Integrated ICT-platform based Distributed Control in electricity grids with a large share of Distributed Energy Resources and Renewable Energy Sources

Reference ICT System Architecture Guidelines

Deliverable D9.2

Identifier:	INTEGRAL_D9.2
Date:	28-02-2011
Class:	Deliverable
Responsible Partners:	NTUA/ICCS, IDEA, BTH, KEMA, INPG, HUMIQ
Annexes:	
Distribution:	PU
Overview:	This document describes the reference ICT architecture for Integral

This project is funded by the European Commission Under the 6th Framework Programme (Project FP6-038576)

The INTEGRAL consortium consist of:

ECN	Principal Contractor & Coordinator	The Netherlands
NTUA/ICCS	Principal Contractor	Greece
IDEA	Principal Contractor	France
Blekinge University of Technology	Principal Contractor	Sweden
KEMA	Principal Contractor	The Netherlands
WattPic Intelligent	Principal Contractor	Spain
EnerSearch AB	Principal Contractor	Sweden
INPGrenoble	Principal Contractor	France
HUMIQ	Principal Contractor	The Netherlands
CRIC	Principal Contractor	Spain
Essent	Principal Contractor	The Netherlands

Control Versions:

Version	Date	Author	Description of Changes
0.1	26-04-2010	Jörgen van der Velde	Initial draft
0.2	05-07-2010	Jörgen van der Velde	State diagram added
0.3	04-08-2010	Jörgen van der Velde	Sequences added, update after review
Final	28-02-2011	Jörgen van der Velde	Final Version

Table of Contents

List of Figures

References

[UML]	The Unified Modelilng Language User Guide Booch, Rumbaugh, Jacobson ISBN 0-201-57168-4
[4+1]	Architectural Blueprints – The “4+1” view model of software architecture’ Phillippe Kruchten http://www.cs.ubc.ca/~gregor/teaching/papers/4+1view-architecture.pdf
[DoW]	Integral Annex I – “Description of Work”

Acronyms and Abbreviations

CA	Commercial Aggregator
DER	Distributed Energy Resources
DSO	Distirbution System Operator
HTTP	HyperText Transfer Protocol
HV	High voltage
MDA	Model Driven Architecture
LV	Low Voltage
MV	Mid Voltage
RES	Renewable energy sources
SCADA	Supervisory Control And Data Acquisition
SOAP	Simple Object Access Protocol
TDD	Test Driven Development
TSO	Transmission System Operator
XML	eXtended Markup Language

Executive Summary

This document describes the architectural model underlying the three field tests. It presents different views on the system architecture. It provides a mapping of all three experiments on this architecture.

1. Introduction

1.1 Goal

The goal of this document is to present the ICT architecture of a system that enables scenarios as have been executed during the three Integral field tests.

During the Integral project three rather separate field trials have been executed. The architecture and design of the systems required have been described in the deliverables of Work Package 5, 6 and 7. This document presents the ICT architecture at a higher level. This document is the result of integration of the individual architectures.

1.2 Scope

This document applies to the Integral project. It is meant to:

- Report on the findings during the Integral field tests
- Present guidelines to other 'smart grid' experiments, pilots and roll-outs

Therefore, the intended readership of this document is

- Integral project members
- The EU Officer
- Members of related 'smart grid' projects

This document is the second deliverable of four of Work Package 9 (see [DoW]).
The first deliverable (D9.1) focuses on the lessons learned from the field tests.
The second deliverable (D9.2, this one) focuses on the common architecture underlying the field tests.
The third deliverable (D9.3) focuses on the data and information model.
The fourth deliverable (D9.4) presents a functional and technical description of the agents used

2. Conventions and definitions

2.1 System architecture vs. software architecture

Modern ICT systems, whether it is a mobile phone or a 'smart grid', are often very complex. Such a system contains many hundreds of thousands lines of software, often written by multiple parties. Creating such a system cannot be managed unless its structure is well chosen and standards and guidelines are set in place defining the development process.

In order to describe how an ICT system must be build, it is common to describe its *architecture*. The main reason for having an architectural description is to cope with complexity. The main strategy of an architecture is 'divide and conquer':

A system architecture describes

- the main components in a system (including their responsibilities) and how they interact.
- major design decisions and rules and guidelines to apply
- the rationale for the decisions

The scope of an architecture is not only the creation of the system, but the entire lifecycle of the system. The reason is that after creation, systems are maintained and often modified.

It is important to discern *system architecture* and *software architecture*.
System architecture describes the system, its hardware and connections. It identifies which entities are part of the system to be created, and which are not.
Software architecture describes the system software

2.2 Modelling language

In order to describe an architectural model, a common language is required amongst all partners. For this the Unified Modelling Language (UML, see [UML]) will be used.

Not all views in the architectural model can be described by this language. For these views alternative symbols may be used. However, these views must be self-explanatory, e.g. by the use of a legend.

2.3 Views

An architectural model consists of a number of views. Each view focuses on one aspect of the system or software. Together these views make up the architecture.

A common approach for describing software architecture is the 4+1 view method [4+1]. The software architecture is described in 5 views.

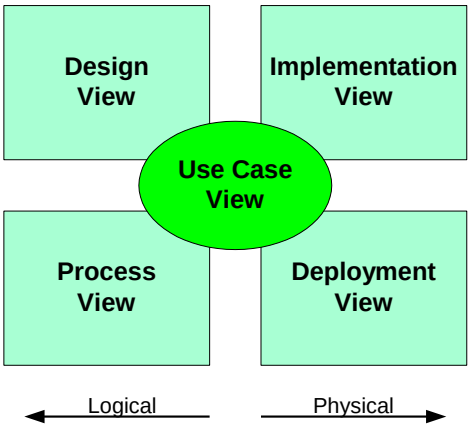


Figure 1 The 4+1 view method

The model defines 4 views

- The Design View, which is the logical model of the design
- The Process View, which captures the concurrency and synchronisation aspects of the design,
- The Implementation View, which shows the software components to be assembled and released,
- The Deployment View, which describes the nodes and hardware topology and how software is deployed on this.

The Use Case View is the starting point for the other views and describes how the System behaves from a perspective of its end users.

