{dataCenterName}/powerplan

URL parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Cardinality</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCControllerUr</td>
<td>1</td>
<td>URL</td>
<td>The URL of the DC controller</td>
</tr>
<tr>
<td>dataCenterName</td>
<td>1</td>
<td>String</td>
<td>The name of the data centre</td>
</tr>
</tbody>
</table>

Table 8: DC Controller Power Plan API - URL parameters

Response body

The output contains the current power plan of the data centre.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Cardinality</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$input</td>
<td>1</td>
<td>Object</td>
<td></td>
</tr>
<tr>
<td>dateFrom</td>
<td>1</td>
<td>ISO-8601</td>
<td>The start date and time of the interval for which the forecast is requested</td>
</tr>
<tr>
<td>dateTo</td>
<td>1</td>
<td>ISO-8601</td>
<td>The end date and time of the interval for which the forecast is requested</td>
</tr>
<tr>
<td>timeSlotDuration</td>
<td>1</td>
<td>String: &quot;&lt;integer&gt;min&quot;</td>
<td>The duration of the time slots in which the interval must be divided.</td>
</tr>
<tr>
<td>powerQuotas</td>
<td>1..n</td>
<td>Array</td>
<td>An item for each time slot in the interval, with the power quota for that time slot.</td>
</tr>
<tr>
<td>timeslot</td>
<td>1</td>
<td>Integer</td>
<td>The time slot number, starting from 1.</td>
</tr>
<tr>
<td>power</td>
<td>1</td>
<td>String: &quot;&lt;integer&gt;W&quot;</td>
<td>The maximum power available in the time slot, in watts.</td>
</tr>
</tbody>
</table>

Table 9: DC Controller Power Plan API - Output parameters
V.2. Energy Adaptive Software Control Interface

V.2.1. EASC OptionPlan API

Ask for an EASC Power Plan Options

This is the protocol specification of the step 5 described in section III.2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Cardinality</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eascName</td>
<td>1</td>
<td>String</td>
<td>The name of the EASC.</td>
</tr>
<tr>
<td>dateFrom</td>
<td>1</td>
<td>Date</td>
<td>Start date and time of the time frame.</td>
</tr>
<tr>
<td>dateTo</td>
<td>1</td>
<td>Date</td>
<td>End date and time of the time frame.</td>
</tr>
<tr>
<td>timeSlotDuration</td>
<td>1</td>
<td>Duration</td>
<td>Duration of time slots within the time frame.</td>
</tr>
<tr>
<td>DCPowerQuotas</td>
<td>1..n</td>
<td>Array</td>
<td>A list of power quotas for each data centres.</td>
</tr>
<tr>
<td>DCName</td>
<td>1</td>
<td>String</td>
<td>The name of data centre.</td>
</tr>
<tr>
<td>powerQuotas</td>
<td>1..n</td>
<td>Array</td>
<td>A list of power quotas</td>
</tr>
<tr>
<td>timeslot</td>
<td>1</td>
<td>Integer</td>
<td>Time slot number.</td>
</tr>
<tr>
<td>power</td>
<td>1</td>
<td>Number</td>
<td>Power quota for the time slot</td>
</tr>
</tbody>
</table>

Table 10: EASC OptionPlan API - Input parameters

In a federated mode, the EASCs controlling applications receive power quotas corresponding to DCs in which support EASC application federation.

The EASC then schedule the “work left to do” according to the federation type (the federation type is configured in the EASC configuration file). As the application controlled by the EASC can span several data centres, some working modes will be composed of resources located in the various data centres. This is why, in the outputted option plan, we include the estimated power consumed by each work in the various data centres composing the federation.

Output:
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Cardinality</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eascName</td>
<td>1</td>
<td>String</td>
<td>The name of the EASC, as registered in the Asset inventory. For example: easc.iaas1.</td>
</tr>
<tr>
<td>activities</td>
<td>1..n</td>
<td>Array</td>
<td>A list of Activity.</td>
</tr>
<tr>
<td>activityName</td>
<td>1</td>
<td>String</td>
<td>The name of the activity executed by EASC.</td>
</tr>
<tr>
<td>optionPlans</td>
<td>1..n</td>
<td>Array</td>
<td>The list of option plan. Only one of the list will be selected to become an ActivityPlan</td>
</tr>
<tr>
<td>optionPlanName</td>
<td>1</td>
<td>String</td>
<td>The name of option plan.</td>
</tr>
<tr>
<td>workOptionList</td>
<td>1..n</td>
<td>Array</td>
<td>List of Work Option that represents the list of possible planned works.</td>
</tr>
<tr>
<td>workID</td>
<td>1</td>
<td>String</td>
<td>An ID used to create the selected Work item.</td>
</tr>
<tr>
<td>timeRanges</td>
<td>1..n</td>
<td>Array</td>
<td>List of possible time ranges. Only one of the list will be selected to be assigned to the created Work.</td>
</tr>
<tr>
<td>startAt</td>
<td>1</td>
<td>Date</td>
<td>Start of time frame. Define when the work can start.</td>
</tr>
<tr>
<td>endAt</td>
<td>1</td>
<td>Date</td>
<td>End of time frame. Define when the work can end.</td>
</tr>
<tr>
<td>powers</td>
<td>1..n</td>
<td>Number</td>
<td>The list of estimated power consumption of the work option in each data centre</td>
</tr>
<tr>
<td>DCName</td>
<td>1</td>
<td>String</td>
<td>The name of the DC in which the power is consumed</td>
</tr>
<tr>
<td>power</td>
<td>1</td>
<td>Number</td>
<td>The estimated power consumption of the work option in the data centre</td>
</tr>
<tr>
<td>workingMode</td>
<td>1</td>
<td>String</td>
<td>The working mode that will be executed by the EASC</td>
</tr>
<tr>
<td>greenPoints</td>
<td>1</td>
<td>Number</td>
<td>The number of green points that will be assigned to this work option if will be selected</td>
</tr>
</tbody>
</table>