3. System architecture

3.1 Use Case view

The Use Case view defines the Use Cases of the system. A Use Case defines a scenario from the user perspective, showing how the system is used.

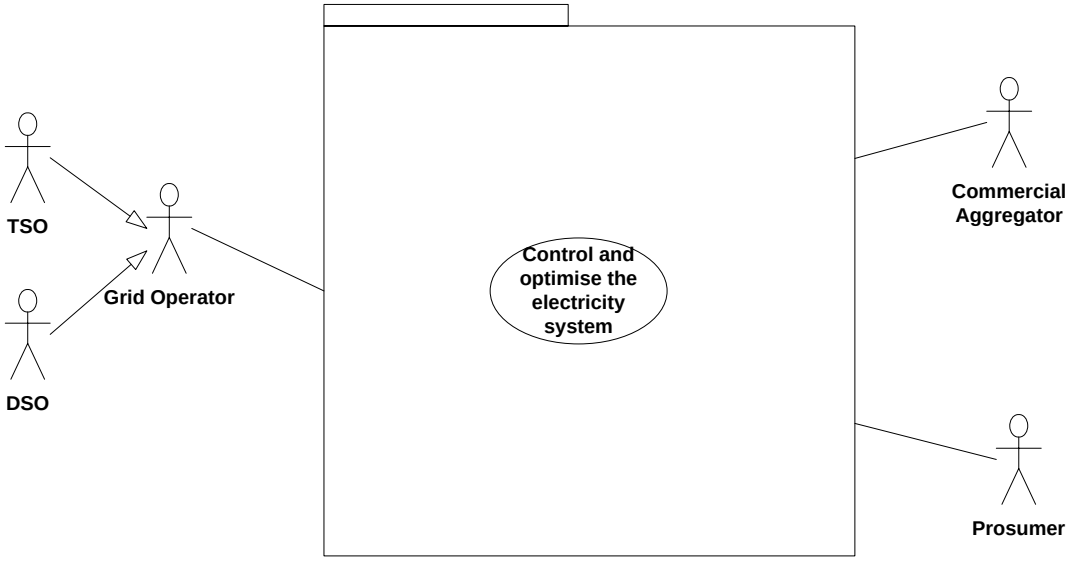


Figure 2 Highest level use case

Figure 2 shows the use case on highest level : 'Control and optimise the electricity systems'. It refers to keeping the electricity system up and running and optimising it. It represents the goal that benifits all stakeholders.

The diagram shows following stakeholders :

Stakeholder	Description
Prosumer	The Producing Consumer. The Consumer not only consumes electricity, but may also generate electricity. The Prosumer wants to minimise costs for energy and maximise income from own generated electricity
Commercial Aggregator	The Commercial Aggregator delivers or buys electricity to/from the Prosumer. The Commercial Aggregator trades on energy markets. The Commercial Aggregator benefits from following delivery/consumption contracts, reducing own contribution to the imbalance in the grid.
Grid Operator	The Grid Operator owns the assets for transport and distribution of electricity. The Grid Operator benefits from correct usage of the assets (flat loads, not overloading, etc). We discern Transmission System Operators (TSO)

and Distribution System Operators (DSO)

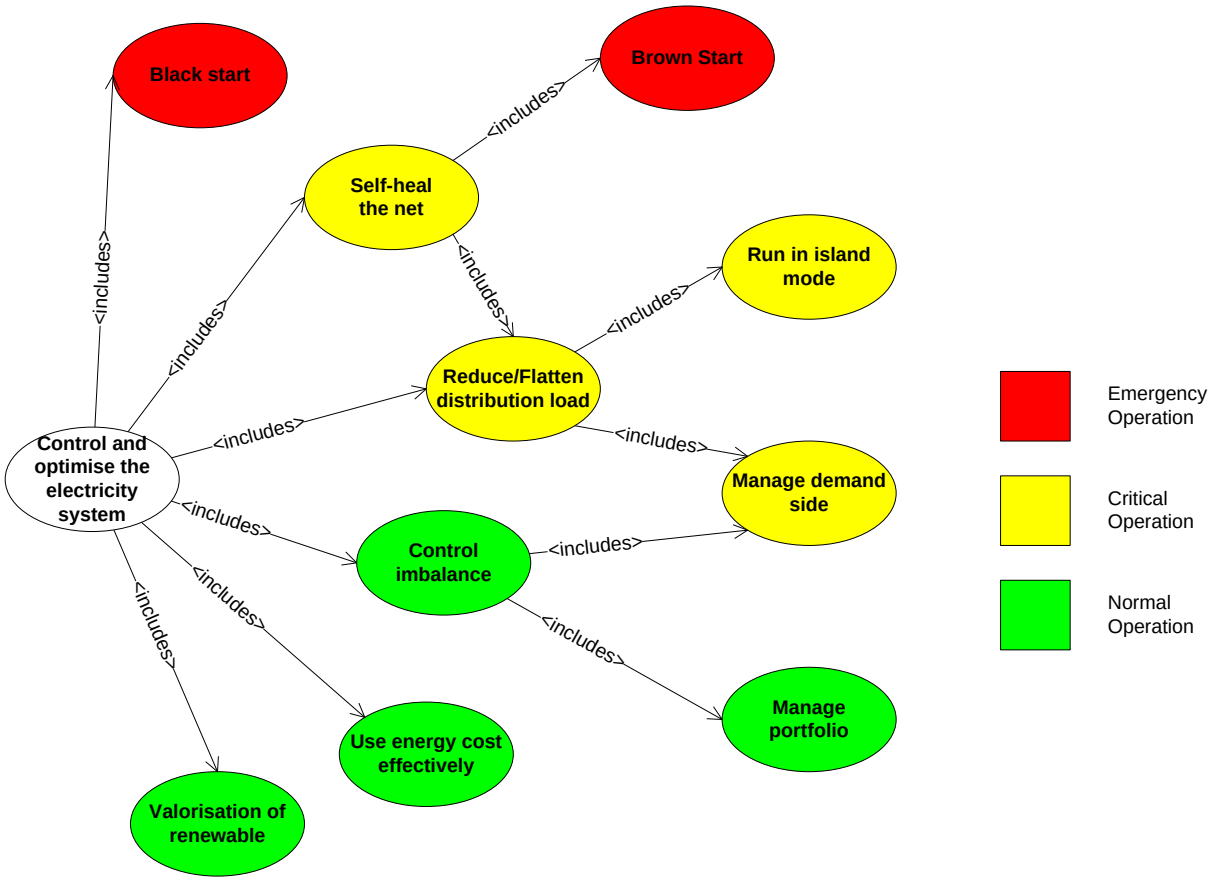


Figure 3 High level use cases

Use Case	Description
Black start	Start the system after an exogenous black-out.
Brown start	Start the system after an endogenous black-out.
Self-heal the net	If a problem occurs in the grid, this use case describes how the problem is solved. Identification, locating and isolation the fault. Rearranging of the grid and bringing up the good parts of the grid are part of this use case.
Run in island mode	Run a level 0 cell in island mode, for example during a black-out.
Reduce/flatten distribution load	This refers to using the distribution grid resource as economically as possible by flatting the load and preventing overload.
Manage demand side	Refers to controlling the demand side on device level.
Control imbalance	Refers to reducing imbalance in the grid.

Manage portfolio	This use case refers to optimise exploitation of all generators and consumers in the portfolio of the CA. This may include clusters of distributed Energy Generators
Use energy cost effectively	This refers to using energy as cost efficiently as possible by the prosumer.
Valorisation of renewables	Use renewable energy as efficient as possible in terms of financial benefit and/or contribution to loss reduction.

By means of colours it is indicated to which operation mode the use case mainly applies.

3.2 Grid view

The Grid View defines the model of the grid. It is not a common ICT view, but more meant as context for the ICT views.

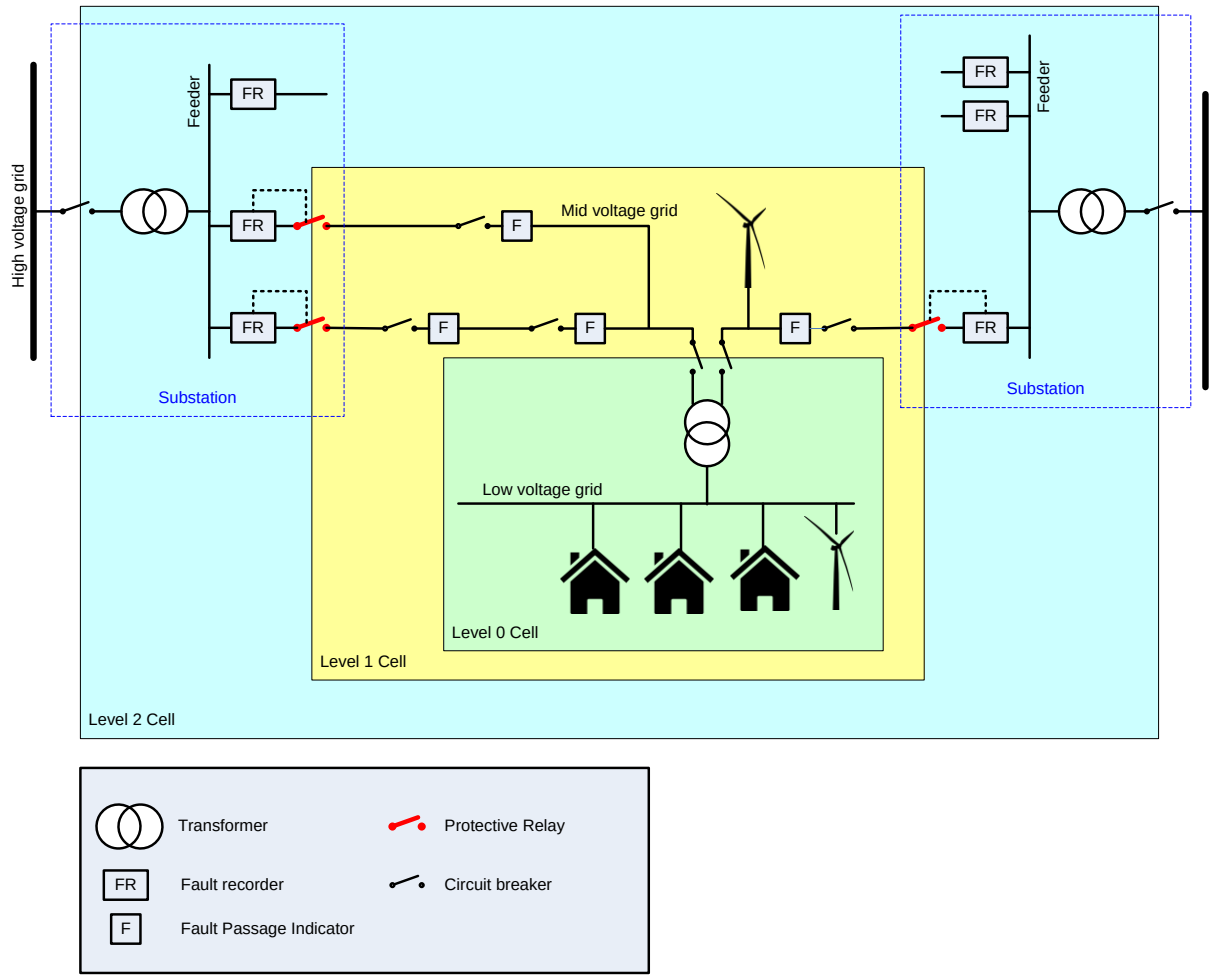


Figure 4 Grid view

Figure 4 shows presents an overview of the grid.

A similar cell structure as used in the Crisp project has been applied in this project. Following cell levels are defined :

- Level 0 cell or Micro-grid
  Low voltage grid behind one MV to LV transformer. To these sections households and LV RES/DER are attached.

- Level 1 cell
  Mid voltage grid consisting of all sections that can be interconnected using circuit breakers. The Level 1 cell can be fed by one or more substations. The boundary of the Level 1 cell are the protective relays through which the cell is fed.

- Level 2 cell
  Mid voltage grid consisting of all sections that can be interconnected using circuit breakers and the feeders in the substations feeding them.

For Integral higher grid levels as well as centralised generation are out of scope.