Table 11: EASC OptionPlan API - Output parameters
V.2.2. EASC ActivityPlan API

Get the current EASC Activity Plan

<table>
<thead>
<tr>
<th>Method</th>
<th>URI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>[EASC]/v1/easc/{eascname}/activityplan</td>
<td>Return the current ActivityPlan for the specified {eascname} resource</td>
</tr>
</tbody>
</table>

Output

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Cardinality</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eascName</td>
<td>1</td>
<td>String</td>
<td>The name of the EASC, as registered in the Asset inventory. For example: easc.iaas1.</td>
</tr>
<tr>
<td>activities</td>
<td>1..n</td>
<td>Array</td>
<td>A list of Activity items.</td>
</tr>
<tr>
<td>activityName</td>
<td>1</td>
<td>String</td>
<td>The name of the activity executed by EASC.</td>
</tr>
<tr>
<td>activityPlan</td>
<td>1</td>
<td>Object</td>
<td>The activity plan executed by EASC.</td>
</tr>
<tr>
<td>activityPlanName</td>
<td>1</td>
<td>String</td>
<td>The name of the activity plan.</td>
</tr>
<tr>
<td>worklist</td>
<td>1..n</td>
<td>Array</td>
<td>A list of Work that represent the activity plan.</td>
</tr>
<tr>
<td>workID</td>
<td>1</td>
<td>String</td>
<td>The Work Id.</td>
</tr>
<tr>
<td>startAt</td>
<td>1</td>
<td>Date</td>
<td>Start of the work timeslot.</td>
</tr>
<tr>
<td>endAt</td>
<td>1</td>
<td>Date</td>
<td>End of the work timeslot.</td>
</tr>
<tr>
<td>workingMode</td>
<td>1</td>
<td>String</td>
<td>The working mode identifier planned for this work timeslot</td>
</tr>
</tbody>
</table>

Table 12: EASC ActivityPlan API – Activity Plan type

Execute an EASC Activity Plan

This is the protocol specification of the step 8 described in section III.2.

<table>
<thead>
<tr>
<th>Method</th>
<th>URI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUT</td>
<td>[EASC]/v1/easc/{eascname}/activityplan</td>
<td>Execute the ActivityPlan, and make it the current, for the specified {eascname} resource</td>
</tr>
</tbody>
</table>

Input

Object of Activity Plan type (see Table 12)
VI. DEVELOPMENT & TESTING

VI.1. Development

The consortium set up a development environment that is suitable to work collaboratively over multiple components. Except for the coding task itself, most of the tasks related to the development are performed automatically by multiple dedicated services. This provides a reliable environment with numerous tools to help developers with writing a good and maintainable code over the whole project duration.

![Development Methodology Diagram]

Figure 38 depicts the development methodology:

1. Each authorised user gets the source code from our private Source Code Manager (SCM), develops some code, tests and documentation;
2. Once the development is done, the resulting updates are send back to the SCM that notifies all the developers about its arrival;
3. The SCM builds the code and executes the unit tests to check for potential failures or regressions. It then notifies all the developers accordingly;
4. The Continuous Integration (CI) server requests a finer grain analysis of the code quality. This analysis generates a report about the current software status with regards to established quality metrics (coding style, API documentation, code complexity, dependencies) and potential bugs. This report is made available to every authorised user;
5. When the source code compiles and the unit tests pass, the resulting artifact is labelled and stored inside a private repository that is available to every authorised user. The API documentation is also generated and made available on a private Web server.

According to our methodology, we choose to rely on the following software stack to manage our development. This stack has been selected according to the area of expertise of the developers, but also the partners who will administrate these services.

- Java as a primary language for development. All the developers know this language. Scala will be a secondary language for some developments as it has the advantage of being fully compatible with the java byte code.
- Maven⁶ to control the development lifecycle. This tool is open source and is known by a large majority of the developers involved in the consortium. It is also compatible with

⁶ [http://maven.apache.org](http://maven.apache.org)
Integrated Development Environment of every user. Its usage will formalise the development process and will provide a common base to manage the software lifecycle.

- Git as a SCM and Gitlab\(^7\) to manage multiple Git repositories. Both are open-source. Gitlab is an active project that is very convenient to create repositories on demand. Some developers in the consortium are not familiar with Git but the administrators and the other partners provide documentation and tutorials to assist them. To avoid developing a massive monolithic repository, it was decided to rely on multiple ones. In practice, each software component that has its own lifecycle will have its dedicated repository.

- Jenkins\(^8\) as a CI server. Jenkins is open source and provides standard plugins to interact with Git, Gitlab, and Maven projects.

- Sonar\(^9\) to analyse the source code quality against common development rules for Java projects.

Gitlab, Jenkins and Sonar are hosted and maintained inside a private virtual machine at Inria Sophia Antipolis. Credentials limit the access to the members of the consortium. Data are replicated locally on multiple disks and mirrored every night to the Inria Lille.

VI.2. Testing

VI.2.1. Issue management in the codebase

Unit tests are written to test for code correctness. They are usually written on the fly during the development phases. Each time a bug is detected by someone in a code he/she is not responsible for, an issue is created and assigned to the most appropriate developer, usually the one that wrote the code. The resolution progress is tracked and once the bug is solved, the assignee integrates his/her patch inside the branch that contains the code under development.

During the trials, a few bugs were discovered. These bugs were critical as they required to restart the trials and were not trivial to fix. To minimize the chances of having an improper resolution that will invalidate several days of trial, we changed our development workflow to be sure the issue was fixed properly before re-launching the trials. When the issue is detected, the assignee fixes the bug in a dedicated branch. Once the bug is marked as resolved, a different developer reviews the patch. Once he acknowledges the fix, the patch is merged into the codebase. This process improved our efficiency at fixing bugs in critical periods.

VI.2.2. Testing with a simulator

During the first trial phases, it became apparent that it is hard to quickly test and estimate how the central system and the EASCs will perform in reality. Unit tests are appropriate to check for the code quality but they are not well suited to check software performance especially in a dynamic environment where the optimisation is performed over multiple days instead of a single timeslot.

It was also difficult to estimate how DC4Cities would perform on a large-scale infrastructure. For phase 1, the central system only manages one EASC running on a few servers. It is however important to observe DC4Cities benefits in a larger data centre, typically running hundreds or thousands of EASCs on a similar amount of servers.

\(^7\) http://www.gitlab.org
\(^8\) http://jenkins.org
\(^9\) http://www.sonarqube.org/
It was decided to implement a simulator for the central system and the EASCs. Its purpose is to simulate a test bed environment as realistic as possible to be able to quickly check the software behaviour and performance without having to run it for a long period on real hardware. As we only want to test the DC4Cities software stack, we decided to simulate only the applications controlled by the EASCs. In practice, each time an EASC wants to start a new working mode for the underlying application, we simulate this call and consider the switch to be completed immediately.

Concisely, the simulator runs as fast as possible the optimisation loop of the central system for every timeslot we want to simulate. Because we want to have simulation results as accurate as possible, we decided to run the largest amount of "production code" possible. Currently, the simulator runs the production code for the optimization process, the power splitter, the option consolidator, the working mode managers, and the EASCs. The only components that use a mock implementation are those devoted to the communication with the EASCs and the working mode actuator. The first component is not critical and we think its behaviour can be checked efficiently with unit tests only. In contrast, the working mode actuator is a critical component that should not use a mock implementation, however its current implementation does not allow for an integration into the simulator. For the next phase, we plan to re-implement it to fix that issue.

The simulator should be able to reproduce the trials, but also to enable testing in large scale infrastructures not available in the DC4Cities trials. We make them it configurable accordingly. Currently, the following configuration changes are possible:

- Change the number of power sources, their nature in terms of production capabilities, and level of renewable fraction;
- Change the number of running EASCs and their nature (either the production code for the trial EASCs or artificial ones);
- Change the optimisation goals;
- Change the simulation period.

The simulator output is a collection of statistics gathered at each time slot. These statistics allow the developer to understand which events occurred during the simulation phase. Currently the following statistics are reported:

- The working mode selected for each EASC;
- The energy consumption for each source;
- The EASC performance using the business metrics;
- The solving duration for the option plan consolidator.

We already use the simulator to replay the trials as they were performed during phase 1. A comparison against the practical results confirmed the high accuracy of the simulator. Aside, we observed the simulator was useful to make quick tests. Table 13 shows the acceleration we observed when we replay the trials on a Macbook with a 3GHz Intel core i7. On average, it is possible to replay the trials 29,000 times faster, which allows for an early phase testing before the practical evaluation.

<table>
<thead>
<tr>
<th></th>
<th>Real duration</th>
<th>Simulated time</th>
<th>Acceleration factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trento</td>
<td>6 days + 1 timeslots</td>
<td>15.7 sec.</td>
<td>33,076</td>
</tr>
<tr>
<td>CSUC</td>
<td>4 days + 57 timeslots</td>
<td>11.5 sec.</td>
<td>34,513</td>
</tr>
<tr>
<td>HP</td>
<td>5 days + 1 timeslot</td>
<td>21.5 sec.</td>
<td>20,134</td>
</tr>
</tbody>
</table>

Table 13: Solving duration for the optimizer when replaying trial experiments
The simulator was also used to evaluate the performance of the option plan consolidator. Firstly because its performance where not measurable for the trials. Secondly because the consolidator aims at solving an NP-hard problem\textsuperscript{10} and its scalability had to be evaluated.