One Substation feeds several Level 1 Cells. Alternatively one Level 1 Cell may be fed by more than one Substation. By means of circuit breakers the Level 1 Cell is arranged in such a way that each Substation feeds only one part of the Level 1 Cell.

When a problem occurs in such a Level 1 Cell a protective relay may trip. The location of the fault can be detected by means of the fault passage indicators and fault recorders. By controlling the circuit breakers the fault can be isolated. The Level 1 Cell may be rearranged in such a way that as much of the Cell as possible may be powered up again.

4. Software architecture

4.1 Logical view

The Logical View presents the decomposition of the software in logical components and defines how the components interact.

4.1.1 Logical view of the software

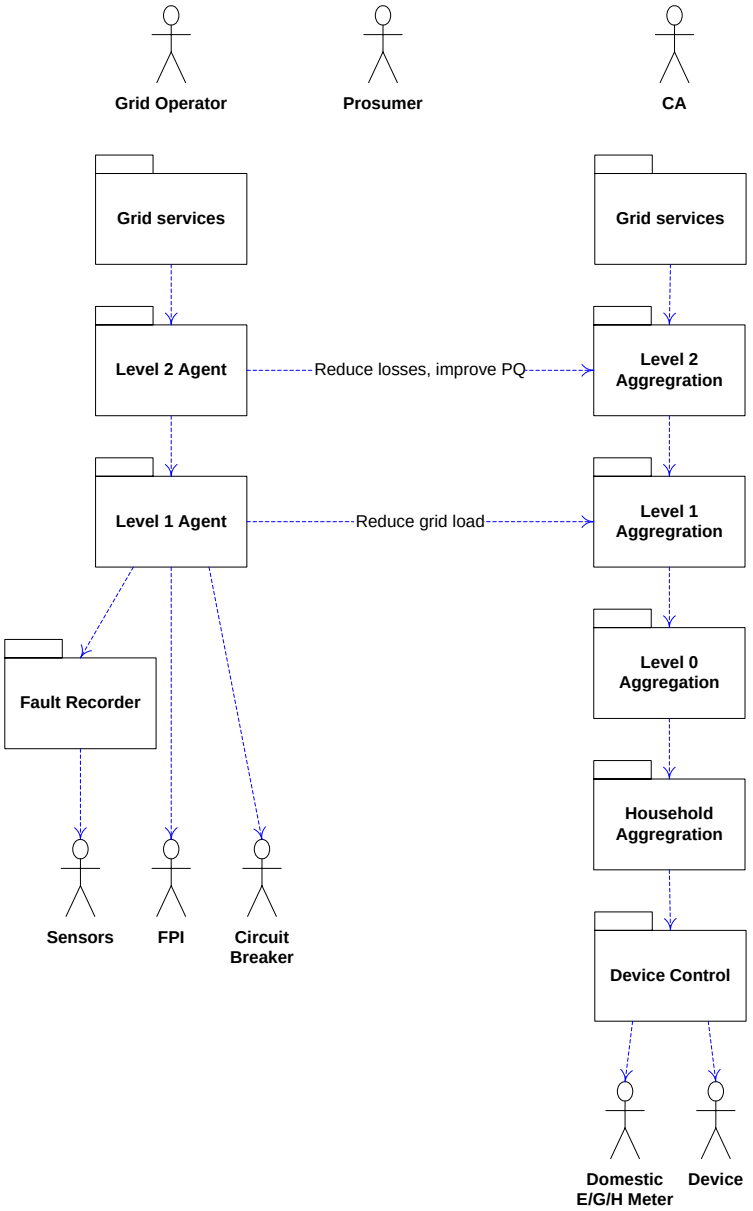


Figure 5 Logical view of the software

Roughly two vertical branches are visible. The left branch is involved in control of the grid and components in it. It is the domain of the SCADA systems of the grid operators. Processes at the lowest level take place on time scale of milliseconds to seconds.

The right branch is involved in controlling devices in households. Its aim is to implement the use cases like demand side management. Processes at the lowest level take place at time scale of seconds to minutes.

Following components are visible:

Component	Description
Fault recorder	The responsibility of this software is to monitor the grid and record information about the fault when it occurs.
Level 1 Agent	The responsibility of this software is to collect data from the Fault Recorder and Fault Passage Indicators and to rearrange the grid when needed (when a fault occurs). It can control circuit breakers for this.
Level 2 Agent	The responsibility of the Level 2 Agent software is to control the grid at Cell Level 2. The concern of this software is to maintain power quality and controlling the grid load.
Grid Services	Grid services offer management functionality to grid operator.
Device Control	The responsibility of this software is to monitor and control devices in the household. This includes consumer devices and meters.
Household Aggregation	The responsibility of this component is to aggregate information on the devices in the households and to present this information to the higher level aggregators.
Level 0 Aggregation	This component is responsible for aggregation of information on low voltage grid level (Level 0 Cell)
Level 1 Aggregation	This component is responsible for aggregation of information on medium voltage grid level (Level 1 Cell)
Level 2 Aggregation	This component is responsible for aggregation of information on the highest levels.
Grid Services	Grid Services is responsible for presenting information and management functionality to the stakeholders. For example user and operator web portals are found on this level.