Table 14 shows the average solving duration of the option plan consolidator while it was replaying the trials. We first observed that the solving phase is negligible compared to the duration of a timeslot (0.028\% of a timeslot). Second, we observed the option plan consolidator was always capable of proving the optimality of the computed solution. This ensures that there is no consolidation that would lead to a more effective usage of energy.

<table>
<thead>
<tr>
<th>Solving duration</th>
<th>Trento</th>
<th>253ms</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSUC</td>
<td>187ms</td>
<td></td>
</tr>
<tr>
<td>HP</td>
<td>195ms</td>
<td></td>
</tr>
</tbody>
</table>

Table 14: Solving duration for the optimizer when replaying trial experiments

Finally, we tested its scalability by varying the number of running EASCs from 1 to 1000. Table 15 shows the results.

<table>
<thead>
<tr>
<th>600 EASCs</th>
<th>800 EASCs</th>
<th>1000 EASCs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trento</td>
<td>0.5s</td>
<td>0.7s</td>
</tr>
<tr>
<td>CSUC</td>
<td>0.7s</td>
<td>0.6s</td>
</tr>
<tr>
<td>HP</td>
<td>7.6s</td>
<td>16.1s</td>
</tr>
</tbody>
</table>

Table 15: Solving duration for the optimizer against the number of EASCs

We observed the solving duration takes up to 3\% of a timeslot when we simulate a data centre running 1000 HP EASCs. With regards to the current trial setup at HP, this test simulates a Moonshot\textsuperscript{11} cluster of 29,000 cartridges, which can be considered a large-scale data centre. The overhead is still less than 1\% when we run 600 EASCs, which corresponds to a medium-scale data centre.

### VI.3. Demo Support

The simulator appears to provide a good backend for a demo of DC4Cities. Our objective is to enhance it to allow creating a proper demo that would be able to show in 30 seconds the daily benefits of DC4Cities when it is run on top of average-size data centre. We are currently investigating the integration of a production ready monitoring system to improve the demo realism but also how to integrate the dashboard we developed.

The demo will be distributed as a virtual appliance that is executable through an open-source virtualisation solution which can be installed easily on traditional operating systems.

\textsuperscript{10} http://en.wikipedia.org/wiki/NP-hard
\textsuperscript{11} http://www8.hp.com/us/en/products/servers/moonshot/
VII. APPENDIX A: LOW LEVEL USE CASES

VII.1. Configuration

<table>
<thead>
<tr>
<th>Use Case ID:</th>
<th>UC2a</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name:</td>
<td>Enter Technical Configuration</td>
</tr>
<tr>
<td>Created By:</td>
<td>Torben Möller</td>
</tr>
<tr>
<td>Last Updated By:</td>
<td>Torben Möller</td>
</tr>
<tr>
<td>Date Created:</td>
<td>10/03/2015</td>
</tr>
<tr>
<td>Date Last Updated:</td>
<td>18/03/2015</td>
</tr>
</tbody>
</table>

| Actors: | • DC4Cities Energy Admin (DC4EA)  
• DC Business Manager (DCBM)  
• Energy Management Authority (EMA)  
  o Smart City Context: Smart City Energy Management Authority (EMA-SC)  
• Energy Adaptive Software Controller (EASC)  
|   |
| Description: | Configure the technical aspect of the DC. |
| Trigger: | • Installation/first configuration of the DC4Cities system.  
• Every time DC4EA/EMA needs to change or check the DC Energy Controller system configurations, depending on their role. |
| Preconditions: | • DC4Cities system is up and running  
• The Configuration Dashboard is reachable (several views provided to DC actors and EMA)  
• Actors have been authenticated and their roles allow to access one of the below features  
• ERDSs of DC are registered inside the DC4Cities system (cf. UC1) |
| Postconditions: | • DC is ready to be optimised to reach the energy objectives. |
| Normal Flow: | 1. DC4EA/EMA accesses the Configuration Dashboard and chooses the technical configuration.  
2. DC4EA sets DC profile attributes (static data like: name, type, location…). This information is used for communication purposes towards external actors (through a dashboard) or external system (smart city) in order to identify from which DC information comes.  
3. DC4EA sets/disCOVERs electricity supply channel(s) for its DC, see UC1 (local, grid, micro grid).  
4. DC4EA configures and tests the channels to acquire energy supply data (past, current, future=prediction models in place).  
5. DC4EA describes the IT service types hosted in the DC. (Purpose: allow the EMA to set realistic objectives and the DC has been fully configured. See further. Note: details of DC ICT services are NOT configured by DC4EA.)  
6. DC4EA sets up the configuration for each Energy Adaptive Software Controllers (EASCs) present in his/her DC (tasks, working modes…). Moreover, DC4EA also sets up the SLAs.  
7. DC4EA configures and tests the communication with the EASCs to acquire energy consumption data (past, current, future = power prediction models in place) and to exchange optimisation requests. |
8. DC4EA configures energy optimization process parameters (time frequency, enable/disable status, etc.).
9. DC4EA confirms to save any introduced configuration changes. The system checks/validates data consistency/completeness and applies it.
10. EMA and DC4EA sees the DC appearing in its energy management dashboard.
11. Use case UC2b will come next.

<table>
<thead>
<tr>
<th>Alternative Flows</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.a</td>
<td>Test of supply channel fails, inform DC4EA.</td>
</tr>
<tr>
<td>7.a</td>
<td>Test of communication with EASC fails, inform DC4EA.</td>
</tr>
<tr>
<td>9.a</td>
<td>Checks/validations of data consistency/completeness fail: system informs DC4EA accordingly.</td>
</tr>
<tr>
<td>15.a</td>
<td>Technical or Energy Configuration is not properly set: system informs DC4EA and does not allow enabling the optimization process.</td>
</tr>
</tbody>
</table>

Exceptions: -

Includes:
- UC4: Compute and Distribute Power Budgets
- UC7: Set Objectives

Priority: Medium

Frequency of Use: One per Day / One per Week

Use Case ID: UC2b

Use Case Name: Enter Energy Configuration

Created By: Torben Möller  Last Updated By: Torben Möller

Date Created: 10/03/2015  Date Last Updated: 18/03/2015

Actors:
- DC4Cities Energy Admin (DC4EA)
- DC Business Manager (DCBM)
- Energy Management Authority (EMA)
- Smart City Context: Smart City Energy Management Authority (EMA-SC)
- Energy Adaptive Software Controller (EASC)

Description: Configure the technical aspect of the DC.

Trigger:
- Installation/first configuration of the DC4Cities system. After
- Every time DC4EA/EMA needs to change or check the DC Energy Controller system configurations, depending on their role.

Preconditions:
- DC4Cities system is up and running
- The Configuration Dashboard is reachable (several views provided to DC actors and EMA)
- Actors have been authenticated and their roles allow to access one of the below features
- ERDSs of DC are registered inside the DC4Cities system (cf. UC1)

Postconditions:
- DC is ready to be optimised to reach the energy objectives.
Normal Flow:

1. EMA (/DC4EA) views a dashboard showing historical supply vs. consumption data and details (service types, supply channels, etc.) about the DC. This allows the EMA to set realistic objectives for this DC.

2. EMA, as part of its overall energy strategy, sets realistic **Power/Energy Objectives** for the DC (e.g. run with 80% RE during weekdays 8am – 6pm): UC7: Set Objectives

3. DCBM(/DC4EA) controls IT services power budget quota allocation/distribution by setting the corresponding **Power Splitting Policy** (e.g. weighted power split function with weight parameter set for each IT Services, cf. UC4: Compute and Distribute Power Budgets).

4. EMA saves objectives and DCBM saves its power splitting policies. This ends the **Energy Configuration**.

5. Now the DC is ready to be optimised to reach these objectives.

6. DC4EA does a last check with the EMA/DCBM about the changed configuration. Once agreed, the DC4EA can enable the “optimisation” process (cf. UC1/UC3/UC4/UC5), based on the new configuration. This is only possible when both technical and energy configuration are correctly set.

Alternative Flows:

7.a Test of communication with EASC fails, inform DC4EA.

9.a Checks/validations of data consistency/completeness fail: system informs DC4EA accordingly.

15.a Technical or Energy Configuration is not properly set: system informs DC4EA and does not allow enabling the optimization process.

Exceptions: -

Includes: • UC4: Compute and Distribute Power Budgets
  • UC7: Set Objectives

Priority: Medium

Frequency of Use: One per Day / One per Week

**VII.2. Ideal Power Plan**

**Use Case ID:** UC3a

**Use Case Name:** Request Energy Forecast

<table>
<thead>
<tr>
<th>Created By:</th>
<th>Torben Møller</th>
<th>Last Updated By:</th>
<th>Torben Møller</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date Created:</td>
<td>10/03/2015</td>
<td>Date Last Updated:</td>
<td>18/03/2015</td>
</tr>
</tbody>
</table>

**Actors:**

- D4C Control System (CTRL)
- DC4Cities Energy Subsystem (DC4ES)
- DC Business Manager (DCBM)
- Energy Management Authority (EMA)

  - Smart City Context: Smart City Energy Management Authority (EMA-SC)
<table>
<thead>
<tr>
<th>Description:</th>
<th>Query energy forecasts.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trigger:</td>
<td>- Time interval (every 15 minutes, for instance)</td>
</tr>
</tbody>
</table>
| Preconditions: | - System properly configured and optimisation process enabled (see UC2: Set/Check Configurations)  
| | - Energy forecast or EMA’s desired power plan available (see UC1: Monitor/Predict Energy Availability)  |
| Postconditions: | - Energy forecast are available. |
| Normal Flow: | 1. The CTRL requests (Renewable) Energy Forecasts for each of the DC’s electricity supply channel(s) from the DC4ES (cf. UC1: Monitor/Predict Energy Availability) for a given time frame.  
| | 2. For given time frame, Power/Energy Objectives are queried from the current Energy Configuration. |
| Alternative Flows: | - |
| Exceptions: | Sources for Energy forecasts are not available. |
| Includes: | - UC1: Monitor/Predict Energy Availability |
| Priority: | High |
| Frequency of Use: | Regular |