4.1.2 States of the system

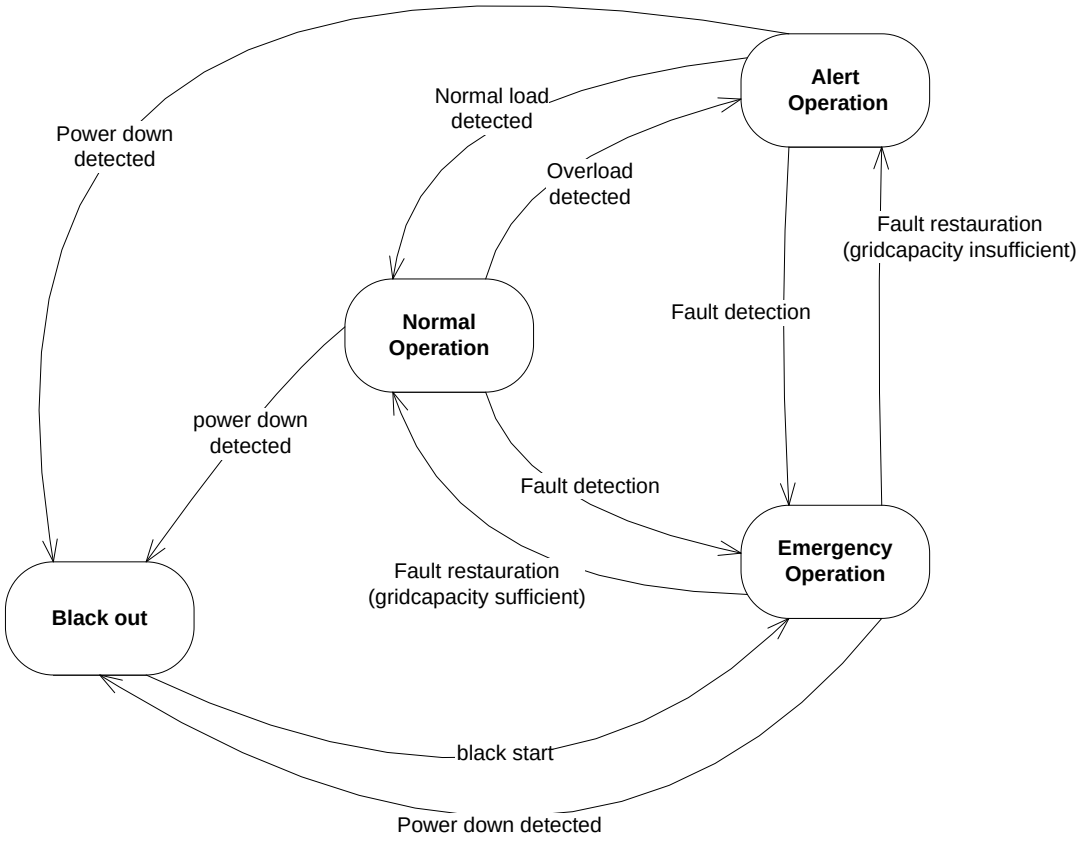


Figure 6 State diagram indicating the states of the system

Figure 6 shows the state diagram indicating the states of the system. Following states are visible:

State	Description
Normal Operation	Operation is normal. Loading of the grid is acceptable. • Load shedding is voluntary • Fastest processes: minutes • Prosumer is in charge
Critical Operation	The system works, but parts of the system are critically overloaded • Load shedding is forced and no longer voluntary • Fastest processes: seconds • DSO is in charge
Emergency Operation	The system is deteriorated. Full service is no longer guaranteed • Power loss, immediate reconfiguration of the grid is needed (brown out) • Fastest processes: milliseconds • DSO and TSO in charge Note: this state should be traversed fast

Black out	The system is off. No power is supplied

4.1.3 Dynamic behaviour

4.1.3.1Load curtailment

In this scenario the load on the grid (e.g. on a transformer) increases. It is reaching 100%. In order not to overload the grid, load is curtailed (voluntary load shedding in normal mode).

We first look how this is executed conceptually (Figure 7), then we translate this to how this is implemented in Demo A using PowerMatcher (Figure 8) and in Demo B using switching of loads (Figure 9).

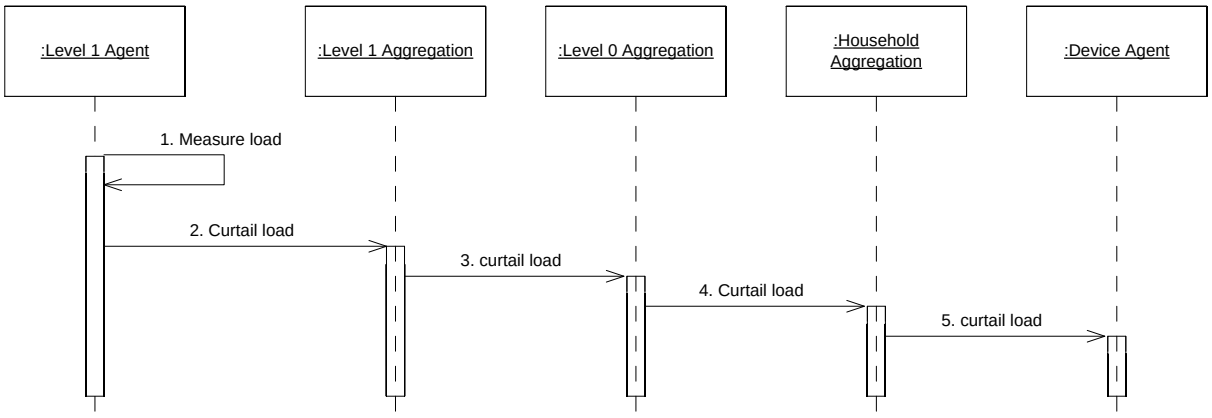


Figure 7 Curtailing of load

1	On the MV transformer the load is measured. The Level 1 Agent detects the load is increasing.
2-5	The Level 1 Agent issues a command for load curtailment. The command is delegated to the lower aggregation level, until the command reaches devices. Devices may respond by switching off.

In a similar way the :Level 1 Agent will stop the curtailment if situation has gone to normal.

Note : Having multiple levels in the system and passing messages over multiple levels is the way to cope with scalability. Via a few of these delegations many (tens of thousands of) devices can be reached. Broadcast mechanisms could be considered to reduce overhead time. However, this will require fully private networks, since these mechanisms are not possible on the internet.

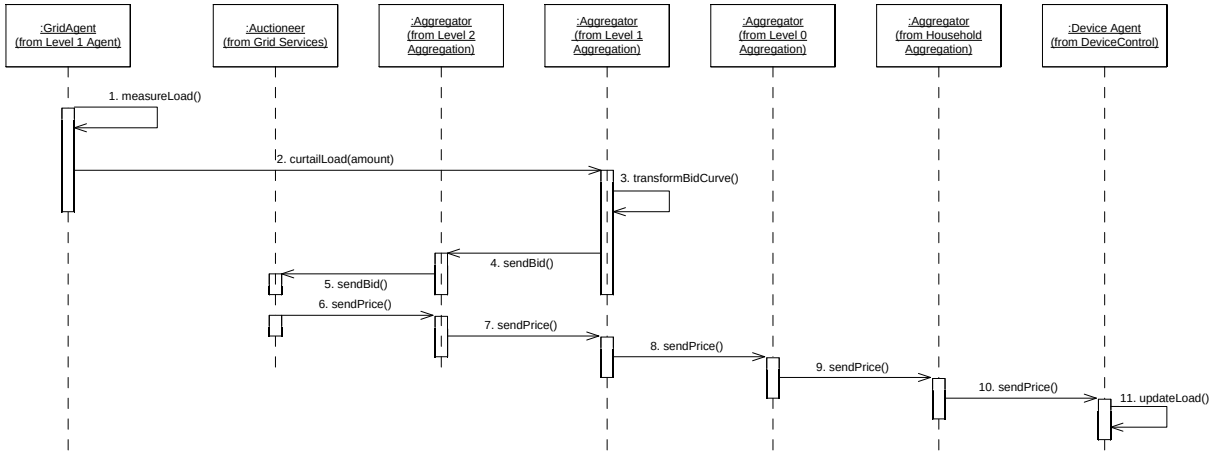


Figure 8 Load curtailment via the PowerMatcher concept in Demo A

1	The GridAgent at Level 1 grid measures the load and detects the load is increasing
2	The GridAgent issues a command to curtail the load
3	Load curtailment is implemented by means of transforming of the PowerMatcher bidcurve
4-5	The transformed bidcurve is send to the Auctioneer
6-11	If this leads to a new PowerMatcher equilibrium price, this price is communicated throughout the PowerMatcher system. Devices may respond by reducing their load (or equivalently, increasing their production)

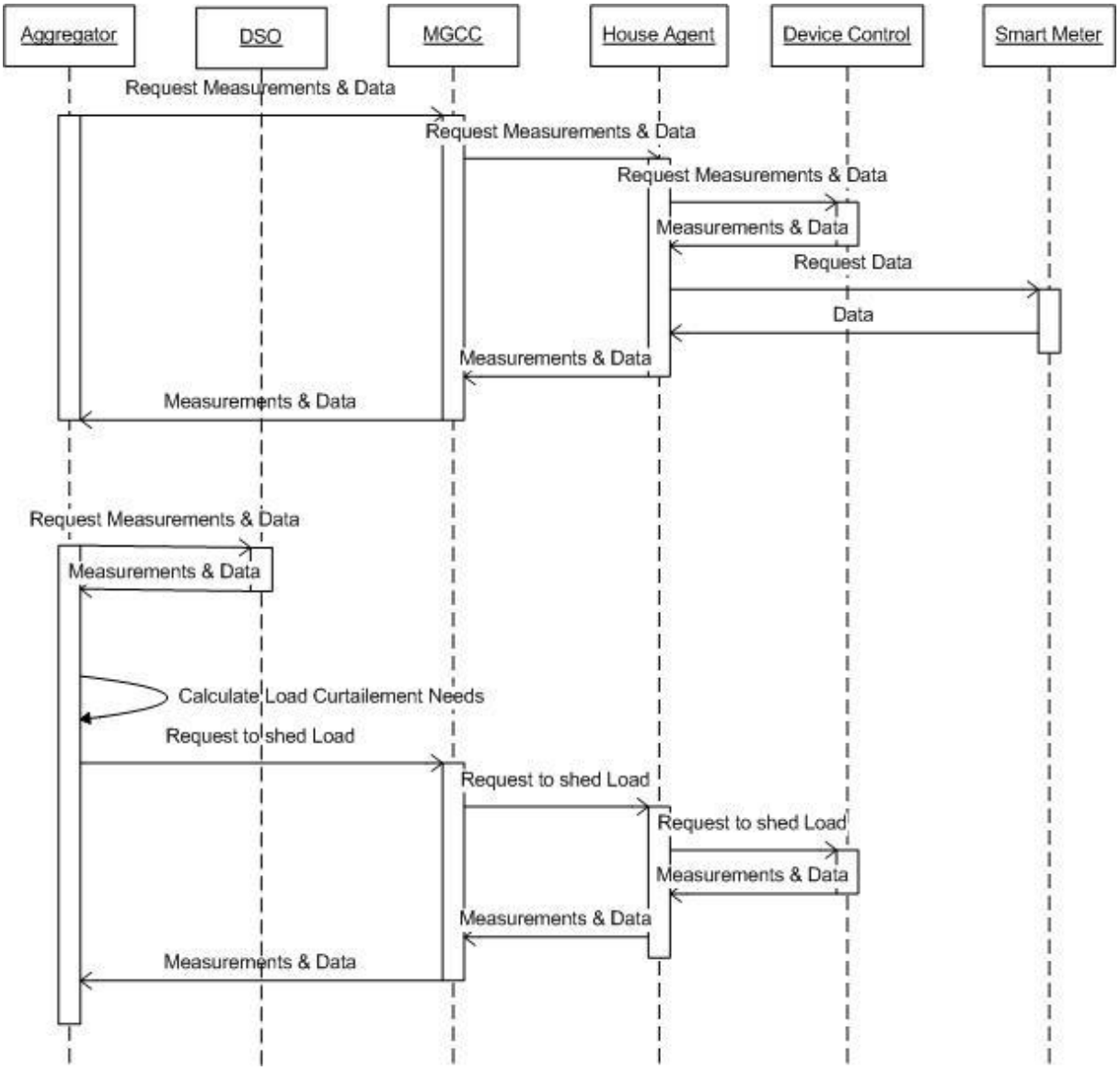


Figure 9 Load curtailment via the PowerMatcher concept in Demo B

1 The aggregator collects data from the households and the DSO

2 The aggregator sends Load Curtailement command to the MGCC

3 The MGCC sends Load Shedding command to the housholds.

4.1.3.2 Forced Load Shedding

As load increases beyond 100%, a state transition is made from Normal Operation to Alert Operation. Forces load shedding is required to prevent the grid from deterioration and power loss.

We first look at this conceptually (Figure 10), then we indicate how this can be implemented by means of the mechanism of Demo B (Figure 11).