---

| Use Case ID: | UC3b |
| Use Case Name: | Compute Ideal DC Power Plan |
| Created By: | Torben Möller | Last Updated By: Torben Möller |
| Date Created: | 10/03/2015 | Date Last Updated: 18/03/2015 |
| Actors: | - D4C Control System (CTRL)  
| | - DC4Cities Energy Subsystem (DC4ES)  
| | - DC Business Manager (DCBM)  
| | - Energy Management Authority (EMA)  
| | o Smart City Context: Smart City Energy Management Authority (EMA-SC) |
| Description: | Computes ideal power plan of the DC based on energy forecasts and configured power/energy objectives for a given time frame (e.g. 24h). |
| Trigger: | - UC3a is completed |
| Preconditions: | - Energy forecast are available. |
| Postconditions: | - The ideal power plan of the DC has been computed for a given time frame. |
| Normal Flow: | 1. Based on retrieved energy forecasts and objectives, the CTRL computes the ideal DC power for each time slot from now to the next 24h (i.e. with a time frame = 15 min => 96 total slots). |
2. In detail, calculations are done considering energy forecasts or desired power draw from the grid requested by EMA at time slot T and Power/Energy objectives as constraints. The result is the MAX DC total power for each slot, without violating any power/energy budget objectives.

3. CTRL proceeds with the computation and distribution of power budgets taken from the computed power plan: UC4: Compute and Distribute Power Budgets

### Alternative Flows:
- 

### Exceptions:
- 3.a Computed ideal DC power plan (or subset of it) violate power/energy objectives (short-term violation). Both DC4EA and EMA see a warning in their energy dashboards. However, it is important to note that EMA is mainly interested in information about long-term violations of objectives.

### Includes:
- UC1: Monitor/Predict Energy Availability

### Priority:
- High

### Frequency of Use:
- Regular

## VII.3. Power Budgets

<table>
<thead>
<tr>
<th>Use Case ID:</th>
<th>UC4a</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case Name:</td>
<td>Compute Power Budgets</td>
</tr>
<tr>
<td>Created By:</td>
<td>Torben Möller</td>
</tr>
<tr>
<td>Date Created:</td>
<td>10/03/2015</td>
</tr>
<tr>
<td>Last Updated By:</td>
<td>Torben Möller</td>
</tr>
<tr>
<td>Date Last Updated:</td>
<td>18/03/2015</td>
</tr>
</tbody>
</table>
| Actors: | • Energy Adaptive Software Controller (EASC)  
• D4C Control System (CTRL)  
• DC4Cities Energy Subsystem (DC4ES) |
| Description: | Computes and distributes power budgets from current ideal DC power plan to all EASCs. |
| Trigger: | • Time interval (every 15 minutes, for instance) |
| Preconditions: | • Ideal DC Power Plan computed (see UC3: Compute Ideal DC Power Plan) |
| Postconditions: | • The power budgets are distributed among all EASCs. |
| Normal Flow: | 1. The CTRL retrieves actual Ideal DC Power Plan (see UC3: Compute Ideal DC Power Plan). |
| Alternative Flows: | - |
| Exceptions: | - |
| Includes: | - |
| Priority: | - |
| Frequency of Use: | - |
### Use Case ID: UC4b

**Use Case Name:** Distribute Power Budgets

<table>
<thead>
<tr>
<th>Created By:</th>
<th>Torben Möller</th>
<th>Last Updated By:</th>
<th>Torben Möller</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date Created:</td>
<td>10/03/2015</td>
<td>Date Last Updated:</td>
<td>18/03/2015</td>
</tr>
</tbody>
</table>

**Actors:**
- Energy Adaptive Software Controller (EASC)
- D4C Control System (CTRL)
- DC4Cities Energy Subsystem (DC4ES)

**Description:** Computes and distributes power budgets from current ideal DC power plan to all EASCs.

**Trigger:**
- Time interval (every 15 minutes, for instance)

**Preconditions:**
- Ideal DC Power Plan computed (see UC3: Compute Ideal DC Power Plan)

**Postconditions:**
- The power budgets are distributed among all EASCs.

**Normal Flow:**
1. The CTRL distributes Ideal Power budget from current DC power plan to each EASC by enacting **Power Splitting Policies** set in **Energy Configuration** (see UC2: Set/Check Configurations).
2. The calculation result is: For each EASC: Max power "quota" per time slot (e.g. time frame = 15mins, next 24 hours)
3. The CTRL sets the ideal power budget for each EASC.
4. The sequence process with UC5: Determine Applications Working Mode by sending power budgets and constraints to each corresponding EASC to determine the applications working mode.

**Alternative Flows:**
- 

**Exceptions:**
- 

**Includes:**
- 

**Priority:**
- 

**Frequency of Use:**
- 

### VII.4. Escalation

**Use Case ID:** UC8a

**Use Case Name:** Escalation

<table>
<thead>
<tr>
<th>Created By:</th>
<th>Torben Möller</th>
<th>Last Updated By:</th>
<th>Torben Möller</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date Created:</td>
<td>10/03/2015</td>
<td>Date Last Updated:</td>
<td>18/03/2015</td>
</tr>
</tbody>
</table>

**Actors:**
- DC4Cities Energy Admin (DC4EA)
- DC Business Manager (DCBM)
- Energy Management Authority (EMA)
  - Smart City Context: Smart City Energy Management Authority (EMA-SC)
- Energy Adaptive Software Controller (EASC)

**Description:** Informs the contact person about occurring problems with the objectives.
**Trigger:**
- A problem with the objectives has occurred.