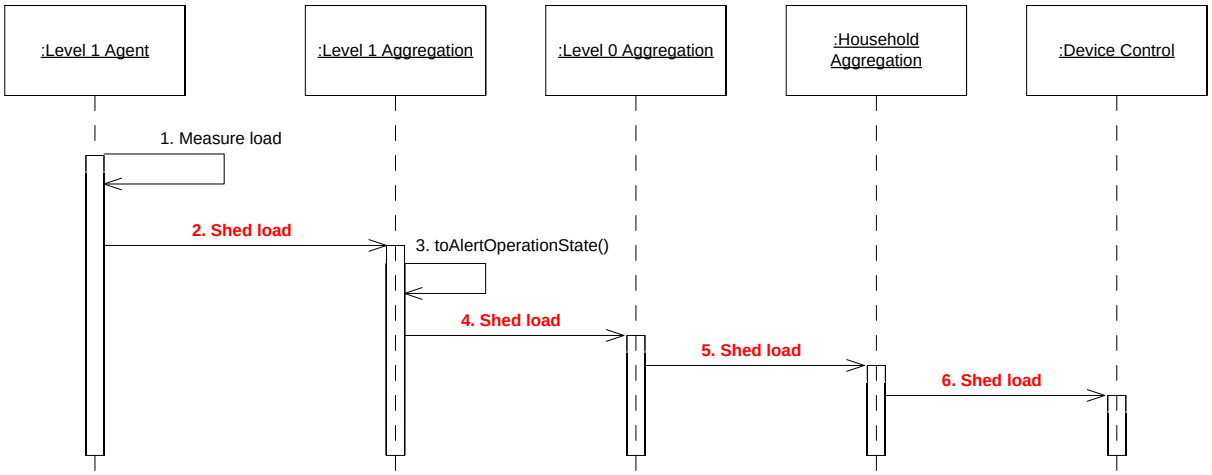


Figure 10 Forced Load Shedding, conceptually

1	On the MV transformer the load is measured. The Level 1 Agent detects the load is increasing beyond 100%.
2-5	The Level 1 Agent issues a command for load shedding. This implicitely means the Level 1 Agent declares the 'Alert Operation' state to the lower level. The command is delegated to the lower aggregation level, until the command reaches devices. Devices *must* respond by switching off.

Coordination in load shedding situations can be continued with a market-based approach as applied in Demo A. Devices can trade load shedding capabilities in addition to trading of the commodity. This concept will be elaborated in deliverable D9.4.