**Preconditions:**
- An objective cannot be accomplished

**Postconditions:**
- User is informed

**Normal Flow:**
1. All necessary information is gathered.
2. Multiple solution options are generated
3. The user is informed that a problem has occurred.
4. The information is shown in the dashboard.
5. The user handles the error 8 UC8b

**Alternative Flows:**
- 

**Exceptions:**
- 

**Includes:**
- 

**Priority:** High

**Frequency of Use:** -

---

**Use Case ID:** UC8b

**Use Case Name:** Escalation Handling

**Created By:** Torben Möller  **Last Updated By:** Torben Möller

**Date Created:** 10/03/2015  **Date Last Updated:** 18/03/2015

**Actors:**
- DC4Cities Energy Admin (DC4EA)
- DC Business Manager (DCBM)
- Energy Management Authority (EMA)
  - Smart City Context: Smart City Energy Management Authority (EMA-SC)
- Energy Adaptive Software Controller (EASC)

**Description:**
The user handles the escalation

**Trigger:**
- The user is informed via the dashboard.

**Preconditions:**
- UC8a

**Postconditions:**
- The problem is solved

**Normal Flow:**
1. The EMA-SC takes one of the solution options
2. Federated data centres are asked for help, by migrating the option to them.
3. The work of the chosen solution option is migrated.

**Alternative Flows:**
- 

**Exceptions:**
- No federated data centre can help
- Work is not migratable

**Includes:**
- 

**Priority:** High

**Frequency of Use:** -
VIII. APPENDIX B: METRICS CATALOGUE

The metric catalog contains the list of metrics used in the context of DC4Cities. The catalog contains for every metric its label and a description of the metric, the category the metric belongs to, the way the metric can be aggregated, its unit, the asset types for which the metric can be measured and the purposes for which it can be used. Centralising all metrics in a catalog makes the metrics used explicit, brings consistency and avoids using similar or duplicated metrics.

Metrics included in the catalog can be classified in the following categories:

- **Power/Energy metrics**: in order to execute the global DC power plan as well as to check if the goals have been achieved, it is needed to measure the power consumption of the different components controlled by DC4Cities. Moreover, if the analysis needs to be done at DC level, power consumption of auxiliary systems as cooling systems or UPS has to be also monitored.

- **Load metrics**: in order to forecast the energy consumptions of the components controlled by DC4Cities and therefore to distribute the power quotas to each EASC, it is needed to obtain data about the work performed by each process in execution. In addition, workload metrics are needed to evaluate performances.

- **KPIs**: key performance indicators have to be continuously calculated in order to have information about the accomplishment of the contractual goals and, from a general perspective, the energetic, economic and environmental behaviour of the DC.

To structure the metrics which are used in DC4Cities a catalog of metrics has been defined. A metric definition has following attributes:

- **metric name**: This name must be unique. It is the name which is used in the Historical database.

- **label**: Used to show the metric to the user. Currently the labels are in English.

- **aggregation**: The function to use to aggregate metric values in time and in space.

- **description**: Textual description of the metric.

- **unit**: Unit of the metric

- **reference**: The same metric can be used for different purposes. The “current” metric gives the real measured value. The “forecasted” metric is a prediction some time ahead of what will be the metric value. The “estimated” metric is a value estimated from other related metrics.

- **asset type**: The type of assets for which this metric can be measured or computed and thus is relevant. The choice can be “site”, “server”, “ERDS”, “VM”, “WM”.

The Table 16 contains the metrics catalog that has been defined in the DC4Cities Control System for the first phase of the trials:

<table>
<thead>
<tr>
<th>Metric Name</th>
<th>Label (en)</th>
<th>Category</th>
<th>Aggregation</th>
<th>Description</th>
<th>Unit (SI)</th>
<th>Asset type</th>
</tr>
</thead>
<tbody>
<tr>
<td>power</td>
<td>Power consumed by data center, server</td>
<td>Energy</td>
<td>AVG</td>
<td>Power consumed by data center, server</td>
<td>W</td>
<td>site, server, erds, vm, wm</td>
</tr>
<tr>
<td>renewable_power</td>
<td>Renewable Power</td>
<td>Energy</td>
<td>AVG</td>
<td>renewable Power</td>
<td>W</td>
<td>x</td>
</tr>
<tr>
<td>renewable_energy</td>
<td>Renewable Energy</td>
<td>Energy</td>
<td>SUM</td>
<td>renewable Energy</td>
<td>Wh</td>
<td>x</td>
</tr>
<tr>
<td>renewable_energy_percentage</td>
<td>Renewable Percentage for grid</td>
<td>Energy</td>
<td>AVG</td>
<td>Time dependent Renewable grid percentage</td>
<td>%</td>
<td>x</td>
</tr>
<tr>
<td>ambient_temperature</td>
<td>Ambient Temperature</td>
<td>Energy</td>
<td>AVG</td>
<td>Ambient Temperature</td>
<td>°C</td>
<td>x</td>
</tr>
<tr>
<td>irradiation</td>
<td>Radiation</td>
<td>Energy</td>
<td>SUM</td>
<td>Radiation</td>
<td>Wh/m2</td>
<td>x</td>
</tr>
<tr>
<td>influence</td>
<td>Influence</td>
<td>Energy</td>
<td>SUM</td>
<td>Influence</td>
<td>Wh/m2</td>
<td>x</td>
</tr>
<tr>
<td>pluse</td>
<td>Power Usage Efficiency</td>
<td>Energy</td>
<td>AVG</td>
<td>the ratio between TOTAL facility energy / IT equipment energy</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>carbon_emission_value</td>
<td>Carbon Emission Factor</td>
<td>Energy</td>
<td>AVG</td>
<td>Carbon Emission Factor</td>
<td>g/kWh</td>
<td>x</td>
</tr>
<tr>
<td>cpu_usage</td>
<td>CPU</td>
<td>Load</td>
<td>AVG</td>
<td>The cpu usage of the physical server. It is the average cpu usage of all the logical cores of the server.</td>
<td>%</td>
<td>x</td>
</tr>
<tr>
<td>memory_usage [*]</td>
<td>Memory</td>
<td>Load</td>
<td>SUM</td>
<td>Memory usage of the physical server in GB</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>disk_read [*]</td>
<td>Disk</td>
<td>Load</td>
<td>SUM</td>
<td>nr of disk reads per sec</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>disk_write [*]</td>
<td>Disk</td>
<td>Load</td>
<td>SUM</td>
<td>nr of disk writes per sec</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>net_read [*]</td>
<td>Network</td>
<td>Load</td>
<td>SUM</td>
<td>nr of disk packets read per sec</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>net_write [*]</td>
<td>Network</td>
<td>Load</td>
<td>SUM</td>
<td>nr of disk packets sent per sec</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>datacenter_number_of_virtual_cores</td>
<td>Number of Virtual Cores</td>
<td>Load</td>
<td>SUM</td>
<td>number of running virtual machines per DC</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>datacenter_number_of_vms</td>
<td>Number of VMs</td>
<td>Load</td>
<td>SUM</td>
<td>number of running virtual machines per DC</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>server_number_of_virtual_cores</td>
<td>Runtime Number of Virtual Cores alocated</td>
<td>Load</td>
<td>SUM</td>
<td>number of running virtual machines per server</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>server_number_of_vms</td>
<td>Runtime Number of Vms</td>
<td>Load</td>
<td>SUM</td>
<td>number of running virtual machines per server</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>sum_vcpu_usage</td>
<td>Sum of Virtual CPU usage</td>
<td>Load</td>
<td>AVG</td>
<td>The sum of the cpu usage percentage for all the Vcores</td>
<td>%</td>
<td>x</td>
</tr>
<tr>
<td>avg_vcpu_usage</td>
<td>Avg of Virtual CPU usage</td>
<td>Load</td>
<td>AVG</td>
<td>The average cpu usage percentage for the Vcores</td>
<td>%</td>
<td>x</td>
</tr>
<tr>
<td>nr_of_active_servers</td>
<td>Number of Active Servers</td>
<td>Load</td>
<td>SUM</td>
<td>number of running servers</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>server_runtime_core_ratio</td>
<td>max nrOfVirtualCores/nrOfPhysicalCore ratio per server</td>
<td>Load</td>
<td>AVG</td>
<td>Current value for max_core_ratio (defines how many vCores can be allocated at this moment)</td>
<td>ratio/r</td>
<td>x</td>
</tr>
<tr>
<td>bspecf²_Items</td>
<td>Number of actual items</td>
<td>Load</td>
<td>SUM</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>bspecf²_Items_rate</td>
<td>Number of actual items / time</td>
<td>Load</td>
<td>AVG</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>renewable_utilisation</td>
<td>Renewable Energy over Total Energy</td>
<td>KPI</td>
<td>AVG</td>
<td>DC consumed using renewable energy when this energy is available. When Pren &gt; Pdc use Pdc and when Pdc &gt; Pren use Pren</td>
<td>%</td>
<td>x</td>
</tr>
<tr>
<td>co2emissions</td>
<td>CO2 Emissions</td>
<td>KPI</td>
<td>SUM</td>
<td>CO2 emissions</td>
<td>kg</td>
<td>x</td>
</tr>
<tr>
<td>ren_percent</td>
<td>Percentage of Renewable Energy used by Data Center</td>
<td>KPI</td>
<td>AVG</td>
<td>Percentage of Renewable Energy used by Data Center</td>
<td>%</td>
<td>x</td>
</tr>
<tr>
<td>dca</td>
<td>Data Center Adaptiveness</td>
<td>KPI</td>
<td>AVG</td>
<td>Data Center Adaptiveness</td>
<td>ratio/r</td>
<td>x</td>
</tr>
<tr>
<td>apc</td>
<td>Adaptability Power Curve</td>
<td>KPI</td>
<td>AVG</td>
<td>Adaptability Power Curve</td>
<td>ratio/r</td>
<td>x</td>
</tr>
</tbody>
</table>

* = postponed

Table 16: Metric Catalog