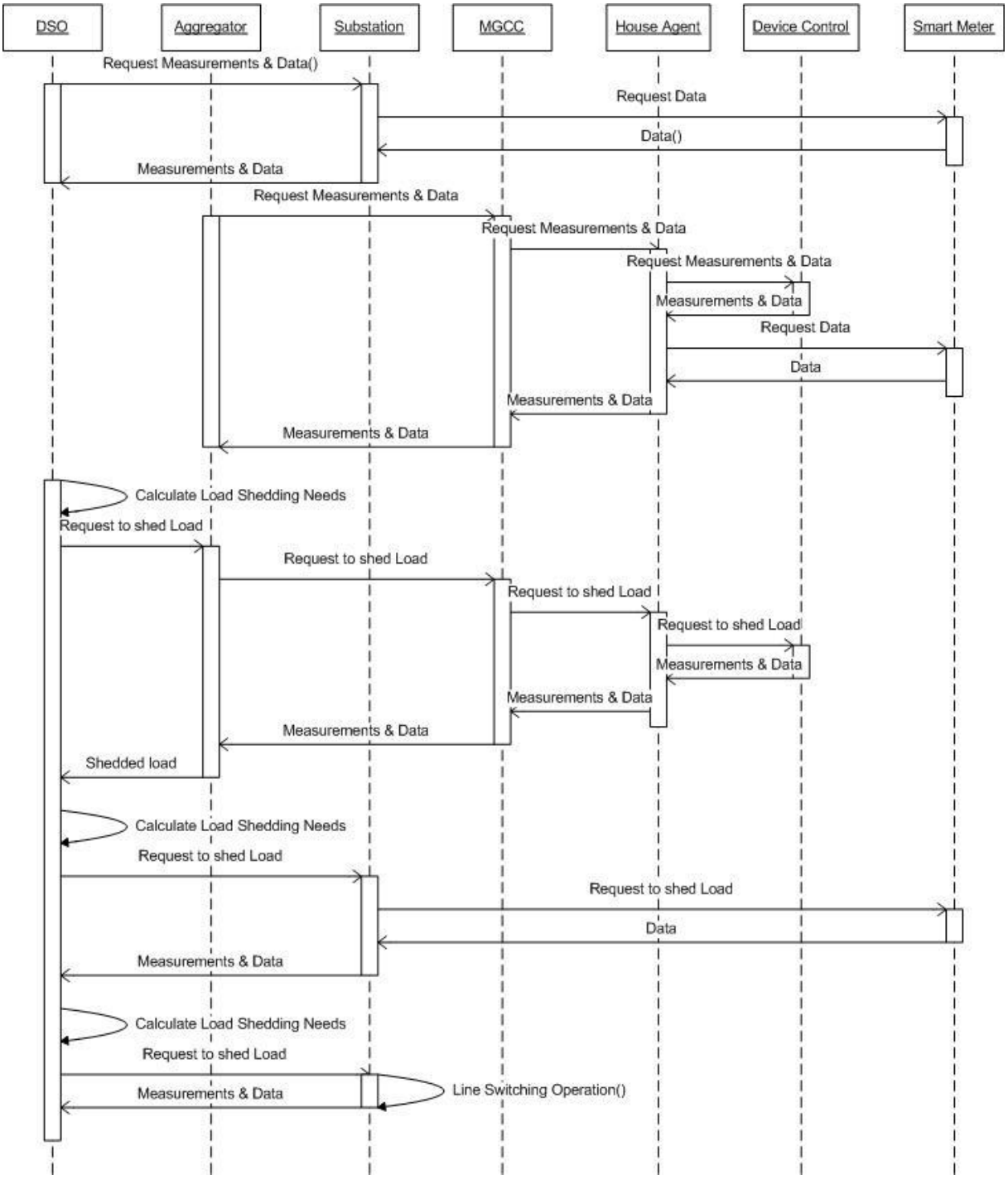


Figure 11 Load shedding as implemented in Demo B

1	The DSO Monitors the substations
2	The substation sends data from the smart meters and local measurements

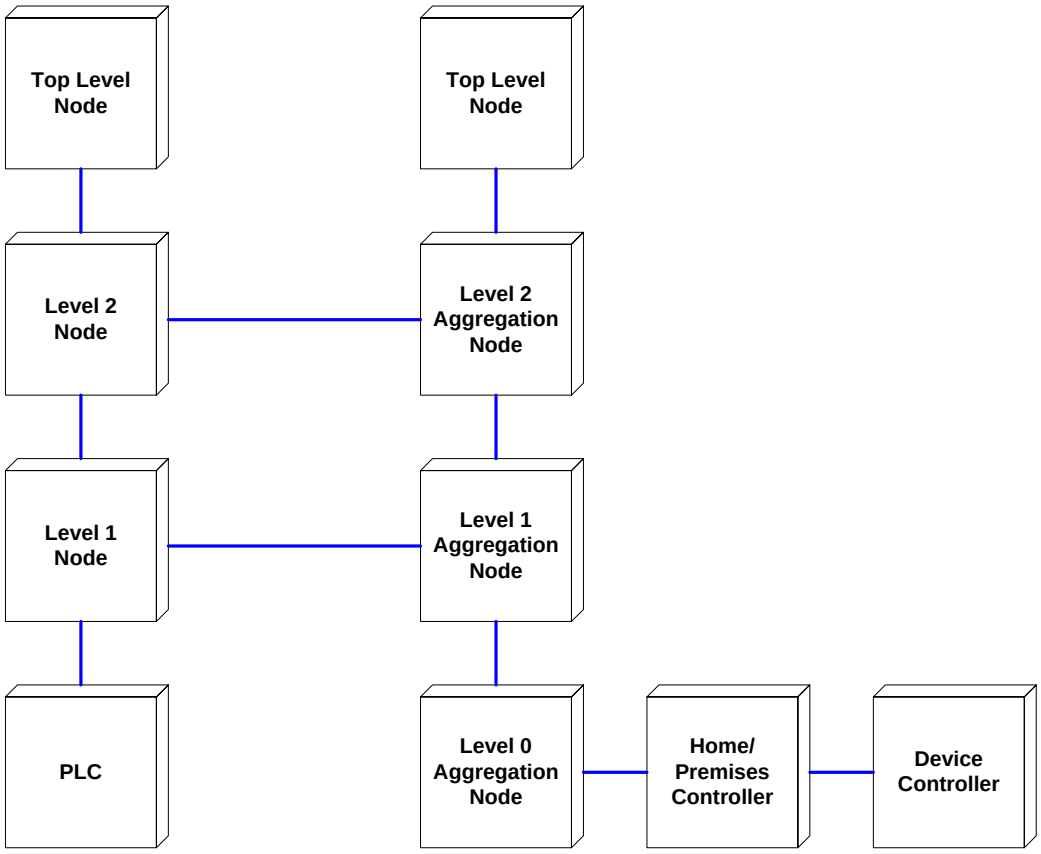
3	The aggregator collects data from the household. The house agent may (or not) have connection to the smart meter.
4	The DSO calculates the needs for Load shedding
5	The DSO sends command to the aggregator in order to shed load
6	If extra Load Shedding required the DSO sends command to the Substation to shut down households (via the smart meter)
7	If extra Load Shedding required the DSO sends command to the Substation to shut down feeders

4.2 Process view

The Process View presents the processes (and threads) that are executed in the software. It defines the means for inter-process communication.
Due to the abstract nature of this document this view is omitted. Refer to the individual architectural design documents in order to obtain information on the processes and thread used in the software.

4.3 Deployment view

The Deployment View presents the hardware (computational) nodes that are part of the system and responsible for executing the software.



4.4 Implementation view

The implementation view describes the 'physical' software entities to be delivered. In this document the Implementation view is omitted due to the abstract nature of this document. Furthermore, the actual Implementation View may depend on the actual roll-out of such a system.

5. Architectural guidelines

5.1 Decompose according to grid structure

The software architecture decomposition should adhere to the physical structure of the grid. The advantages are:

- This results in a clear subdivision of responsibility of software components
- In the grid as well as in the software aggregation levels are necessary. In the grid these levels are the cell levels from high voltage to low voltage, in the software the aggregation concerns aggregation of information

- Decisions are based on events in the grid. Decisions can be made and supported by software on the right level.

- Computation nodes can be placed at the physical asset on each level (e.g. substation, transformer station)

5.2 Distributed intelligence

Intelligence should be distributed as much as possible. This has following advantages:

- Computational power is distributed over all nodes. This omits the need for large centralised computational power.

- Information availability increases with decreasing level. For example inside a MicroCHP (lowest level) all information about the device is available. On the home controller (one level higher) only a subset of this information is available. At the highest level probably no information at all is available on individual devices. Therefore, placing the intelligence at the lowest possible level provides these components with most information without the need for large communication bandwidth. Agent technology fits this paradigm.

5.3 Fail save mechanisms

By introducing systems like the Integral system, comfort of the consumer becomes dependent on complex processes and technology that implements them. Therefore provisions must be in place for guaranteeing comfort when processes or technology fail.

- If communication is interrupted, the nodes depending on it shall fall back to a default mode, providing comfort. Of course this might have a drawback on the optimal or most economical use of the system (graceful degradation).

- Operation of lower level nodes shall not stop if higher level nodes fail. Also here the lower level node shall fall back to a safe mode.

- Measures shall be taken to guarantee availability of critical parts of the system

- Watchdog mechanisms can be implemented to safeguard processes from crashing

- The Operator of the system shall be able to monitor the system and should be alerted if critical parts of the system fail.

5.4 Aggregation for scalability

Potentially millions of households can be attached to a Integral system. Therefore measures must be taken to make the system scalable. Aggregation has been applied as means for scalability. Information from lower level nodes is aggregated and sent passed towards higher levels.

5.5 Standard technology and open standards

In order to make system components from the highest level to the lowest level interoperable, open standards must be used. Using web services (based on HTTP, XML and SOAP messages) for communication is a good choice.

Using standard/common technologies, like .NET or JAVA and state-of-the-art development technologies (model driven architecture (MDA), test driven development (TDD), etc) results in well tested components that are maintainable.